МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
імені ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ

# ВСТУП ДО МАТЕМАТИЧНИХ ОСНОВ ЗАХИСТУ ІНФОРМАЦІЇ

С. Л. Кривий

Київ – 2022

ББК 22.17я73
К82

У навчальному посібнику розглядаються основні математичні поняття, на базі яких будуються криптографічні системи: основи теорії складності обчислень, алгебраїчні структури (групи, кільця, поля), елементи теорії чисел, теоретико-числові функції та алгоритми їх обчислення, елементи теорії ймовірностей та інформації. Описуються класичні системи обміну ключами, електронного підпису та взаємної автентифікації, а також симетричні та асиметричні криптографічні системи (класичні і сучасні). Розглядаються прості способи генерації випадкових чисел.

Для викладачів, аспірантів та студентів, старших курсів вищої школи, за напрямками "Комп'ютерні науки", "Програмна інженерія", "Програмне забезпечення систем".

Рецензенти:

Опанасенко В.М., доктор технічних наук, професор, провідний науковий співробітник, Інститут кібернетики імені В.М. Глушкова НАН України

Ганюшкін О.Г., кандидат фізико-математичних наук, доцент, Київський національний університет імені Тараса Шевченка

*Рекомендовано до друку рішенням Вченої ради*
*факультету комп'ютерних наук та кібернетики*
*Київського національного університету ім. Тараса Шевченка*
Протокол № 6 від 16 листопада 2021 року.

УДК 511+512.624.95+519.2:004.056.5(075.

**ISBN 978-966-00-1165-6**

# Зміст

# СПИСОК СКОРОЧЕНЬ

| | | |
|---|---|---|
| ВТ | – | відкритий текст |
| КК | – | комутативне кільце |
| АКК | – | асоціативно-комутативне кільце |
| АКК1 | – | асоціативно-комутативне кільце з одиницею |
| Клас | – | $P$ – часової поліноміальної складності |
| | – | $NP$ – часової експоненціальної складності |
| | – | $PSPACE$ поліноміальної складності за пам'яттю |
| | – | $NSPACE$ експоненціальної складності за пам'яттю |
| ЛКГ | – | лінійний конгруентний генератор псевдовипадкових чисел |
| МПД | – | метод пробних ділень |
| МС | – | математичний сейф |
| МТ | – | машина Тьюрінга |
| ДМТ | – | детермінована машина Тьюрінга |
| НДМТ | – | недетермінована машина Тьюрінга |
| НСД | – | найбільший спільний дільник |
| НСК | – | найменше спільне кратне |
| ШТ | – | шифрований текст (шифрограма) |
| DES | – | стандарт шифрування США (замінений шифром AES) |
| AES | – | стандарт шифрування США |
| OFB | – | генератор псевдовипадкових чисел |
| RSA | – | криптосистема Рівеста–Шаміра–Адлемана |
| $\mathcal{C}$ | – | множина комплексних чисел |
| $\mathcal{D}$ | – | множина дійсних чисел |
| $\mathcal{F}_p$ | – | поле лишків за модулем простого числа $p$ |
| $\mathcal{F}_{p^k}$ | – | поле лишків за модулем степеня простого числа $p$ |
| $G_k$ | – | асоціативно-комутативне кільце з одиницею порядку $k$ |
| $\mathcal{N}$ | – | множина натуральних чисел |
| $\mathcal{Q}$ | – | множина раціональних чисел |
| $\mathcal{Z}$ | – | множина цілих чисел |
| $\mathcal{Z}_m$ | – | кільце лишків за модулем числа $m$ |
| $\mathcal{Z}_{p^k}$ | – | примарне кільце лишків |
| $\varphi(n)$ | – | функція Ойлера – кількість чисел менших $n$ і взаємно простих з $n$ |
| $\lfloor x \rfloor$ | – | функція "низ"– найбільше ціле число, що не пепевищує $x$ |
| $\lceil x \rceil$ | – | функція "верх" – найменше ціле число, що перевищує $x$ |

# ПЕРЕДМОВА

У посібнику розглянуто елементарний вступ до математичних основ криптології. Ця наука включає два підрозділи – *криптографію* та *криптоаналіз*. Особливо актуальною є проблема захисту інформації, безпека її збереження та цілісності. Розв'язання цих проблем покладається на криптографію, а отримання несанкціонованого доступу до інформації – на криптоаналіз.

Оскільки людство вступило в еру, коли ділова переписка, фінансові потоки й обмін урядовими документами відбувається за допомогою відкритих комп'ютерних систем зв'язку, то пересічна людина, де б вона не перебувала, може скористатися такою інформацією не завжди з добрими намірами. У зв'язку з тим, що велика частина інформації за такого способу комунікації міститься в електронних сховищах і на електронних носіях, то проблема захисту від несанкціонованого доступу до такого типу інформації потребує спеціального дослідження та розроблення відповідних засобів такого захисту. Отже, для розроблення систем захисту інформації потрібні фахівці (системні адміністратори, криптографи, розробники), які у змозі їх створювати, супроводжувати та модернізувати програмне забезпечення, призначене для захисту інформації.

Серед великої сукупності засобів захисту на перший план виступають програмні засоби, оскільки вони набагато дешевші, ніж інші засоби захисту (апаратні, технічні тощо). Тому в цьому посібнику основу складають математичні підстави й алгоритмічні методи побудови такого типу систем.

Посібник структурно складається зі вступу та двох розділів. У вступі коротко представлено етапи розвитку криптографічних засобів та основні поняття криптології.

У першому розділі наведено основні математичні поняття теорії складності обчислень, теорії ймовірностей та інформації, теорії чисел, загальної алгебри.

У другому розділі надано характеристики симетричних і асиметричних криптографічних систем. Розглянуто також системи обміну ключами, електронного підпису й автентифікації абонентів.

*Автор*

# Розділ 1

# ВСТУП

Спочатку дамо неформальне означення головних об'єктів нашого посібника – інформації і проблеми її захисту.

Безпека діяльності людей потрібна практично для кожного підприємства, організації, банку, торгових мереж тощо. Різниця полягає тільки в тому, які методи будуть потрібні для забезпечення такої безпеки. У загальному випадку згідно з історичною і міжнародною практикою основними об'єктами, які потребують захисту (відповідно до їхніх пріоритетів), є такі:

1) люди; 2) інформація; 3) цінності матеріальні.

Якщо пріоритет захисту і безпеки людей є очевидним, то пріоритет захисту інформації над цінностями матеріальними потребує особливого розгляду.

Неформально під поняттям **інформація** розуміють обмін відомостями між людьми, людиною і автоматом, автоматом і автоматом, а також обмін сигналами в живому і рослинному світі.

**Проблема захисту** і безпеки інформації полягає в перетворенні її в такий спосіб, щоб вона не була доступна особам небажаним. Ця проблема є основним завданням науки, яка має назву **криптології**. Історія криптології є ровесницею історії природної мови, за допомогою якої виконується передача інформації від одного суб'єкта до іншого. Розвиток писемності привів до розвитку криптології як науки. На початку історії людства сама писемність була

в певному сенсі криптографічною системою, оскільки письмо було доступне лише особам вибраним. І протягом багатьох століть захист інформації використовувався в основному у процесі обміну повідомленнями між державними чиновниками і військовими. Тому коло людей, які займалися захистом повідомлень, було досить обмеженим, а самі методи захисту трималися в секреті. Ситуація змінилася з появою комп'ютерів і комп'ютерних систем, що привело до нового розвитку криптологічних систем. Тепер обмін інформацією відбувається за допомогою комп'ютерних систем зв'язку, таких як інтернет. Кожна людина, де б вона не перебувала, може скористатися цими можливостями, а це стало величезною перевагою таких систем. Основними сферами діяльності, де використовується комп'ютерний обмін інформацією, є такі:

1. Військова. 2. Дипломатична. 3. Фінансова.
4. Банківська. 5. Комерційна. 6. Промислова.
7. Наукова. 8. Юридична. 9. Медична.
10. Приватна таємниця.

Головною метою та задачами захисту інформації є:

1. Секретність інформації. 2. Цілісність інформації.
3. Автентичність інформації. 4. Доступність інформації.

До основних напрямків, аспектів, методів і засобів захисту інформації належать:

1. Юридичні, правові.
2. Методично-нормативні.
3. Організаційні.
4. Фізичні.
5. Технічні – захист від витоку по технічних каналах: електромагнітному, оптичному, акустичному, віброакустичному.
6. Стеганографічні.
7. Криптографічні. Методи математичного захисту інформації.
8. Методи квантової криптографії.
9. Морально-етичні норми.

З точки зору реалізації у методах 5-8 можна виокремити апаратні, програмні та програмно-апаратні засоби. Кожен із цих засобів має свої переваги та недоліки.

## 1.1. Криптографічний захист інформації

**Криптографічний захист інформації** – це різновид захисту, який реалізується за допомогою криптографічних перетворень, спеціальних ключових даних із метою приховування та відновлення змісту інформації, підтвердження достовірності, авторства, запобігання несанкціонованому використанню тощо.

**Криптографічне перетворення** – це перетворення інформації відповідно до певних правил (логічних, математичних) з метою забезпечення функціонування криптографічних протоколів.

**Криптографічний ключ** – це параметр, який використовується в криптографічному алгоритмі для вибору конкретного криптографічного перетворення; ключі можуть бути таємними або відкритими.

**Криптографічний протокол** – це послідовність узгоджених дій згідно з деякими правилами, відповідно до яких відбувається обмін інформацією між сторонами або учасниками протоколу та її перетворення з використанням криптографічних методів і засобів. Простий приклад криптографічного протоколу – це зашифрування та розшифрування повідомлення.

**Криптографія** – науково-технічна дисципліна, яка вивчає принципи, методи та засоби інформаційних технологій криптографічного захисту інформації, предметом якої є розроблення криптографічних систем.

**Криптоаналіз** – це науково-технічна дисципліна, яка вивчає методи, способи та засоби аналізу криптографічного захисту інформації (криптографічних систем, криптографічних алгоритмів і протоколів) з метою отримання несанкціонованого доступу до інформації без знання таємних ключів, розкриття будови криптосистем, підробки даних тощо. Криптоаналіз оцінює складність таких способів розкриття (злому) і стійкість криптографічного захисту інформації. Фахівця, який займається криптоаналізом називають **криптоаналітиком**.

**Криптологія**, за найбільш поширеною сучасною термінологією, об'єднує в собі дисципліни – *криптографію* і *криптоаналіз*.

### 1.1.1. Етапи розвитку криптографічних засобів

1. Криптографія заміни і перестановки (з давнини до середини і кінця XIX ст.).

а) Перший шифр перестановки, застосування якого зафіксовано у військовій справі, (Спарта, V ст. до н.е.) – шифр Скитала. У цьому шифрі використовувався круглий барабан (скитал), на який намотувалась стрічка. На намотану стрічку наносився текст повідомлення, а після розмотування стрічки з барабана на ній була зовні випадкова послідовність літер – шифроване повідомлення. Таємним ключем був діаметр барабана. Криптоаналіз шифру Скитала запропонував Арістотель за допомогою барабана змінного діаметра: якщо при намотуванні на нього стрічки із шифрованим повідомленням у деякому місці вгадувались якісь сенсові частини слів, то цьому місцю відповідав діаметр справжнього барабана.

б) Прикладом шифру заміни є шифр Цезаря – заміна кожної літери повідомлення на літеру циклічно віддалену в алфавіті на фіксоване число позицій (див. пункт 3.2.1).

2. Застосування телеграфу для шифрування та кодування (із середини XIX ст.).

3. Використання механічних пристроїв (кінець XIX ст. – 20-ті рр. XX ст.).

4. Електромеханічні пристрої (20-ті рр. XX ст. – середина XX ст.). Приклад – електромеханічний ротор (ENIGMA і PURPLE).

5. Електронні машини (з кінця 40-х р. XX ст.).

6. Напівпровідникові криптосистеми.

7. Криптосистеми, засновані на мікросхемах.

8. Застосування комп'ютерної техніки для криптографічного захисту.

9. Квантова криптографія.

До епохи Ренесансу криптографія була заняттям непрофесіональним. У ті часи шифри були здебільшого головоломками. Хоча тоді побудовано шифри, які не були зламані навіть через 100 і більше років! За сучаними поняттями, враховуючи залучення до криптоаналізу комп'ютерів, це були слабкі, нестійкі шифри. Велике

просування в криптографії сталося після залучення до цієї справи відомих математиків (Відродження – Ф. Вієт, Д. Кардано, середні віки К.Ф. Гаусс, Х. Гольдбах і ін.). Згодом стали створюватися спеціалізовані державні служби шифрування і розшифрування. Такі служби були створені, наприклад, Кромвелем у Англії, кардиналом Рішельє у Франції, Б. Хмельницьким в Україні, Петром I у Росії.

У 1883 р. вийшла книга Керкгоффа "Військова криптографія", в якій уперше були сформульовані певні вимоги до криптосистем, правила побудови, експлуатації, стійкості криптографічних пристроїв. Частина цих правил вважаються актуальними і в наші дні.

Наукою, у повному розумінні цього слова, криптографія стала визнаватися з 1949 р. після появи у відкритому друці статті Клода Шеннона "Теорія зв'язку в секретних системах" [22]. А з 1976 р. з появою алгоритмів Діффі і Хеллмана "Нові напрямки в криптографії" [29] почався новий етап у розвитку криптографії – застосування криптосистем із відкритим ключем, яке створило умови успішного розв'язання цілого ряду назрілих проблем криптографічного захисту інформації.

**Класифікація сучасних криптосистем**. Сучасні криптосистеми поділяють на декілька класів.

1. **Симетричні (одноключові, із секретним ключем)** системи є *блокові* та *потокові*. У відправника й одержувача повідомлення один і той самий секретний ключ, вони перебувають у рівних (симетричних) умовах, можуть як зашифрувати повідомлення, так і розшифрувати його за допомогою таємного ключа.

2. **Асиметричні (двоключові: з відкритим ключем (загальнодоступним) і секретним ключем)** системи стали відомі з 1976 р., активно використовуються на практиці з 1978 р. У найпростішому випадку мають два ключі: один (відкритий) – у відправника для шифрування, другий у одержувача (секретний) для розшифрування.

3. **Квантові** (перебувають на етапі експериментів). Квантові системи ґрунтуються на принципах роботи квантових комп'ютерів.

### 1.1.2. Квантовий комп'ютер

Квантовий комп'ютер складається із квантово-механічної системи, обов'язково ізольованої від навколишнього середовища таким чином, щоб її поведінкою можна було ззовні керувати так, аби жодна подія, яка не пов'язана з процедурами контролю, не мала змоги змінити цю поведінку.

Модель для такої системи включає:

– простір станів системи, асоційований з ізольованою квантово-механічною системою, є векторним простором над полем комплексних чисел із визначеним скалярним добутком (гільбертів простір);

– стани системи в довільний момент часу, який повністю описується вектором стану і є одиничним вектором у просторі станів;

– зміна стану замкнутої квантово-механічної системи описується тільки унітарним перетворенням, причому стани системи в різні моменти часу зв'язані унітарним перетворенням, яке залежить тільки від моментів часу;

– вимірювання квантової системи складається з набору лінійних операторів, що діють на просторі станів системи, і фактично є проєкцією на ортогональні підпростори.

Принцип квантових обчислень ґрунтується на таких кроках.

Виконується послідовність унітарних операцій простого типу над одним, двома або трьома **кубітами**, яка контролюється класичним комп'ютером. У кінці обчислень стан квантового комп'ютера вимірюється, що і дає шуканий результат обчислень.

Ідея квантових обчислень, уперше висловлена Ю. І. Маніним і Р. Фейнманом, полягає в тому, що квантова система із $L$ дворівневих квантових елементів (квантових бітів, кубітів) має $2^L$ лінійно незалежних станів і, відповідно до принципу квантової суперпозиції, простір станів такого квантового регістра є $2^L$-вимірним гільбертовим простором. Операція в квантових обчисленнях відповідає повороту вектора стану регістра в цьому просторі. Отже, квантовий пристрій обчислень розміру $L$ кубіт фактично задіює одночасно $2^L$ класичних станів.

За допомогою базових квантових операцій можна симулювати роботу звичних логічних елементів, з яких побудовано класичні комп'ютери. Тому довільну задачу, яка розв'язується на класичних комп'ютерах, квантовий комп'ютер також розв'яже. Отже, нова схема обчислень буде не слабкішою нинішньої.

Чим же квантовий комп'ютер кращий класичного? Більшість сучасних комп'ютерів працюють за такою схемою: $n$ бітів пам'яті зберігають стан і в кожний такт часу ці стани змінюються процесором. У випадку квантового комп'ютера система з $n$ кубітів знаходиться в стані, який є суперпозицією всіх базових станів і тому зміна в системі стосується всіх $2^n$ базових станів одночасно. Теоретично нова схема може працювати набагато швидше (в експоненціальне число разів) класичної схеми. Практичний алгоритм пошуку в базі даних показує квадратичний приріст обчислювальної потужності порівняно з класичними алгоритмами.

**Проблеми квантових обчислень**, які виникають на шляху використання квантових комп'ютерів, зводяться до таких:

Декогеренція (нестабільність) при обчисленнях.

Масштабування (мала кількість кубітів).

Компактність.

Висока вартість і доступність.

**Практичні досягнення** на шляху розвитку квантових комп'ютерів і обчислень:

2018 – 72 qubits, Google, US (Bristlecone 5 березня 2018);

2017 – 50 qubits, IBM, US;

2017 – 17 qubits, IBM, US;

2006 – 12 qubits, Institute for Quantum Computing, Perimeter Institute for Theoretical Physics, and MIT US;

2000 – 7 qubits, Los Alamos National Laboratory, US;

2000 – 5 qubits, Technical University of Munich, Germany;

1998 – 2 qubits, IBM, UC Berkeley, Stanford University, and MIT, US;

1998 – 2 qubits, Oxford University, UK.

### 1.1.3. Основні поняття криптології

*Відкритий текст (ВТ)* – це повідомлення, дані, елемент простору повідомлень, до якого застосовується процедура криптографічного перетворення, шифрування. Під ВТ розуміють текст, заданий у вигляді послідовності символів скінченного алфавіту, з доступним семантичним змістом. ВТ отримують після правильного розшифрування.

*Шифрований текст, шифротекст або криптограма* (ШТ) – це інформація, яка одержана після застосування до відкритого тексту процедури зашифрування.

*Зашифрування* – це криптографічне перетворення відкритого тексту із застосуванням таємних ключів, у результаті якого дістають ШТ з недоступним для незаконного користувача семантичним змістом.

*Розшифрування* – зворотне криптографічне перетворення ШТ із застосуванням таємних ключів, у результаті якого законний користувач дістає ВТ, що був зашифрований. Для даного перетворення також використовується термін *дешифрування*.

*Шифратор* – пристрій, що здійснює процедуру зашифрування.

*Дешифратор* – пристрій, що виконує процедуру розшифрування.

**Криптографічна система** – система безпеки інформації з використанням криптографічних методів, яка повинна забезпечувати збереження конфіденційності, цілісності, автентичності. З практичної точки зору – це набір апаратних і (або) програмних засобів, інструкцій і правил, за допомогою яких, використовуючи криптографічні перетворення, можна зашифрувати повідомлення і розшифрувати шифрограму різними способами, один з яких вибирається за допомогою секретного ключа, а також виконувати інші криптографічні протоколи. Математичну модель криптографічної системи буде розглянуто далі.

*Криптографічна стійкість* у широкому розумінні – це здатність криптосистеми або криптоалгоритму протистояти атакам із використанням методів криптоаналізу; у вузькому розумінні (практична стійкість) – це чисельна характеристика часової та ємнісної скла-

дності розкриття криптографічного алгоритму з урахуванням тих науково-технічних методів і засобів, які може використати криптоаналітик.

*Теоретична стійкість* у широкому розумінні – це стійкість криптосистеми за наявності у криптоаналітика необмеженого часу, необмежених обчислювальних ресурсів, якнайкращих методів криптоаналізу, у вузькому розумінні – це в певному сенсі гарантована стійкість. Основні підходи до визначення поняття теоретичної стійкості нині розглядаються у межах деяких математичних моделей. Так, розглядається стійкість теоретико-інформаційна, теоретико-складнісна, математично доведена.

*Практична стійкість* – це стійкість криптосистеми на даний час з урахуванням того, що криптоаналітик володіє обмеженим часом, обмеженими обчислювальними ресурсами і сучасними методами криптоаналізу, а також чисельна характеристика часової та ємнісної складності розкриття криптографічного алгоритму.

### 1.1.4. Основні види криптографічних атак

Види криптографічних атак (нападів) залежать від типу інформації, відомої криптоаналітику.

У криптології загальноприйняте правило Керкгоффа:
**стійкість криптосистеми не повинна опиратися на секретність її будови й алгоритм шифрування, а має ґрунтуватися на секретності ключа (при надійному алгоритмі шифрування і достатньому розмірі ключа)**.

Залежно від додаткової інформації атаки класифікують у порядку їхнього посилення наступним чином.

1. *Атака з використовуванням тільки шифротексту.* У криптоаналітика є шифротексти декількох повідомлень, зашифрованих одним і тим самим алгоритмом шифрування і невідомим ключем (ключами). Задача криптоаналітика полягає в розкритті як можна більшого числа повідомлень або, що краще, отриманні ключа (ключів), використаних для шифрування повідомлень із метою розши-

фрування також і інших повідомлень, зашифрованих тими ж ключами.

2. *Атака з використанням відкритого тексту.* У криптоаналітика є доступ не лише до шифротекстів декількох повідомлень, але і до відповідних відкритих текстів цих повідомлень. Його завдання полягає в отриманні ключа (або ключів), використаного (використаних) для шифрування повідомлень, з метою розшифрування інших повідомлень, зашифрованих тим же ключем (ключами).

3. *Атака на основі вибраного відкритого тексту.* У криптоаналітика є не тільки доступ до шифротекстів і відповідних відкритих текстів декількох повідомлень, але й можливість вибирати відкритий текст (тексти) і отримати шифрований (шифровані). Це надає більше варіантів, ніж атака з використанням відкритого тексту, оскільки криптоаналітик буде вибирати відкриті тексти зі спеціальними властивостями, що може надати більше інформації про ключ. Його завдання полягає в отриманні ключа (або ключів), використаного для шифрування повідомлень, або алгоритму, що дозволяє розшифрувати нові повідомлення, зашифровані тим же ключем (або ключами).

4. *Адаптивна атака з використанням відкритого тексту.* Криптоаналітик не тільки може вибирати тексти для шифрування, але також може будувати свій подальший вибір текстів на базі одержаних результатів шифрування попередніх. При розкритті з використанням вибраного відкритого тексту криптоаналітик може вибрати для шифрування тільки один або декілька ВТ одночасно для отримання відповідних ШТ, при адаптивному нападі з використанням вибраного відкритого тексту він може вибрати спочатку один ВТ і отримати криптограму, потім вибрати наступний ВТ, використовуючи результати першого вибору та шифрування, і так далі. Атаки 2-4 можливі при шифруванні з відкритим ключем.

5. *Атака на основі вибраного шифротексту.* Криптоаналітик має можливість вибирати різні шифротексти для розшифрування і має доступ до розшифрованих відкритих текстів (наприклад, криптоаналітик має доступ до шифрувального апарата).

6. *Адаптивна атака на основі вибраного шифротексту.* Кри-

птоаналітик має можливість для ШТ, що послідовно вибираються з урахуванням попередніх результатів, отримувати відповідні ВТ (аналогічно п. 4). Задача знайти таємний ключ, або розшифрувати повідомлення. Атаки п. 5, 6 особливо небезпечні для криптосистем із відкритим ключем.

7. *Атака на основі вибраного тексту* – об'єднує можливості атак п. 3, п. 5.

8. *Адаптивна атака на основі вибраного тексту* – об'єднує можливості атак п. 4, п. 6.

Атаки в цьому списку з більшим номером загалом сильніші й небезпечніші, ніж із меншим. Для всіх сучасних шифраторів обов'язкова вимога – стійкість до атак типу 1 і 2. Якщо у криптоаналітика є деяка додаткова інформація про ключі (окрім їхньої довжини) або про зв'язок між різними ключами, то напади на криптосистему стають ще небезпечнішими. До такої атаки належить, зокрема, атака з використанням інформації з так званого побічного каналу, що міститься, наприклад, в електромагнітному випромінюванні шифратора.

Для ілюстрації введених понять і понять, які використовуються в задачах критографії, розглянемо приклад нескладної криптографічної задачі.

**Задача**. Аліса і Боб хочуть провести вечір разом, але не можуть вирішити куди йти: в кінотеатр чи в театр. З метою знаходження відповіді на це питання вони вирішили провести жеребкування шляхом *підкидання монети* (цей спосіб добре відомий всім).

Аліса бере монету і говорить Бобу: "Ти вибираєш сторону монети, а я її підкидаю". Боб згоден і Аліса підкидає монету. Потім вони обоє дивляться на результат підкидання. Якщо результат той, що вибрав Боб, то він вирішує куди йти, інакше – це вирішує Аліса.

Вищеописана процедура обміну інформацією між Алісою і Бобом є багатосторонньою грою, яка має спеціальну назву – **протокол**. Протокол – це точно визначена послідовність дій, які виконуються декількома учасниками. Наявність декількох учасників є суттєвою умовою, оскільки, коли вся процедура виконується

одним учасником, то її назвати протоколом не можна.

**Задача (продовження)**. Уявімо тепер, що Аліса і Боб виконують наведений у задачі протокол по телефону. Аліса пропонує Бобу: "Вибери сторону монети, а я її підкину і скажу хто виграв." Очевидно, що Боб не погодиться на такі умови, тому що він не в змозі перевірити результат жеребкування.

Як мусить поступити Боб у цій ситуації?

Розв'язати проблему Бобу допомагає криптографія. Будемо розглядати результат жеребкування як значення функції $f : \mathcal{Z} \to \mathcal{Z}$, яка визначена на множині цілих чисел $\mathcal{Z}$ зі значеннями в цій же множині і яка має такі властивості.

**Властивість 1**. *Для функції* $f$

а) *і кожного числа* $x \in \mathcal{Z}$ *можна легко обчислити значення* $f(x)$, *але за значенням* $f(x)$ *неможливо знайти її аргумент* $x$;

б) *неможливо знайти пару чисел* $(x, y)$ *таких, що* $x \neq y$ *і* $f(x) = f(y)$.

В умові а) використовуються слова "легко" і "неможливо". Сенс цих слів буде строго означений далі засобами теорії складності обчислень [23], а тут ці слова виражають рівень складності певної дії. Припустимо, що Аліса і Боб погодилися використовувати функцію $f(x)$ і що парне число означає ОРЕЛ, а непарне – РЕШКА. Тепер вони готові виконати такий протокол – протокол 1.

**Протокол 1 (підкидання монети по телефону)**

*Початкові умови*:

а1) функція $f(x)$ має властивість 1.

а2) якщо $x$ – парне, то $f(x)$ означає ОРЕЛ, інакше – РЕШКА.

1. Аліса вибирає велике випадково згенероване ціле число $x$, обчислює значення $f(x)$ і повідомляє це значення Бобу по телефону.

2. Боб повідомляє свою догадку про число $x$: парне чи непарне.

3. Аліса повідомляє Бобу число $x$.

4. Боб перевіряє значення $f(x)$ і переконується у правильності своєї догадки.

Очевидно, що протокол 1 цілком працездатний. Але одразу виникає низка питань, одне з яких є таким: "Чи надійний цей протокол?" Це питання в криптографії пов'язане з питанням "аналізу стійкості протоколу". Друге питання "Чи достатньо потужний генератор випадкових чисел, який використовує Аліса?" Якість такого генератора надзвичайно важлива для багатьох серйозних застосувань, де необхідно приймати правильні рішення.

Простий аналіз стійкості протоколу 1 показує таке.

По-перше, властивість б) функції $f(x)$ означає, що Аліса не може знайти два різні числа $x$ і $y$, одне з яких парне, а друге непарне. Отже, повідомивши Бобу значення $f(x)$ (крок 1), Аліса фіксує свій вибір числа $x$ і не може від нього відмовитися. На цьому жеребкування закінчується.

По-друге, завдяки властивості а) функції $f(x)$, Боб не може визначити число $x$, яке відоме Алісі. Отже, припущення Боба на кроці 2 протоколу 1 дійсно є догадкою, оскільки в нього відсутня інформація про число $x$. На цьому кроці Аліса може переконати Боба в правильності його догадки, назвавши число $x$ (крок 3). Боб може перевірити це самостійно, знайшовши значення $f(x)$ (крок 4) і порівнявши його із числом, яке повідомила Аліса.

Зауважимо, що жеребкування вважається коректним, якщо число $x$ вибирається з достатньо великої множини, так щоб імовірність вгадати парність числа $x$ не перевищувала 50 відсотків.

Аналіз протоколу 1 показує, що багато чого в ньому спрощено і багато деталей не враховано. У результаті наведена версія протоколу 1 не є досконалою. Деякі деталі аналізу протоколів будуть розглянуті далі, тому що для цього необхідно познайомитися з певними математичними поняттями і методами.

Таким чином, структура посібника схематично набуває вигляду, як на рис. 1.1.1.

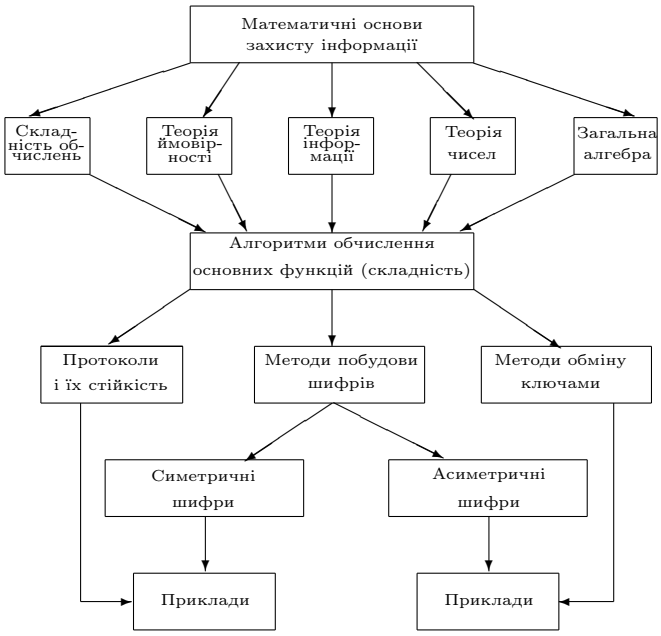



Рис. 1.1.1. Схема зв'язків розділів посібника

За цією схемою і йтиме виклад математичних основ захисту інформації у посібнику, матеріал якого розраховано на двосеместровий курс із математичних основ криптографічного захисту інформації.

**Контрольні запитання**

1. Які мета та задачі захисту інформації?
2. Назвати напрямки та методи захисту інформації.
3. Що таке криптографічний ключ?
4. З яких частин складається криптологія?
5. Чи були відомі способи захисту інформації до нашої ери?
6. Яка дата вважається початком розвитку криптографії з відкритим ключем?
7. Які завдання реалізує стеганографія?
8. Які умови має задовольняти криптографічний протокол?
9. Які умови має задовольняти криптографічна функція?
10. Яка різниця між симетричними та асиметричними криптосистемами?
11. Дати характеристику основних атак на криптосистеми.
12. Сформулювати криптологічне правило Керкгоффа.

# Розділ 2

# МАТЕМАТИЧНІ ОСНОВИ

## 2.1. Множини

**Інтуїтивне поняття множини.** Як відомо, поняття множини належить до аксіоматичних понять математики і точне його означення дати неможливо. Часто приймають означення інтуїтивного поняття множини Георга Кантора – основоположника цієї теорії.

*Довільне зібрання певних предметів нашої інтуїції чи інтелекту, які можна відрізнити один від одного і які уявляються як єдине ціле, називається множиною. Предмети, які входять до складу множини, називаються її елементами.*

Із цього означення випливає, що довільна множина повністю визначається своїми елементами. Отже,

**дві множини рівні, якщо кожний елемент однієї з них є елементом другої і навпаки (принцип об'ємності)**.

Множина називається **скінченною**, якщо вона складається зі скінченного числа елементів. Запис $a \in A$ ($a \notin A$) означає, що $a$ є (не є) елементом множини $A$. Однозначно визначена множина, елементами якої є $a_1, a_2, \ldots, a_n$, позначається $\{a_1, a_2, \ldots, a_n\}$.

Подання множини за допомогою фігурних дужок з явним переліком її елементів можливе лише тоді, коли множина має невелику кількість елементів. Якщо ж множина має хоча й скінченну, але велику кількість елементів, то таке подання множини стає досить

громіздким, а у випадку нескінченної множини його застосування взагалі неможливе. У такому випадку задають множини за допомогою властивості її елементів.

Під **властивістю** відносно предмета $x$ розуміють таке розповідне речення, в якому щось стверджується відносно $x$ і яке можна характеризувати як істинне або хибне щодо $x$.

**Приклад 2.1.1.** Властивостями є такі записи:

1) 3 ділить 10; 2) $x < x$; 3) $x^2 = 2$; 4) $x^2 + 1 > 0$.

Наведені нижче вирази не є властивостями:

5) стійте тут; 6) чому важлива математична індукція,

тому що їх не можна характеризувати як істинні чи хибні. ♠

Нехай $P(x)$ означає деяку властивість, тоді $P(a)$ буде означати цю властивість, але із заміною $x$ на $a$. Задання множини в термінах властивостей досягається за допомогою такого принципу згортання.

**Довільна властивість $P(x)$ визначає множину $A$ умовою: елементами множини $A$ є ті і тільки ті предмети $a$, які задовольняють властивість $P$.**

Отже, довільна властивість $P(x)$ визначає єдину множину, яку позначають $\{a|\ P(a)\}$ і читають так: "множина всіх тих предметів $a$, що $P(a)$". Зауважимо, що властивість $P$ може бути способом побудови елементів множини $\{a|\ P(a)\}$.

Безпосередньо з принципу згортання випливає існування такої множини. Нехай $A$ – деяка множина, а $P(x)$ має вигляд $x \neq x$, тоді множина $\{a \in A|P(a)\} = \{a \in A|a \neq a\}$, очевидно, не має елементів. Із принципу об'ємності випливає, що може існувати лише одна множина, яка не має елементів. Ця множина називається **пустою множиною** і позначається $\emptyset$.

### 2.1.1. Операції, алфавіти, мови

Уведемо символи $\Leftrightarrow, \exists x, \forall x, \Rightarrow$, які надалі будуть служити для скорочення виразів *"тоді і тільки тоді, коли", "існує x такий, що", "для кожного x" і "випливає"* відповідно.

**Означення 1.** *Множина $A$ називається* **підмножиною** *множини $B$ ($A \subseteq B$), якщо всі елементи множини $A$ є також елементами множини $B$ ($A \subseteq B \Leftrightarrow (a \in A \Rightarrow a \in B)$). Множина $B$ називається* **надмножиною** *множини $A$.*

Тепер принцип об'ємності можна записати так:

$$A = B \Leftrightarrow (A \subseteq B \text{ і } B \subseteq A).$$

Уведемо знак строгого включення $\subset$ для множин. $A \subset B$ означає для множин $A$ і $B$, що $A \subseteq B$ і $A$ не рівна $B$ ($A \neq B$). Якщо $A \subset B$, то множина $A$ називається **власною підмножиною** множини $B$, а множина $B$ – **власною надмножиною** множини $A$.

Основними операціями на множинах є об'єднання, перетин, різниця та її окремий випадок – доповнення.

**Означення 2. Об'єднанням $A \cup B$ множин** $A$ і $B$ *називається множина, $A \cup B = \{a | a \in A$ або $a \in B\}$.*

**Перетином множин** *$A$ і $B$ називається множина, $A \cap B = \{a | a \in A$ і $a \in B\}$. Якщо $A \cap B = \emptyset$, то множини $A$ і $B$ називаються такими, що* **не перетинаються**.

**Різницею $B \setminus A$ множин** *$B$ і $A$ називається множина $\{a | a \in B$ і $a \notin A\}$. Очевидно, що $B \setminus A = B \setminus (A \cap B)$.*

*Якщо $A \subseteq B$, то $B \setminus A$ називається* **доповненням** *множини $A$ в множині $B$ і позначається $\bar{A}_B$ або просто $\bar{A}$, коли $B$ можна визначити з контексту.*

**Симетричною різницею $A \div B$ множин** *$A$ і $B$ називається множина $(A \setminus B) \cup (B \setminus A)$.*

З означення операції $A \div B$ очевидним чином випливає така рівність: $A \div B = (A \cup B) \setminus (A \cap B)$.

**Приклади деяких спеціальних множин**.

1) **Алфавітом** $X$ називається скінченна непуста множина попарно різних між собою елементів, які називаються **літерами** або **символами**: $X = \{x_1, x_2, \ldots, x_n\}$. **Словом** в алфавіті $X$ називається довільна скінченна послідовність літер цього алфавіту. У криптографії слова довжини $n$ називаються $n$-грамами, наприклад, якщо $n = 2$ – біграми, якщо $n = 3$ – триграми і т. д.

Множина всіх слів скінченної довжини в алфавіті $X$ позначається $F(X)$.

Якщо $p \in F(X)$ – слово, то його **довжиною** називається кількість літер алфавіту $X$, з яких воно складається, рахуючи кожне входження літери. Будемо позначати довжину слова $p$ через *l(p)*. Наприклад, якщо $X = \{a, b\}$, то $l(aab) = 3$, $l(baabb) = 5$.

Якщо $p = y_1 y_2 \dots y_k$ і $q = z_1 z_2 \dots z_r$ – слова в алфавіті $X$, то $p = q \Leftrightarrow k = r$ і $y_i = z_i$ для всіх $i = 1, 2, \dots, k$.

Довільна підмножина множини $F(X)$ називається **мовою в алфавіті** $X$. Наприклад,

а) нехай $X = \{a, b, c, d, ..., z\}$ складається з 26 літер англійського алфавіту. Довільна послідовність літер з $X$ належить до $F(X)$. Отже, $F(X)$ включає слова *math*, *is*, *fun*, *men*, *amour* і т. д. Оскільки $F(X)$ включає множину слів $a, aa, aaa, aaaa, aaaaa, ...$, то $F(X)$ є нескінченною множиною.

б) якщо $X = \{0, 1\}$, то множина $B$ тих слів із множини $F(X)$, які починаються з 1 є множиною двійкових (бінарних) цілих додатних чисел, тобто $B = \{1, 10, 11, 100, 101, 110, 111, 1000, 1001, ...\}$.

Уведемо до множини $F(X)$ деяке спеціальне слово, яке певною мірою аналогічне до пустої множини і яке називають *пустим словом*: пусте слово за означенням не включає жодного символу алфавіту $X$ і позначається через $e$ (отже, довжина $l(e) = 0$). Наприклад,

в) якщо $X = \{a\}$, то $F(X) = \{e, a, aa, aaa, aaaa, aaaaa, ...\}$;

г) якщо $X = \{a, b\}$, то $F(X) = \{e, a, b, aa, ab, ba, bb, aaa, aab, ...\}$;

д) якщо $X = \{0, 1, 2\}$, то $F(X) = \{e, 0, 1, 2, 00, 01, 02, 10, 11, 12, 20, 21, 22, 000, ...\}$.

**Конкатенацією** двох слів $p$ і $q$ в алфавіті $X$ називається слово $pq$, одержане в результаті приписування праворуч до слова $p$ слова $q$. Очевидно, що для пустого слова справедливі такі рівності: $\forall p \in F(X)(ep = pe = p)$.

Нехай дано слово $q = uw$, де $u, w \in F(X)$. Слово $u$ називається **початком** або **префіксом** слова $q$, а слово $w$ – **кінцем** або **суфіксом** слова $q$. Слово $p$ є **підсловом** слова $q$ в алфавіті $X$, якщо $q = p'pq'$, де $p', q'$ – деякі підходящі слова з $F(X)$. Якщо $p'$ є словом найменшої можливої довжини, то входження слова $p$ в слово $q$ називається **першим**. Аналогічно можна означити друге входження слова, третє і т. д. Наприклад, якщо $X = \{a, b\}$, то

$$L = \{p \in F(X) | l(p) = 2\} = \{aa, ab, ba, bb\};$$

якщо $L' = \{p \in F(X) | l(p)$ є числом парним $\}$, то

$$L' = \{aa, ab, ba, bb, aaaa, aaab, aabb, abbb, ...\}.$$

Зазначимо, що $L$ є підмножиною множини $L'$. Слово $p = baac$ є підсловом слова $q = abaacb$. Слова $e, a, ab, aba, abaa, abaac, abaacb$ є префіксами слова $q$, а слова $e, b, cb, acb, aacb, baacb, abaacb$ є суфіксами слова $q$.

2) Множина натуральних чисел $\{0, 1, 2, \dots, n, \dots\}$, яку позначають буквою $\mathcal{N}$, задається за допомогою принципу згортання з використанням поняття мови. Нехай $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ – множина з десяти елементів, яка складає алфавіт. Словами в цьому алфавіті є довільна скінченна послідовність символів з $A$. Наприклад, $p = 0020034985700$ – слово в алфавіті $A$, а $p = 00a26543c$ не є словом в алфавіті $A$, оскільки до його запису входять символи (символи $a, c$), що не належать множині $A$. Тобто $p$ буде словом в алфавіті $A \Leftrightarrow$ кожний його символ належить алфавіту $A$.

Тоді множина $\mathcal{N} = \{p = xy \cdots z | x, y, \dots z \in A\}$ складає множину натуральних чисел. Зауважимо, що слово $p = 00 \cdots 0$ і $p' = 0$ – одне і те саме число 0, так само, як і числа 00057 і 57. Отже, запис одного і того самого елемента множини $\mathcal{N}$ може

бути різним. Для однозначності запису чисел необхідно поставити вимогу, щоб перший символ у слові $p$, яке складається більше ніж з одного символу, був відмінний від нуля, тобто $\mathcal{N} = \{p = xy\ldots z | x, y, \ldots, z \in A$ і $x \neq 0$ або 0, якщо $p = 0\}$.

3) Множина натуральних додатних чисел $\mathcal{N}^+ = \mathcal{N} \setminus \{0\} = \{1, 2, \ldots, n, \ldots\}$.

4) Множина цілих чисел $\mathcal{Z} = \{\ldots, -n, \ldots, -2, -1, 0, 1, 2, \ldots\ \ldots, n, \ldots\}$ – це множина слів у алфавіті $B = \{-, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ таких, що $\mathcal{Z} = \{p = xy\ldots \ldots z | x, y, \ldots, z \in B$ і $x \neq 0$ і $y, \ldots, z \neq -$ або 0, якщо $p = 0\}$. Слово, першим символом якого є символ "–", називається від'ємним, а слово, першим символом якого є символ, відмінний від символу "–", називається додатним.

5) Множина раціональних чисел $\mathcal{Q}$ складається із множини всіх цілих чисел і всіх нескорочуваних раціональних дробів вигляду $m/n$, де $m, n \in \mathcal{Z}, n \neq 0$, тобто $\mathcal{Q} = \{m/n | m, n \in \mathcal{Z}\ i\ n \neq 0\}$. ♠

Зазначимо, що для довільної множини $S$ із $x \in S$ випливає $x \in S$ і тому $S \subseteq S$. Це означає, що довільна множина є своєю підмножиною. Щоб розрізняти підмножини і власні підмножини і вводилися позначення $\subseteq$ і $\subset$. Очевидно, що пуста множина є підмножиною довільної множини $A$.

Ще одна важлива множина, яка буде зустрічатися: *для будь-якої множини $U$ існує множина $B(U)$ всіх підмножин множини $U$, яка називається* **булеаном**. *Множину $B(U)$ також називають* **множиною-степенем** *множини $U$.*

Неважко показати, що булеан $B(S)$ $n$-елементної множини $S$ має $2^n$ елементів. Якщо множина $S$ нескінченна, то її булеан $B(S)$ теж нескінченна множина.

## 2.2. Відношення і функції

Нехай $A$ і $B$ – дві множини. Розглянемо множину вигляду

$$C = \{(a, b) | a \in A, b \in B\}.$$

Ця множина називається **декартовим добутком множин** $A$ і $B$ і позначається $A \times B$. Якщо множини $A$ і $B$ скінченні та мають відповідно $m$ і $n$ елементів, то очевидно, що $C$ матиме $m \cdot n$ елементів.

Самостійний інтерес складає випадок, коли множини $A$ і $B$ рівні між собою: $A = B$. Для його розгляду введемо поняття упорядкованої пари.

*Упорядкованою парою* елементів множини $A$ будемо називати об'єкт $(a, a')$, що складається з двох, не обов'язково різних, елементів $a$ і $a'$ множини $A$, де вказано який із них потрібно вважати першим, а який – другим. Множина $C = \{(a, a')|a, a' \in A\}$ всіх упорядкованих пар із множини $A$ називається **декартовим квадратом** множини $A$ і позначається $A^2$.

Наведені означення декартового добутку двох множин і декартового квадрата множини узагальнюються і на випадок довільної скінченної сукупності множин.

**Означення 3. Декартовим добутком** $A_1 \times A_2 \times \ldots \times A_n$ *множин* $A_1, A_2, \ldots, A_n$ *називається сукупність послідовностей* (*сукупність упорядкованих* $n$*-ок елементів*) $(a_1, a_2, \ldots, a_n)$, *де* $a_i \in$ $\in A_i, 1 \leq i \leq n$.

*Довільна підмножина* $R$ *множини* $A_1 \times A_2 \times \ldots \ldots \times A_n$ *називається* **відношенням**, *заданим або визначеним на множинах* $A_1, A_2, \ldots, A_n$. *Якщо* $A_1 = A_2 = \ldots = A_n = A$, *то декартів добуток* $A_1 \times A_2 \times \ldots \times A_n$ *називається* **декартовим добутком** $n$**-го степеня множини** $A(A^n)$, *а відношення* $R$, *задане на множинах* $A_1, A_2, \ldots, A_n$, – $n$**-арним відношенням** *на множині* $A$.

Зокрема, при $n = 1$ відношення називається *унарним*, при $n =$ $= 2$ – *бінарним*, при $n = 3$ – *тернарним* і т. д.

Коли $(a_1, a_2, \ldots, a_n) \in R$, то говорять, що елементи $a_i$ $(i = 1,$ $2, \ldots, n)$ знаходяться у відношенні $R$ або що відношення $R$ **істинне** для $a_1, a_2, \ldots, a_n$. Якщо $(a_1, a_2, \ldots, a_n) \notin R$, то говорять, що $R$ **хибне** для $a_1, a_2, \ldots, a_n$. Якщо $R = A_1 \times A_2 \times \ldots \times A_n$, то відношення $R$ називається **тотожно істинним**, а коли $R = \emptyset$, то відношення $R$ називається **тотожно хибним**.

### 2.2.1. Бінарні відношення. Основні властивості

Частіше інших як у теорії, так і на практиці використовуються бінарні відношення. Якщо $R$ – бінарне відношення, то запис $aRb$ означає, що $(a, b) \in R$, тобто що $R$ істинне для $a, b$.

Оскільки відношення є множинами, то до них застосовуються операції об'єднання, перетину, різниці та доповнення. Але крім цих

операцій на бінарних відношеннях визначаються ще дві операції.

**Означення 4.** *Нехай дано бінарні відношення* $R \subseteq A \times B$, $R_1 \subseteq$ $\subseteq B \times C$. *Відношення* $R^{-1} = \{(b,a)|aRb\}$, *задане на множині* $B \times A$, *називають* **оберненим** *до відношення* $R$, *а відношення* $R * R_1 =$ $= \{(a,c)|a \in A, c \in C \text{ і } (\exists b \in B)\ aRb \text{ і } bR_1c\}$, *задане на множині* $A \times C$, – **добутком**, *або* **суперпозицією** *відношень* $R$ і $R_1$.

Розглянемо різновиди бінарних відношень.

**Відношення тотожності.** Бінарне відношення **тотожності**, задане на множині $A$, складається точно з усіх пар вигляду $(a,a)$, де $a \in A$, і позначається через $i_A$ або просто $i$, якщо $A$ фіксовано. Пари $(a,a)$ називають *діагональними*, а відношення $i_A$ – **діагоналлю**. Очевидно, що для довільного бінарного відношення $R$, визначеного на множині $A$, справедлива рівність $i_A * R = R * i_A = R$.

**Рефлексивні відношення.** Бінарне відношення $R$, задане на множині $A$, називається **рефлексивним**, якщо $i_A \subseteq R$, тобто, коли воно включає діагональ.

**Антирефлексивні відношення.** Бінарне відношення $R$ називається **антирефлексивним**, якщо $aRa$ хибне для довільного елемента $a \in A$. Антирефлексивні відношення також називаються **ірреflексивними** відношеннями. Наприклад, відношення $a < a$ хибне на всіх числових множинах для довільного числа $a$.

**Симетричні відношення.** Бінарне відношення $R$, задане на множині $A$, називається **симетричним**, якщо із $aRb$ випливає $bRa$ $(R \subseteq R^{-1})$.

**Антисиметричні відношення.** Бінарне відношення $R$, задане на множині $A$, називається **антисиметричним**, якщо із $aRb$ і $bRa$ випливає $a = b$ $(R \cap R^{-1} \subseteq i_A)$.

**Транзитивні відношення.** Бінарне відношення $R$, задане на множині $A$, називається **транзитивним**, якщо з $aRb$ і $bRc$ випливає $aRc$ $(R * R \subseteq R)$.

### 2.2.2. Відношення еквівалентності

Бінарне відношення $R$, задане на множині $A$, називається **відношенням еквівалентності**, або просто **еквівалентністю** на $A$,

якщо воно рефлексивне, симетричне і транзитивне, тобто, якщо для довільних елементів $a, b, c$ із $A$ справедливі властивості:

а) $aRa$ ($i_A \subseteq R$) (рефлексивність);

б) $aRb \Rightarrow bRa$ ($R \subseteq R^{-1}$) (симетричність);

в) $aRb$ і $bRc \Rightarrow aRc$ ($R^2 \subseteq R$) (транзитивність),

де $i_A$ – відношення тотожності, а $R^2 = R * R$.

Неважко показати, що умови а), б), в) еквівалентні таким: $i_A \subseteq \subseteq R, R = R^{-1}, R^2 = R$.

Відношення еквівалентності, задане на множині $A$, тісно пов'язане з розбиттям множини $A$ на класи. Цей зв'язок виражається таким твердженням.

**Теорема 1.** *Між розбиттями множини на класи і відношеннями еквівалентності, заданими на цій множині, існує взаємно однозначна відповідність.*

Якщо $R$ – еквівалентність на $A$, то класи розбиття, визначені відношенням $R$, називають **класами еквівалентності** відношення $R$, а множину всіх класів розбиття – **фактор-множиною** множини $A$ і позначають її $A/R$. Число класів еквівалентності відношення еквівалентності $R$ називається **індексом** множини $A$. Коли число класів еквівалентності скінченне, то множина $A$ називається **множиною скінченного індексу.**

Властивості відношенньь еквівалентності відносно теоретико-множинних операцій дає теорема 2.

**Теорема 2.** *Якщо $R, R_1$ – відношення еквівалентності, що задані на множині $A$, то*

а) $R^{-1}$ *– відношення еквівалентності на* $A$;

б) $R * R_1$ *– відношення еквівалентності на* $A \Leftrightarrow R * R_1 = R_1 * R$, *тобто коли відношення* $R$ *і* $R_1$ *можна міняти місцями*;

в) $R \cap R_1$ *– відношення еквівалентності на* $A$;

г) $R'$ *не є відношенням еквівалентності на* $A$.

Об'єднання відношень еквівалентності загалом не завжди буде відношенням еквівалентності, як показує теорема 3.

**Теорема 3.** *Об'єднання $R \cup R_1$ відношень еквівалентності $R$ і $R_1$ є еквівалентністю тоді і тільки тоді, коли перетин довільного класу еквівалентності по $R$ із довільним класом еквівалентності по $R_1$ або збігається з одним із них, або пустий. Якщо $R \cup R_1$ – еквівалентність, то $R \cup R_1 = R * R_1$.*

### 2.2.3. Відношення часткового порядку

Одним із фундаментальних бінарних відношень є відношення часткового порядку.

**Означення 5.** *Бінарне відношення $O$, визначене на множині $A$, називається* **частковим порядком** *на $A$, якщо воно рефлексивне, транзитивне і антисиметричне, тобто, якщо для довільних елементів $a, b, c$ із $A$ виконуються властивості:*

а) $aOa$ $\quad (i_A \subseteq O)$ (*рефлексивність*);

б) $aOb$ і $bOc \Rightarrow aOc$ $(O^2 \subseteq O)$ (*транзитивність*);

в) $aOb$ і $bOa \Rightarrow a = b$ $(O \cap O^{-1} \subseteq i_A)$ (*антисиметричність*).

Частковий порядок на множині $A$, як правило, позначають символом $\leq$, а саму частково впорядковану множину $A$ називають скорочено **чум** і позначають $(A, \leq)$. Якщо $a \leq b$ для деяких $a, b \in \in A$, то говорять, що елементи $a$ і $b$ є такими, що порівнюються між собою і $a$ менше або рівне $b$, або що $a$ включається в $b$ або рівне $b$.

**Означення 6.** *Транзитивне й антирефлексивне відношення називається відношенням строгого порядку. Це відношення позначають $<$.*

*Транзитивне і рефлексивне відношення називається відношенням квазіпорядку. Це відношення позначають $\preceq$.*

**Найпростіші властивості чум**. Одна з важливих властивостей часткового порядку випливає з теореми.

**Теорема 4 (принцип двоїстості).** *Відношення, обернене до відношення часткового порядку, теж буде відношенням часткового порядку.*

**Означення 7.** *Відношення часткового порядку* $\leq^{-1}$ *називається* **двоїстим до** *відношення часткового порядку* $\leq$.

Відношення $\leq^{-1}$ позначається $\geq$ і $a \leq^{-1} b$ означає $a \geq b$. Якщо $a \leq b$ або $b \leq a$, то $a, b$ називають елементами, що **порівнюються відносно порядку** $\leq$.

З принципу двоїстості випливає, що коли в якому-небудь твердженні про чум замінити частковий порядок на двоїстий до нього порядок, то одержане твердження теж буде правильне.

**Означення 8.** *Елемент* $x$ *із множини* $(A, \leq)$ *називається* **мінімальним (максимальним) елементом** $A$*, якщо для довільного елемента* $a$ *з* $A$*, що порівнюється з* $x$*, справедлива нерівність* $x \leq a$ $(x \geq a)$.

*Елемент* $x$ *із* $A$ *називається* **найбільшим (найменшим)**, *якщо* $(\forall a \in A)$ $x \geq a$ $(x \leq a)$.

Із цього означення очевидним чином випливає, що коли $a$ і $a'$ різні мінімальні (максимальні) елементи деякої чум $(A, \leq)$, то вони не порівнюються між собою відносно порядку $\leq$.

**Теорема 5.** *У довільній чум* $(A, \leq)$ *існує не більше одного найменшого* (*а на підставі принципу двоїстості і найбільшого*) *елемента.*

Якщо довільні два елементи із множини $A$ порівнюються відносно $\leq$, то таке відношення називають **лінійним порядком** на $A$, а множину $A$ – **лінійно упорядкованою, або ланцюгом**.

Зрозуміло, що коли лінійно упорядкована множина $A$ має найбільший (найменший) елемент, то цей елемент буде її єдиним максимальним (мінімальним) елементом.

Важливим прикладом відношення лінійного порядку є лексикографічний порядок. Нехай маємо скінченний алфавіт $X = \{x_1, x_2 \ldots, x_n\}$, літери якого лінійно упорядковані: $x_i < x_j \Leftrightarrow i < j$. **Лексикографічним порядком** на множині $F(X)$ називається відношення $\leq$, для якого $p = x_{11}x_{12}\cdots x_{1k} \leq x_{21}x_{22}\cdots x_{2r} = q$ тоді і тільки тоді, коли виконується одна з двох умов:

а) слово $p$ є початком слова $q$;

б) існує таке натуральне число $j$, що $x_{1j} < x_{2j}$, і для всіх $i < j$ справедливі рівності $x_{1i} = x_{2i}$.

Згідно з лексикографічним порядком упорядковано слова в словниках, якщо літери алфавіту лінійно упорядковані.

### 2.2.4. Функції, операції, предикати

Відношення $F$, задане на множинах $A_1, A_2, \ldots, A_n, B$, називається **функціональним,** якщо для довільного елемента $(a_1, a_2, \ldots, a_n) \in A_1 \times A_2 \times \ldots \times A_n$ існує не більше одного елемента $b$ із $B$ такого, що $(a_1, a_2, \ldots, a_n, b) \in F$. Якщо такий елемент $b$ із $B$ існує для деякого $(a_1, a_2, \ldots, a_n)$, то він позначається через $F(a_1, a_2, \ldots, a_n)$ і записується так: $b = F(a_1, a_2, \ldots, a_n)$.

Нехай $F^{-1}(b) = \{(a_1, \ldots, a_n) \in A_1 \times \ldots \times A_n | F(a_1, \ldots, a_n) = b\}$ і

$$\mathrm{Dom}(F) = \bigcup_{b \in B} F^{-1}(b).$$

Очевидно, що для довільного функціонального відношення $F$, заданого на $A_1, A_2, \ldots, A_n, B$, справедливе включення

$$\mathrm{Dom}(F) \subseteq A_1 \times A_2 \times \ldots \times A_n.$$

**Означення 9.** *Відношення $F$ називається* **повністю визначеним**, *якщо* $\mathrm{Dom}(F) = A_1 \times A_2 \times \ldots \times A_n$, *і* **частково визначеним** *або просто* **частковим**, *якщо* $\mathrm{Dom}(F) \subset A_1 \times A_2 \times \ldots \times A_n$.

*Відношення $F$, задане на множинах $A_1, A_2, \ldots, A_n, B$, називається* **відображенням**, *або* **функцією** *із $A_1 \times A_2 \times \ldots \times A_n$ у $B$ ($F : A_1 \times A_2 \times \ldots \times A_n \to B$), якщо $F$ функціональне і повністю визначене. Відношення $F$ називається* **частковим відображенням**, *або* **частковою функцією**, *якщо $F$ функціональне і часткове. Число $n$ називається* **арністю** *відображення $F$.*

Якщо $F : A_1 \times A_2 \times \ldots \times A_n \to B$ й існує $b$ із $B$ такий, що $F(a_1, a_2, \ldots, a_n) = b$, то елемент $b$ називають *образом* елемента $(a_1, a_2, \ldots, a_n)$ при відображенні $F$, а елемент $(a_1, a_2, \ldots, a_n)$ – *прообразом* елемента $b$. Множину

$$F^{-1}(b) = \{(a_1, a_2, \ldots, a_n) | F(a_1, a_2, \ldots, a_n) = b\},$$

введену раніше, називають *повним прообразом* елемента $b$ у множині $A_1 \times A_2 \times \ldots \times A_n$.

Нехай $F : A_1 \times A_2 \times \ldots \times A_n \to B$ і $F^{-1}(b)$ – повний прообраз елемента $b$ в $A_1 \times A_2 \times \ldots \times A_n$ при відображенні $F$. Уведена вище множина $\mathrm{Dom}(F) = \bigcup\limits_{b \in B} F^{-1}(b)$ називається **областю визначення** відображення $F$, а $\mathrm{Im}(F) = \{b | F^{-1}(b) \neq \emptyset\}$ – **областю значень.** Якщо $F : A_1 \times A_2 \times \ldots \times A_n \to B$ і $F_1 : A_1 \times A_2 \times \ldots \times A_n \to B$, то $F = F_1 \Leftrightarrow \mathrm{Dom}(F) = \mathrm{Dom}(F_1)$, $\mathrm{Im}(F) = \mathrm{Im}(F_1)$ і $F(a_1, a_2, \ldots, a_n) = F_1(a_1, a_2, \ldots, a_n)$ для довільної $n$-ки $(a_1, a_2, \ldots, a_n) \in A_1 \times A_2 \times \ldots \times A_n$.

**Означення 10.** *Відображення* $\varepsilon : A_1 \times A_2 \times \ldots \times A_n \to A_1 \times A_2 \times \ldots \times A_n$, *яке ставить у відповідність кожному елементові множини* $A_1 \times A_2 \times \ldots \times A_n$ *той самий елемент, тобто* $\forall (a_1, a_2, \ldots, a_n) \in A_1 \times A_2 \times \ldots \times A_n$ *маємо* $\varepsilon(a_1, a_2, \ldots, a_n) = (a_1, a_2, \ldots, a_n)$, *називається* **тотожним відображенням**.

*Відображення* $F : A_1 \times A_2 \times \ldots \times A_n \to B$ *називають* **вкладенням**, *або* **ін'єкцією**, *тоді і тільки тоді, коли із* $(a_1, a_2, \ldots, a_n) \neq (a'_1, a'_2, \ldots, a'_n)$ *випливає* $F(a_1, a_2, \ldots, a_n) \neq F(a'_1, a'_2, \ldots, a'_n)$.

*Відображення* $F : A_1 \times A_2 \times \ldots \times A_n \to B$ *називають* **відображенням "на"**, *або* **сюр'єкцією**, *тоді і тільки тоді, коли* $(\forall b \in B)(F^{-1}(b) \neq \emptyset)$.

*Відображення* $F$ *множини* $A_1 \times A_2 \times \ldots \times A_n$ *на множину* $B$ *називається* **взаємно однозначним відображенням**, *або* **взаємно однозначною відповідністю**, *або* **бієкцією**, *тоді і тільки тоді, коли воно є ін'єкцією і сюр'єкцією одночасно.*

*Якщо* $f : A \to B$ – *відображення і* $A_1 \subseteq A$, *то відображення* $f_{A_1} : A_1 \to B$ *називається* **звуженням**, *або* **обмеженням**, *відображення* $f$ *із множини* $A$ *на множину* $A_1$, *коли* $f_{A_1}(a) = f(a)$ *для всіх* $a \in A_1$.

*Якщо* $A_1 \subseteq A$ *і* $f : A_1 \to B$ – *відображення, то відображення* $g : A \to B$ *називається* **розширенням** *відображення* $f$ *із множини* $A_1$ *на множину* $A$, *коли* $g(a) = f(a)$ *для всіх* $a \in A_1$.

Якщо $f : A \to B$ – відображення, а множини $A$ і $B$ – скінченні і мають відповідно $m$ та $n$ елементів, то необхідною умовою ін'єктивності $f$ є умова $m \leq n$, сюр'єктивності $f$ – умова $n \leq m$, бієктивності $f$ – умова $m = n$.

**Приклад 2.2.1.** Відображення відіграють основну роль у процесі передачі та обробки інформації. Це пов'язано, перш за все, з поняттям кодування.

Нехай $X = \{x_1, x_2, \ldots, x_r\}$ і $Y = \{y_1, y_2, \ldots, y_m\}$ – два скінченні алфавіти.

**Алфавітним відображенням** називається відображення

$$f : X \to F(Y),$$

таке, що $f(x_i) = q_i$, де $i = 1, 2, \ldots, r$, а $F(Y)$ – множина всіх слів скінченної довжини в алфавіті $Y$. Зазначимо, що відображення $f$ розширюється на множину слів $F(X)$ таким чином: $f(x_{i_1} x_{i_2} \ldots x_{i_s}) = f(x_{i_1}) f(x_{i_2}) \ldots f(x_{i_s}) = q_{i_1} q_{i_2} \ldots q_{i_s}$. Алфавіт $Y$ у такому випадку називається кодуючим, а множина слів $\{q_1, q_2, \ldots, q_r\}$ – **множиною кодових слів**, або просто **кодом**.

Кодування $f$ називається **взаємно однозначним**, якщо для довільних $p_1, p_2 \in F(X)$ із $p_1 \neq p_2$ випливає $f(p_1) \neq f(p_2)$.

Код називається префіксним (суфіксним), якщо жодне з кодових слів не є початком (кінцем) іншого слова. Неважко показати, що довільний префіксний код є взаємно однозначним кодом.

Найчастіше в застосуваннях кодуючим алфавітом виступає двійковий алфавіт $Y = \{0, 1\}$. ♠

Вище була введена операція добутку відношень. Оскільки відображення – це відношення спеціального вигляду, то з'ясуємо, що є добутком таких відображень.

Нехай $F_1 : A \to B$, а $F : B \to C$ – деякі відображення. З означення операції добутку відношень маємо: $(a, c) \in F_1 * F \Leftrightarrow$ існує елемент $b \in B$ такий, що $(a, b) \in F_1$ і $(b, c) \in F$, тобто $F_1(a) = b$ і $F(b) = c$, або $F(F_1(a)) = c$ згідно з прийнятими вище позначеннями. Таким чином, добуток відображень є добре відомою операцією суперпозиції функцій.

Для ін'єкцій, сюр'єкцій і бієкцій справедливе твердження 1.

**Твердження 1.** а) *добуток ін'єктивних відображень є ін'єкцією*;

б) *добуток сюр'єктивних відображень є сюр'єкцією*;

в) *добуток бієктивних відображень є бієкцією.*

*Доведення.* Не обмежуючи загальності, будемо вважати, що відображення $f$ і $f_1$ мають вигляд $f : A \to B$ і $f_1 : B \to C$.

а) Нехай $a \neq a'$, тоді на підставі ін'єктивності $f$ отримуємо $f(a) \neq f(a')$. Але тоді на підставі ін'єктивності $f_1$ дістаємо $f_1(f(a)) \neq f_1(f(a'))$. А це означає ін'єктивність відображення $f * f_1$.

б) На підставі сюр'єктивності відображень $f$ і $f_1$ для довільного елемента $c \in C$ існує елемент $b \in B$ такий, що $f_1(b) = c$. Далі для довільного елемента $b \in B$ існує елемент $a \in A$ такий, що $f(a) = b$. Але тоді $f_1(f(a)) = c$, що означає сюр'єктивність відображення $f * f_1$;

в) доведення випливає з попередніх пунктів *а*), *б*). ■

Нехай $\varepsilon_A$ і $\varepsilon_B$ означають тотожні відображення множин $A$ і $B$ відповідно. Очевидно, що $\varepsilon_A * f = f$ і $f * \varepsilon_B = f$ для довільного відображення $f : A \to B$.

**Означення 11.** *Відображення $g : B \to A$ називається* **оберненим до відображення** *$f : A \to B$, якщо $f * g = \varepsilon_A$ і $g * f = \varepsilon_B$. Обернене відображення до відображення $f$ позначають $f^{-1}$. Це означає, що $f(a) = b \Leftrightarrow f^{-1}(b) = a$.*

**Приклад 2.2.2.** 1) Нехай дано множини $X = \{a, b, c, d\}$, $Y = \{2, 3, 4, 5, 6\}$ і функція $f$, яка визначена таким чином: $f(a) = 2, f(b) = 4, f(c) = 6, f(d) = 3$. Функцію $f$ можна подати у вигляді такої діаграми (рис. 2.2.1), з якої видно, що $f$ ін'єкція:

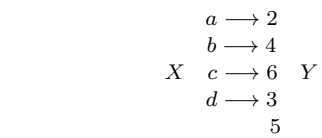



Рис. 2.2.1. Діаграма функції $f$

Прообразом елемента 4 буде одноелементна множина $b$, прообразом елемента 5 буде пуста множина $\emptyset$, а множина значень даної функції складається з елементів $\{2, 4, 6, 3\}$.

2) Нехай $X = \{1, 2, 3, \ldots, 10\}$, $Y = X$ і функція $f$ є такою:

$$f(1) = 1,\ f(2) = 4,\ f(3) = 9,\ f(4) = 5, f(5) = 3,$$
$$f(6) = 3,\ f(7) = 5,\ f(8) = 9,\ f(9) = 4, f(10) = 1.$$

Для цієї функції маємо таку множину значень: $Y = \{1, 3, 4, 5, 9\}$. Звідси випливає, що дана функція є повністю визначеною, є відображенням у множину $Y$, а не на множину $Y$ (елементи 2, 6, 7, 8, 10 мають пусті прообрази в $X$), а отже, не є взаємно однозначним відображенням.

3) Нехай $X = \{a, b, c, d, e\}$, $Y = \{1, 2, 3, 4, 5\}$, і розглянемо функцію $f$, визначену за допомогою такої діаграми:

$$X \quad \begin{array}{l} a \longrightarrow 5 \\ b \longrightarrow 2 \\ c \longrightarrow 1 \\ d \longrightarrow 4 \\ e \longrightarrow 3 \end{array} \quad Y \qquad\qquad Y \quad \begin{array}{l} 5 \longrightarrow a \\ 2 \longrightarrow b \\ 1 \longrightarrow c \\ 4 \longrightarrow d \\ 3 \longrightarrow e \end{array} \quad X.$$

Із цієї діаграми випливає, що функція $f : X \to Y$ ін'єкція і, навіть, бієкція, і для цієї функції існує обернена до неї функція $g = f^{-1} : Y \to X$.

4) У випадку нескінченних областей означення і значень функцій установлення існування обернених функцій вимагає певних зусиль. Розглянемо приклади.

а) Довести, що функція $f : \mathcal{D} \to \mathcal{D}$, задана виразом $f(x) = 4x + 3$, є бієкцією.

б) Довести, що функція $f : \mathcal{D} \setminus \{1\} \to \mathcal{D} \setminus \{1\}$, задана виразом $f(x) = \frac{x}{x-1}$, є бієкцією. Знайти для неї обернену функцію.

*Доведення.* а) Припустимо, що $f(a) = f(b)$ для деяких $a, b \in \mathcal{D}$, тобто $4a + 3 = 4b + 3$. Звідси отримуємо $4a = 4b$ і, отже, $a = b$. Таким чином, функція $f$ є ін'єкцією.

Нехай $b \in \mathcal{D}$ – довільний елемент. Покажемо, що існує $a \in \mathcal{D}$ такий, що $f(a) = b$. Досить прості перетворення дають $a = \frac{1}{4}(b - 3)$.

Отже, дана функція є ін'єкцією і сюр'єкцією, і тому є бієкцією.

б) Припустимо, що $f(a) = f(b)$ для деяких $a, b \in \mathcal{D} \setminus \{1\}$, тобто $\frac{a}{a-1} = \frac{b}{b-1}$. Звідси отримуємо $ab - a = ab - b$ і, отже, $a = b$. Таким чином, функція $f$ є ін'єкцією.

Нехай $b \in \mathcal{D} \setminus \{1\}$ – довільний елемент. Покажемо, що існує $a \in \mathcal{D} \setminus \{1\}$ такий, що $f(a) = b$. Досить прості перетворення дають $a = \frac{b}{b-1}$. На підставі довільності елемента $b$ отримуємо, що $f$ – сюр'єкція.

Отже, дана функція є ін'єкцією і сюр'єкцією, а тому є бієкцією.

Обернена функція $f^{-1}$ знаходиться з умови $f^{-1}(b) = a \Leftrightarrow f(a) = b$. Але зі знайденого вище маємо $a = \frac{b}{b-1}$.

Таким чином, $f^{-1} : \mathcal{D} \setminus \{1\} \to \mathcal{D} \setminus \{1\}$ має вигляд $f^{-1}(x) = \frac{x}{x-1}$, тобто $f$ обернена до самої себе.

5) Нехай для $A = \{1, 2, 3, 4, 5\}$ відображення $f : A \to \mathcal{D}$, де $\mathcal{D}$ – множина дійсних чисел і $f = \{(1, 10), (2, 13), (3, 16), (4, 19), (5, 22)\}$, $g : \mathcal{Q} \to \mathcal{D}$, де $g(m) = 3m + 7$ для всіх $m \in \mathcal{Q}$ і $h : \mathcal{D} \to \mathcal{D}$ – відображення таке, що $h(r) = 3r + 7$ для всіх $r \in \mathcal{D}$. Тоді

а) $g$ – розширення $f$ з $A$ на $\mathcal{Q}$; б) $f$ – звуження $g$ із $\mathcal{Q}$ на $A$;

в) $h$ – розширення $f$ з $A$ на $\mathcal{D}$; г) $f$ – звуження $h$ із $\mathcal{D}$ на $A$;

д) $h$ – розширення $g$ з $\mathcal{Q}$ на $\mathcal{D}$; е) $g$ – звуження $h$ із $\mathcal{D}$ на $\mathcal{Q}$. ♠

Розглянемо питання про умови, за яких існує обернене відображення, та властивості обернених відображень.

**Теорема 6.** а) *відображення $f : A \to B$ має обернене відображення $f^{-1}$ тоді і тільки тоді, коли $f$ – бієкція*;

б) *якщо відображення $f$ має обернене відображення, то це відображення єдине;*

в) *якщо відображення* $f : A \to B$ *– бієкція, то обернене відображення* $f^{-1}$ *теж бієкція і* $(f^{-1})^{-1} = f$;

г) *якщо відображення* $f : A \to B$ *і* $g : B \to C$ *– бієкції, то їхній добуток* $f * g$ *теж бієкція і* $(f * g)^{-1} = g^{-1} * f^{-1}$;

д) *якщо множина* $A$ *скінченна і* $f : A \to A$ *є ін'єкцією або сюр'єкцією, то* $f$ *буде бієкцією.*

*Доведення.* а) Доведемо, що коли $f : A \to B$ і $g : B \to A$ – довільні відображення, які задовольняють умову $f * g = \varepsilon_A$, то $f$ – ін'єкція, а $g$ – сюр'єкція. Дійсно, якщо $a, a' \in A$ і $f(a) = f(a')$, то

$$a = \varepsilon_A(a) = f * g(a) = g(f(a)) = g(f(a')) = f * g(a') = \varepsilon_A(a') = a'.$$

Отже, відображення $f$ – ін'єкція. Якщо $a \in A$ – довільний елемент, то $a = \varepsilon_A(a) = f * g(a) = g(f(a))$, а це доводить сюр'єктивність відображення $g$.

Припустимо, що відображення $f$ має обернене $f^{-1}$. Тоді із $f * *f^{-1} = \varepsilon_A$ і $f^{-1} * f = \varepsilon_B$ випливає, що $f$ – сюр'єкція і ін'єкція, тобто $f$ – бієкція.

Навпаки, припустимо, що $f$ – бієкція. Тоді для довільного елемента $b \in B$ знайдеться єдиний елемент $a \in A$, який є прообразом елемента $b$, тобто $f(a) = b$. Покладаючи $g(b) = a$, визначаємо відображення $g : B \to A$, яке задовольняє умові $f * g = \varepsilon_A$ і $g * f = \varepsilon_B$, Отже, $g = f^{-1}$.

б) Припустимо, що існує два відображення $g$ і $g'$, які обернені до відображення $f$, тобто $f * g = \varepsilon_A$ і $g * f = \varepsilon_B$ та $f * g' = \varepsilon_A$ і $g' * f = \varepsilon_B$. Тоді отримуємо

$$g' = \varepsilon_B * g' = (g * f) * g' = g * (f * g') = g * \varepsilon_A = g.$$

в) Якщо $f$ – бієкція, то за попереднім пунктом *а*) існує єдине $f^{-1}$. На тій же підставі відображення $f^{-1}$ теж бієкція. Із симетричності умов $f * f^{-1} = \varepsilon_A$ і $f^{-1} * f = \varepsilon_B$ випливає $(f^{-1})^{-1} = f$.

г) Те, що добуток $f * g$ є бієкцією, випливає з попереднього твердження. А з умови цього пункту і пункту *а*) теореми випливає існування відображень $f^{-1} : B \to A$ і $g^{-1} : C \to B$, а також їхні суперпозиції $g^{-1} * f^{-1} : C \to A$. Звідси отримуємо:

$(f*g)*(g^{-1}*f^{-1}) = ((f*g)*g^{-1})*f^{-1} = (f*(g*g^{-1}))*f^{-1} = f*f^{-1} = \varepsilon_A,$
$(g^{-1}*f^{-1})*(f*g) = ((g^{-1}*f^{-1})*f)*g = (g^{-1}*(f*f^{-1}))*g = g^{-1}*g = \varepsilon_C.$

А це означає, що відображення $g^{-1}*f^{-1}$ обернене до відображення $f*g$.

д) Нехай $f$ – ін'єкція, необхідно показати, що $f$ – сюр'єкція. Покладемо $\forall a \in A$ $f^k(a) = \underbrace{f(f\ldots f}_{k \text{ разів}}(a)\ldots) = f(f^{k-1}(a))$, $k = 0, 1,$ $2, \ldots$. Оскільки множина $A$ скінченна, то в цій послідовності елементи повинні повторитися. Нехай, наприклад, $f^m(a) = f^n(a)$ і $m > n$. Якщо $n > 0$, то з рівності $f(f^{m-1}(a)) = f(f^{n-1}(a))$ випливає $f^{m-1}(a) = f^{n-1}(a)$. Повторивши це перетворення $n$ разів, скоротимо $f$ і приходимо до рівності $a = f^{m-n}(a) = f(f^{m-n-1}(a))$. А звідси отримуємо елемент $a' = f^{m-n-1}(a)$, для якого $f(a') = = f^{m-n}(a) = a$.

Доведення у випадку сюр'єкції $f$ проводиться аналогічно. ■

Розглянемо приклад побудови, яка зв'язує відображення множини $A$ на множину $B$ та індуковане ним відношення еквівалентності $R$ на $A$ з фактор-множиною $A/R$.

Дійсно, якщо $F_1 : A \to B$ – сюр'єкція, то їй відповідає цілком визначене відношення еквівалентності $R_{F_1}$ на $A$: якщо $a, b \in A$, то $aRb$ тоді і тільки тоді, коли $F_1(a) = F_1(b)$. Уводячи відображення $F : A \to A/R$, яке ставить у відповідність елементу $a$ із $A$ клас розбиття, якому належить $a$ (відображення $F$ називається **натуральним відображенням** $A$ на $A/R$), і ставлячи у відповідність кожному елементу $x$ із $B$ його повний прообраз в $A$, дістаємо відображення $F_2 : B \to A/R$, основну властивість яких випливає з теореми 7.

**Теорема 7.** *Якщо $F_1$ є відображенням множини $A$ на множину $B$ і $R$ – відношення еквівалентності на $A$, що відповідає $F_1$, то відображення $F_2 : B \to A/R$ є взаємно однозначним відображенням, причому $F_1 * F_2 = F$, де $F$ – натуральне відображення $A$ на $A/R$.*

*Доведення.* Визначимо $F_2 : B \to A/R$ так: $F_2(b) = K(b) = \{a \in$ $\in A | F_1(a) = b\}$ (рис. 2.2.2). Зрозуміло, що коли $b \neq b'$, то $K(b) \neq$

$\neq K(b')$. Отже, відображення $F_2$ взаємно однозначне. Далі, нехай $F_1(a) = b$, а $F_2(b) = K(b)$, тоді $a \in K(b)$. Таким чином, добуток $F_1 * F_2$ збігається з натуральним відображенням $F$. ■

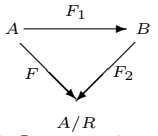



Рис. 2.2.2. Ілюстрація теореми 7

**Приклад 2.2.3.** Дано функції $f : \mathcal{D} \to \mathcal{D}$ і $g : \mathcal{D} \to \mathcal{D}$, де $f(x) = x^2$, $g(x) = 2x + 1$. Знайти $f * g$, $g * f$, $f * f$ і $g * g$.

*Розв'язання.* Усі функції визначені на множині дійсних чисел $\mathcal{D}$. Отже,
$(f * g)(x) = g(f(x)) = g(x^2) = 2x^2 + 1$;
$(g * f)(x) = f(g(x)) = f(2x + 1) = (2x + 1)^2 = 4x^2 + 4x + 1$;
$(f * f)(x) = f(f(x)) = f(x^2) = x^4$;
$(g * g)(x) = g(g(x)) = g(2x + 1) = 2(2x + 1) + 1 = 4x + 3$.

**Приклади функцій**, які часто зустрічаються в застосуваннях.

а) *Селекторні функції.* Функція $I_m^n(x_1, \ldots, x_n) = x_m$ називається **селекторною функцією**, де $1 \leq m \leq n$.

б) *Функція "низ"* – $\lfloor x \rfloor$ – дає найбільше ціле число, яке не перевищує $x$, де $x \in \mathcal{D}$.

в) *Функція "верх"* – $\lceil x \rceil$ – дає найменше ціле число, яке не менше $x$, де $x \in \mathcal{D}$.

г) *Функція "остача".* Для довільного цілого числа $a$ і натурального $n$ функція $rest(a, n)$ є остачею від ділення $a$ на $n$: $rest(a, n) = a - \lfloor \frac{a}{n} \rfloor \cdot n$. Ця функція інколи буде позначатися як $\mathrm{mod}(a, n)$.

д) *Функція логарифм.* Існує декілька функцій цього типу, залежно від основи:
$\lg n = \lg_{10} n$ – десятковий; $\log n = \log_2 n$ – двійковий;
$\ln n = \log_e n$ – натуральний, $\log \log n = \log(\log n)$, $\log^k n = (\log n)^k$.

є) $\chi(a)$ – характеристична функція деякої множини $A$, значення якої дорівнює одиниці, якщо $a \in A$, і дорівнює нулю, якщо $a \notin A$.

ж) *Факторіал* $n!$ натурального числа $n$ покладається для $n = 0$ рівним 1, а для $n > 1$ рівним $n \cdot (n - 1)!$.

и) *Монотонна функція.* Функція $f(n)$ називається **монотонно неспадною (незростаючою)**, якщо із нерівності $m \leq n$ випливає нерівність $f(m) \leq f(n)$ $(f(n) \leq \leq f(m))$. Функція $f(n)$ називається **монотонно спадною (зростаючою)**, якщо із $m < n$ випливає $f(m) > f(n)$ $(f(m) < f(n))$. ♠

## 2.2.5. Асимптотичне порівняння функцій

Для аналізу якісних властивостей алгоритмів важливим є встановлення класу складності алгоритму (строгі означення будуть наведені далі). Складність алгоритму визначається функцією, аргу-

ментами якої виступають розміри вхідних даних алгоритму, а її значення на цих даних є його характеристикою. Ця функція називається функцією складності алгоритму, а такий аналіз зводиться до асимптотичного порівняння швидкостей зростання функцій.

Існує принаймні чотири категорії алгоритмів за складністю: алгоритми, складність яких зростає

– швидше, ніж дана функція $f(n)$;

– не швидше, ніж дана функція $f(n)$;

– з тією самою швидкістю, що і $f(n)$;

– повільніше, ніж перша функція і швидше, ніж друга функція.

Розглянемо категорії зростання детальніше. Асимптотична швидкість зростання функції є сумою складностей окремих частин алгоритму і визначається старшим, домінуючим членом цієї суми. За таких умов молодшими членами суми нехтують, оскільки вони зростають повільніше ніж старший. Нехтуючи молодшими членами, отримуємо те, що називається **порядком зростання функції**, який відображає складність алгоритму. У такому випадку аргументи функцій складності, як і їхні значення належать множині натуральних чисел. Для ілюстрації розглянемо приклад. Нехай з аналізу алгоритму, встановлено, що він виконує $n^3+30n$ порівнювань. Тоді, згідно зі сказаним вище, говоримо, що складність алгоритму росте, як $n^3$. Підставою для такого висновку є те, що вже за розміру вхідних даних $n = 100$ різниця між $n^3$ і $n^3 + 30n$ складає всього лише 0,3 %.

Нехай дано дві функції $f(n)$ і $g(n)$, де значення аргументу $n$ і самих функцій належать множині натуральних чисел $\mathcal{N}$.

**Означення 12.** а) (*О-велике*) *Вираз* $f(n) = O(g(n))$ *означає, що існують константа* $c > 0$ *і* $n_0 \in \mathcal{N}^+$ *такі, що* $0 \leq f(n) \leq cg(n)$, *для всіх* $n \geq n_0$;

б) (нижня асимптотична границя) *Вираз* $f(n) = \Omega(g(n))$ *означає, що існують константа* $c > 0$ *і* $n_0 \in \mathcal{N}^+$ *такі, що* $0 \leq cg(n) \leq \leq f(n)$, *для всіх* $n \geq n_0$;

в) (пасмова асимптотична границя) *Вираз* $f(n) = \Theta(g(n))$ *означає, що існують константи* $c_1 > 0$ *і* $c_2 > 0$, *де* $c_1 \leq c_2$, *та число* $n_0 \in \mathcal{N}^+$ *такі, що* $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$, *для всіх* $n \geq n_0$;

г) (*о-мале*) *Вираз* $f(n) = o(g(n))$ *означає, що для довільної константи* $c > 0$ *існує число* $n_0 \in \mathcal{N}^+$ *таке, що* $0 \leq f(n) < cg(n)$, *для всіх* $n \geq n_0$.

Якщо $f(n) = \Omega(g(n))$, то це означає, що функція $f(n)$ не може набувати значень менших, ніж $c \cdot g(n)$, тобто

$$f(n) \in \Omega(g(n)), \text{ якщо } \lim_{n\to\infty} \frac{f(n)}{g(n)} \geq c,$$

де $n \geq n_0, g(n) \neq 0$.

Якщо $f(n) = o(g(n))$, то це означає, що функція $g(n)$ зростає набагато швидше, ніж функція $f(n)$, тобто

$$f(n) \in o(g(n)), \text{ якщо } \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0,$$

де $n \geq n_0, g(n) \neq 0$.

**Приклад 2.2.4.** а) Поліном $f(n) = 4n^5 + 3n^2 + 2n + 3$ є $O(n^5)$, оскільки

$$\lim_{n\to\infty} \frac{4n^5+3n^2+2n+3}{n^5} = 4.$$

б) Поліном $f(n) = 4n^5 + 3n^2 + 2n + 3$ є $o(n^6)$ і $o(2^n)$, оскільки

$$\lim_{n\to\infty} \frac{4n^5+3n^2+2n+3}{n^6} = 0 \text{ і } \lim_{n\to\infty} \frac{4n^5+3n^2+2n+3}{2^n} = 0.$$

в) Функція $f(n) = 2^n$ є $o(n!)$, оскільки $\lim_{n\to\infty} \frac{2^n}{n!} = 0$. ♠

Назвемо функцію $f(n)$ *асимптотично додатною*, якщо вона набуває додатних значень при достатньо великих значеннях $n$.

Деякі основні властивості введеної нотації випливають із теореми 8.

**Теорема 8.** *Для довільних асимптотично додатних функцій f(n), g(n), h(n) і l(n) справедливі такі властивості:*

а) $f(n) = O(g(n)) \Leftrightarrow g(n) = \Omega(f(n))$;

б) $f(n) = \Theta(g(n)) \Leftrightarrow$ *f(n) = O(g(n)) і* $f(n) = \Omega(g(n))$;

в) *якщо* $f(n) = O(h(n)), g(n) = O(h(n))$, *то* $(f + g)(n) = O(h(n))$;

г) *якщо* $f(n) = O(h(n)), g(n) = O(l(n))$, *то* $(f \cdot g)(n) = O(h(n) \cdot l(n))$;

д) (*рефлексивність*)

$f(n) = O(f(n)), f(n) = \Theta(f(n)), f(n) = \Omega(f(n))$;

є) (*симетричність*) $f(n) = \Omega(g(n)) \Leftrightarrow g(n) = \Omega(f(n))$;

ж) (*транзитивність*) *якщо*

$f(n) = O(g(n))$ *і* $g(n) = O(h(n))$, *то* $f(n) = O(h(n))$,
$f(n) = \Theta(g(n))$ *і* $g(n) = \Theta(h(n))$, *то* $f(n) = \Theta(h(n))$,
$f(n) = \Omega(g(n))$ *і* $g(n) = \Omega(h(n))$, *то* $f(n) = \Omega(h(n))$,
$f(n) = o(g(n))$ *і* $g(n) = o(h(n))$, *то* $f(n) = o(h(n))$.

Зауважимо, що коли йдеться про ефективність алгоритмів у програмуванні, то найбільший інтерес має клас *O-велике*. А коли говоримо про складність алгоритмів у криптографічних застосуваннях, то тут важливу роль відіграють алгоритми, які мають великі оцінки часової складності.

## 2.3. Елементи теорії складності алгоритмів

У зв'язку з тим, що строгого означення поняття "алгоритм" не існує, то користуються інтуїтивним означенням алгоритму.

**Означення 13.** *Алгоритмом називається цілком визначена процедура, яка після отримання вхідних даних генерує вихідні дані у скінченному проміжку часу.*

Зрозуміло, що це означення не є математично строгим, але його достатньо для нашої мети. Визначимо, що є вхідними і вихідними даними. Для цього скористаємося алфавітом $X = \{0, 1\}$.

**Означення 14.** *Розміром вхідних* (*вихідних*) *даних алгоритму називається довжина слова p в алфавіті X, яке зображує ці дані.*

Пояснимо це поняття прикладами.

**Приклад 2.3.1**. 1) Натуральне число $n \in \mathcal{N}$ зображується двійковим словом довжини $(1 + \lceil \log\ n \rceil)$, де $\lceil \log\ n \rceil$ означає найменше ціле число, яке не менше $\log\ n$. Далі будемо користуватися записом $\log\ n$, маючи на увазі $\lceil \log\ n \rceil$.

2) Поліном $f$ степеня $k$, коефіцієнти якого є невід'ємними числами не більшими ніж число $n$, зображується двійковим словом довжини $(k + 1) \cdot \log\ n$.

3) Матриця $A$ з $r$ рядками і $s$ стовпцями та невід'ємними цілими коефіцієнтами не більшими ніж число $n$ зображується двійковим словом довжини $rs \cdot \log\ n$. ♠

### 2.3.1. Машинні моделі

Класи складності найбільш природно описуються за допомогою машинних моделей обчислень. Стандартним варіантом таких моделей є машини Тьюрінга (МТ) та їхні варіації: детерміновані, недетерміновані, багатострічкові та ін.

**Детермінована машина Тьюрінга**. Неформально детермінована машина Тьюрінга (ДМТ) є пристроєм, що має чутливу **головку** $G$, яка здатна пересуватися вздовж нескінченної в одну (праву) сторону стрічки, розбитої на **клітинки** (рис. 2.3.1).

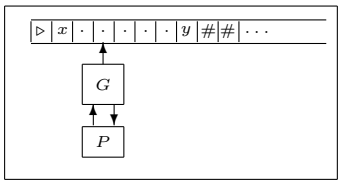



Рис. 2.3.1. Схема ДМТ

Правила руху головки вздовж стрічки визначаються її **програмою** $P$, яка в теорії ДМТ називається **функцією переходів**. Головка здатна **читати символ зі стрічки, писати символ на стрічку** і **пересуватися** ліворуч чи праворуч на одну клітинку або залишатися в тій самій позиції відповідно до програми цієї ДМТ. Формальне означення ДМТ є таким.

**Означення 15.** *Детермінованою МТ називається упорядкована четвірка* $M = (K, X, \delta, s_0)$*, де* $K$ *– скінченна множина, елементи якої називаються* **станами ДМТ**, $X$ *– скінченна множина попарно різних елементів, що називається* **алфавітом ДМТ** (*причому* $K \cap X = \emptyset$), *а алфавіт* $X$ *включає символи* # (*пустий символ*), $\triangleright$ (*початковий символ*) *і пусте слово* $e$, $\delta : K \times X \to$ $\to K' \times X \times \{l, r, t\}$ – **функція переходів** *ДМТ, де* $K' = K \cup$ $\cup \{h, \text{“}yes\text{”}, \text{“}no\text{”}\}$ *і* $s_0 \in K$ – **початковий стан ДМТ**.

Символи $h$, *“yes”, “no”* означають відповідно **заключний (або фінальний) стан, стан, що сприймає, і стан, що не сприймає**, а символи $l, r, t$ означають напрямок руху головки ДМТ відповідно ліворуч, праворуч і незмінність позиції. Нехай $X^*$ означає множину

всіх скінченних слів у алфавіті $X$.

Робота ДМТ відбувається таким чином. Деяке слово $q = \triangleright\, p \in$ $\in X^*$ записано на стрічці ДМТ. У початковий момент часу ДМТ перебуває в початковому стані $s_0$ і оглядає перший символ слова $q$ – символ $\triangleright$. Слово $p \in X^*$ називається **вхідним словом** або **вхідними даними ДМТ**. З урахуванням своєї початкової комбінації, ДМТ виконує крок обчислень таким чином. Для кожної комбінації поточного стану $k \in K$ і символу $x \in X$ функція $\delta$ визначає трійку $(k', y, d)$, де $k'$ – черговий стан, $y \in X$ – символ, який записується на стрічці замість символу $x$, а $d \in \{l, r, t\}$ визначає напрямок руху головки ДМТ. У випадку, коли робота ДМТ тільки починається, завжди $\delta(s_0, \triangleright) = (k', \triangleright, r)$, тобто символ $\triangleright$ завжди приводить до руху головки ДМТ праворуч і на його місце не можна нічого записувати. Ця обставина дає можливість розглядати ДМТ зі стрічкою, необмеженою в одну (праву) сторону, оскільки вихід за початковий символ $\triangleright$ у такій ситуації неможливий.

Попри те, що головка не може вийти за лівий кінець вхідного слова $p$, вона може вийти за його правий кінець. У цьому разі говорять, що головка читає пустий символ #, на місце якого може бути записаний якийсь символ з алфавіту $X$. Таким чином, послідовність символів на стрічці може зростати, а це необхідно для того, щоб виконувати довільні обчислення за допомогою ДМТ.

Зупинка ДМТ настає тоді, коли вона в процесі обчислень досягає одного з трьох станів $h$, *"yes"*, *"no"*. Якщо ДМТ зупинилась

– у стані *"yes"*, то говоримо, що ДМТ **сприймає** або **розпізнає вхідне слово** $p \in X^*$;

– у стані *"no"*, то говоримо, що ДМТ **не сприймає** або **не розпізнає вхідне слово** $p \in X^*$;

– у стані $h$, то це означає невизначеність функції переходів ДМТ, і в цьому випадку результатом роботи ДМТ вважається слово, записане на її стрічці в момент зупинки.

Оскільки обчислення у разі зупинки ДМТ відбуваються в скінченному часі, то на стрічці буде записане слово скінченної довжини, яке починається символом $\triangleright$ і закінчується символом, який передує першому пустому символу, якщо дивитися символи слова зліва

направо (це слово може бути, зокрема і пустим). Якщо слово $q$ є результатом роботи ДМТ на вхідному слові $p$, то пишемо $M(p) = q$.

Може трапитися й таке, що ДМТ ніколи не досягає заключного стану і тому її робота триватиме нескінченно. У цьому випадку вважається, що результат роботи ДМТ на вхідному слові $p$ **невизначений** або, що ДМТ **незастосовна до вхідного слова** $p$.

Наведене означення ДМТ відрізняється від загальноприйнятого, оскільки в загальноприйнятому означенні ДМТ її стрічка нескінченна в обидві сторони. Але це не впливає на обчислювальну можливість МТ, оскільки справедлива теорема 9.

**Теорема 9.** *Для довільної ДМТ $M$ з нескінченною в обидві сторони стрічкою існує еквівалентна їй ДМТ $M'$ з нескінченною в одну (праву) сторону стрічкою [15].*

Отже, далі будемо користуватися лише поняттям ДМТ з нескінченною в одну (праву) сторону стрічкою, оскільки обчислювальна потужність ДМТ при цьому не змінюється.

Операції, що виконуються ДМТ у процесі обчислень, можна визначити формально за допомогою поняття **конфігурації ДМТ**.

**Означення 16.** *Конфігурацією ДМТ називається трійка $(s, p, q)$, де $s \in K$, а слова $p$ і $q$ з $X^*$ такі, що $p$ є словом, яке записане на стрічці ДМТ зліва від клітинки, яку оглядає головка, включаючи і символ, записаний у цій клітинці, а $q$ – слово, записане на стрічці праворуч від клітинки, яку оглядає головка ДМТ.*

Поняття конфігурації дає можливість формально описати функціонування ДМТ.

**Означення 17.** *Нехай $M$ – деяка ДМТ. Говоримо, що ДМТ $M$ із конфігурації $(s, p, q)$ безпосередньо досягає конфігурації $(s', p', q')$ (позначення $(s, p, q) \xrightarrow{M} (s', p', q')$), якщо виконуються такі умови:*

а) $\delta(s, x) = (s', y, d)$, *якщо $x$ останній символ слова $p$;*

б) *якщо $d = l$, то $p'$ є словом, що отримане зі слова $p$ шляхом викреслювання символу $x$, а $q' = yq$;*

в) *якщо $d = r$, то $p' = py$, а $q'$ є словом, що отримане зі слова $q$ шляхом викреслювання його першого символу;*

г) *якщо* $d = t$, *то* $p'$ *є словом, у якого останній символ* $x$ *замінений на символ* $y$, *а* $q' = q$.

Нехай $M^*$ означає транзитивне замикання відношення безпосередньої досяжності для конфігурацій ДМТ $M$ (позначення $(s,p,q) \xrightarrow{M^*} (s',p',q')$). Відношення $M^*$ називається просто відношенням досяжності для конфігурацій і означає, що існує таке натуральне число $n \in \mathcal{N}$, для якого справедливі такі переходи $(s,p,q) \xrightarrow{M^n} (s',p',q')$, тобто для всіх $i = 1,2,\ldots,n$ маємо $s = s_1,\ s_{n+1} = s',\ p_1 = p,\ q_{n+1} = q'$ і $(s_i,p_i,q_i) \xrightarrow{M} (s_{i+1},p_{i+1},q_{i+1})$. У цьому випадку говорять також, що ДМТ із конфігурації $(s,p,q)$ досягає конфігурації $(s',p',q')$ або переходить до неї за $n$ кроків.

**МТ як алгоритми**. МТ є природними моделями для розв'язання проблем на словах, а саме: **обчислення функцій на словах**, а також **розпізнавання мов**.

**Означення 18.** *Нехай* $L \subseteq (X \setminus \{\#\})^*$ *– деяка мова в алфавіті* $X' = X \setminus \{\#\}$, *а* $M$ *– ДМТ така, що для довільного слова* $p \in (X \setminus \{\#\})^*$ *отримуємо* $M(p) =$ *"yes", якщо* $p \in L$, *і* $M(p) =$ *"no", якщо* $p \notin L$. *У цьому випадку говорять, що ДМТ* $M$ **розв'язує мову** $L$.

*Мова* $L$ *називається* **рекурсивною**, *якщо існує ДМТ, яка розв'язує цю мову.*

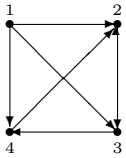
*Говоримо, що ДМТ* $M$ *тільки* **розпізнає мову** $L$, *якщо для довільного слова* $p \in (X \setminus \{\#\})^*$ *маємо* $M(p) =$ *"yes" тоді і тільки тоді, коли* $p \in L$, *і* $M$ *незастосовна до* $p$, *якщо* $p \notin L$.

ДМТ застосовують не тільки до розпізнавання мов, а й для обчислення значень словарних функцій.

**Означення 19.** *Функція* $f : (F(X) \setminus \{\#\}) \to F(X)$, *де* $X$ *– деякий алфавіт, називається* **словарною функцією**.

*Нехай* $M$ *– деяка ДМТ в алфавіті* $X$. *Говорять, що словарна функція* $f$ *обчислюється ДМТ* $M$, *якщо для довільного слова* $p \in (F(X)\setminus\{\#\})$ *справедлива рівність* $M(p) = f(p)$. *Якщо така ДМТ існує, то функція* $f$ *називається* **частково рекурсивною**.

**Зображення проблем словами.** Слова в алфавіті $X \setminus \{\triangleright, \#\} = \{0, 1\}$, про що йшлося вище, служать для зображення проблем. З такого підходу має бути очевидним, що зображення проблеми у вигляді слова в алфавіті $\{0, 1\}$ є достатньо загальним: довільний скінченний об'єкт може бути зображений словом скінченної довжини в цьому алфавіті. Наприклад, такий об'єкт, як скінченний граф, може бути записаний у вигляді слова скінченної довжини в алфавіті $\{0, 1\}$. Справді, використовуючи матрицю інцидентності, можна наведений нижче граф подати таким словом у цьому алфавіті:




"(1,10),(1,11),(1,100),(10,11),
(11,100),(100,10)"
списки інцидентності
"(0111, 0010, 0001, 0100)"
матриця інцидентності

Як випливає із цього простого прикладу, цілі числа, скінченні множини, скінченні графи тощо можна зображувати різними способами, і такі зображення можуть відрізнятися одне від одного як формою, так і довжиною. Але відома важлива властивість таких зображень:

**Усі допустимі зображення-кодування є поліноміально еквівалентними**.

Слова "поліноміально еквівалентні" означають, що коли $A$ і $B$ – два сенсові зображення проблеми $P$ і зображення $A$ у вигляді слова має довжину $n$, то представлення $B$ тієї самої проблеми $P$ у вигляді слова має найбільшу довжину $p(n)$, де $p(n)$ – деякий поліном. Наприклад, зображення скінченного графа без ізольованих вершин у вигляді матриці інцидентності вимагає найбільшої довжини слова, яка є квадратом довжини зображення цього графа у вигляді списків інцидентності.

Зауважимо, що теорія складності обчислень не залежить від зображення проблеми і її можна розв'язувати незалежно від слів і мов. Але розумний вибір представлення робить її результати адекватними реальним проблемам і практичній очислювальності.

### 2.3.2. Багатострічкові МТ

Для означення часу і пам'яті, необхідних для оцінювання складності проблеми, потрібне певне узагальнення ДМТ, а саме, багатострічкові МТ, тобто ДМТ з декількома стрічками. Це узагальнення, як буде показано далі, можна змоделювати за допомогою звичайної однострічкової ДМТ. Отже, прийняття більшої кількості стрічок у моделі ДМТ не виводить нас за клас МТ.

**Означення 20.** *ДМТ із $k$ стрічками ($k \geq 1$) називається четвірка $M = (K, X, \delta, s_0)$, де $K$ і $X$ ті ж, що і в означенні звичайної однострічкової ДМТ, а функція переходів $\delta$, яка називається програмою, визначає наступний стан таким чином:*

$$\delta : K \times X^k \to (K \cup \{h, \text{“}yes\text{”}, \text{“}no\text{”}\}) \times (X \times \{l, r, t\})^k,$$

*де $\delta(s, y_1, \ldots, y_k) = (s', z_1, d_1, \ldots, z_k, d_k)$ означає, що коли ДМТ в деякий момент перебуває в стані $s$, головка на першій стрічці оглядає символ $y_1$ і т. д., головка на $k$-й стрічці оглядає символ $y_k$, то в наступний момент ДМТ перебуватиме в стані $s'$, головка на першій стрічці запише символ $z_1$ замість символу $y_1$ і перейде або залишиться на місці залежно від значення $d_1$ і т. д., головка на $k$-й стрічці запише символ $z_k$ замість символу $y_k$ і перейде або залишиться в тій самій позиції, залежно від значення $d_k$.*

На місце символу $\triangleright$ не дозволяється нічого записувати, тобто це означає, що коли $y_i = \triangleright$, то $d_i = r$. На початку на всіх стрічках записано символ $\triangleright$, а на першій стрічці записано ще й вхідне слово $p$ (тобто $q = \triangleright\, p$).

Результат роботи багатострічкової ДМТ визначається так само, як і для звичайної ДМТ, з тією лише відмінністю, що результат обчислень словарної функції після зупинки ДМТ записується на останній $k$-й стрічці.

**Приклад 2.3.2.** Паліндроми можна розв'язувати ефективніше за допомогою двострічкової ДМТ, порівняно з однострічковою ДМТ. Двострічкова ДМТ починає свою роботу з копіювання вхідного слова з першої стрічки на другу. Після цього ДМТ встановлює головку першої стрічки на перший символ вхідного слова, а головку другої

стрічки – на останній символ скопійованого слова. А далі робота ДМТ зводиться до руху головок у протилежних напрямках і порівняння на кожному кроці символів, що оглядаються головками. У разі рівності цих символів символ на другій стрічці замінюється пустим (тобто цей символ просто стирається на другій стрічці).

Таблицю функції переходів цієї ДМТ наведено нижче.

| $k \in K$ | $x \in X$ | $y \in X$ | $\delta(k,x,y)$ |
|---|---|---|---|
| $s_0$ | 0 | # | $(s_0, 0, r, 0, r)$ |
| $s_0$ | 1 | # | $(s_0, 1, r, 1, r)$ |
| $s_0$ | $\triangleright$ | $\triangleright$ | $(s_0, \triangleright, r, \triangleright, r)$ |
| $s_0$ | # | # | $(s, \#, l, \#, t)$ |
| $s$ | 0 | # | $(s, 0, l, \#, t)$ |
| $s$ | 1 | # | $(s, 1, l, \#, t)$ |
| $s$ | $\triangleright$ | # | $(s_1, \triangleright, r, \#, l)$ |
| $s_1$ | 0 | 0 | $(s_1, 0, r, \#, l)$ |
| $s_1$ | 1 | 1 | $(s_1, 1, r, \#, l)$ |
| $s_1$ | 0 | 1 | $(\text{“}no\text{”}, 0, t, 1, t)$ |
| $s_1$ | 1 | 0 | $(\text{“}no\text{”}, 1, t, 0, t)$ |
| $s_1$ | # | $\triangleright$ | $(\text{“}yes\text{”}, \#, t, \triangleright, r)$ |

♠

**Означення 21.** *Конфігурацією $k$-стрічкової ДМТ $M$ називається кортеж вигляду $(s, p_1, q_1, \ldots, p_k, q_k)$, де $s$ – поточний стан ДМТ, $p_i, q_i$ – слова, що записані на $i$-й стрічці $(1 \le i \le k)$, головка якої оглядає останній символ слова $p_i$.*

*Говоримо, що ДМТ $M$ із конфігурації $(s, p_1, q_1, \ldots, p_k, q_k)$ безпосередньо досягає конфігурації $(s', p'_1, q'_1, \ldots, p'_k, q'_k)$ (позначення $(s, p_1, q_1, \ldots, p_k, q_k) \xrightarrow{M} (s', p'_1, q'_1, \ldots, p'_k, q'_k)$), якщо виконуються такі умови. Нехай $p_i = p_{i-1}y_i$, $q_i = xq_{i-1}$ для $i = 1, 2, \ldots, k$ і нехай $\delta(s, y_1, \ldots, y_k) = (s', z_1, d_1, \ldots, z_k, d_k)$. Тоді для кожного $i = 1, 2, \ldots, k$*

– *якщо $d_i = r$, то $p'_i = p_{i-1}z_ix$, а $q'_i = q_{i-1}$;*

– *якщо $d_i = l$, то $p'_i = p_{i-1}$, а $q'_i = z_ixq_{i-1}$;*

– *якщо $d_i = t$, то $p'_i = p_{i-1}z_i$, а $q'_i = q_i$.*

*Тобто, для кожної стрічки мають виконуватися умови переходу від однієї конфігурації до іншої, як для ДМТ з однією стрічкою.*

*Транзитивне замикання відношення безпосередньої досяжності для конфігурацій $k$-стрічкової ДМТ називається* **відношенням досяжності** *для конфігурацій (позначення $(s, p_1, q_1, \ldots, p_k, q_k) \xrightarrow{M^*} (s', p'_1, q'_1, \ldots, p'_k, q'_k)$). Це означає, що існує таке число $n \in \mathcal{N}$, що*

$$(s, p_1, q_1, \ldots, p_k, q_k) \xrightarrow{M^n} (s', p'_1, q'_1, \ldots, p'_k, q'_k),$$

*і в цьому випадку будемо говорити, що друга конфігурація досяжна з першої конфігурації за n кроків.*

$k$-стрічкова ДМТ починає обчислення на вхідному слові $p$, перебуваючи в конфігурації $(s_0, \triangleright, p, \triangleright, e, \ldots, \triangleright, e)$, де $e$ – пусте слово. Це означає, що вхідне слово $p$ записується на першій стрічці, а на решті стрічок записаний тільки початковий символ $\triangleright$. Якщо

$$(s_0, \triangleright, p, \triangleright, e, \ldots, \triangleright, e) \xrightarrow{M^*} (\text{“}yes\text{”}, p_1, q_1, \ldots, p_k, q_k)$$

для деяких слів $p_1, q_1, \ldots, p_k, q_k$, то говоримо, що $M(p) =$ “$yes$”. Якщо $(s_0, \triangleright, p, \triangleright, e, \ldots, \triangleright, e) \xrightarrow{M^*} (\text{“}no\text{”}, p_1, q_1, \ldots, p_k, q_k)$ для деяких слів $p_1, q_1, \ldots, p_k, q_k$, то говоримо, що $M(p) =$ “$no$”. Нарешті, якщо ДМТ зупиняється в конфігурації $(h, p_1, q_1, \ldots, p_k, q_k)$, то $M(p) = p'_k q_k$, де $p'_k$ збігається зі словом $p_k$, взятим без першого символу $\triangleright$, а $q_k$ – слово, яке записане без пустих символів. Отже, у разі зупинки $k$-стрічкової ДМТ в стані $h$ результат записується на останній $k$-й стрічці. Завдяки цьому звичайна однострічкова ДМТ є окремим випадком $k$-стрічкової ДМТ за $k = 1$. Крім того, це дає змогу визначити поняття обчислюваної словарної (рекурсивної) функції, розв'язуваності й розпізнавання мов аналогічно тому, як це визначалось для однострічкових ДМТ.

### 2.3.3. Класи $P, PSPACE, L$ і нижче

Для побудови класів складності використовують тристрічкові ДМТ. Точніше, така ДМТ має 3 стрічки, де

– на першій стрічці, яка називається **вхідною**, записується вхідне слово і з цієї стрічки дозволяється лише читати і не дозволяється нічого писати;

– друга стрічка, яка називається **робочою** або **внутрішньою**, така, що на неї можна писати і з неї можна читати;

– третя стрічка, яка називається **вихідною**, така, що на неї дозволяється тільки писати і не дозволяється нічого з неї читати.

Розв'язання проблеми такою МТ виконується так. На вхідній стрічці записується вхідне слово в алфавіті $X = \{0, 1, \triangleright, \#\}$, починаючи з "найлівішої" клітинки (тобто з першої клітинки стрічки), а в решту клітинок стрічки записано символ # – символ пустої клітинки. Машина працює із цим словом відповідно до своєї програми (функції переходів), записує необхідні символи на вихідну і робочі стрічки. Запис на вихідну стрічку теж виконується з "найлівішої" клітинки, а в решту клітинок стрічки записано символ #. Якщо в деякий момент машина зупиняється з певною відповіддю, то вона записує цю відповідь – "*yes*" або "*no*" – на вихідну стрічку.

В теорії складності обчислень принциповий інтерес мають два ресурси для базової моделі обчислень – *час* і *пам'ять*.

**Означення 22.** *Якщо для тристрічкової ДМТ і вхідного слова $p$ справедливе відношення*

$$(s_0, \triangleright, p, \triangleright, e, \triangleright, e) \xrightarrow{M^t} (H, p_1, q_1, p_2, q_2, p_3, q_3),$$

*де $H \in \{h, \text{"}yes\text{"}, \text{"}no\text{"}\}$, то число $t$ називається* **часом обчислень** *або* **часовою складністю** *розв'язання проблеми. Інакше кажучи, часовою складністю розв'язання проблеми називається число кроків тристрічкової ДМТ, які вона виконала до зупинки.*

**Пам'яттю** *або* **простором обчислень** *називається число клітинок робочої стрічки, які використала ДМТ під час обчислень.*

Зауважимо, що для ДМТ пам'ять, яка використовувалась під час обчислень, може бути набагато меншою, ніж розмір вхідного слова, а час обчислень має бути принаймні не меншим від довжини вхідного слова. Це пов'язано з тим, що відповідь часто залежить тільки від деякого початкового підслова вхідного слова, а не від усього слова.

**Означення 23.** *Нехай $f : \mathcal{N} \to \mathcal{N}$ – деяка функція. Говоримо, що ДМТ $M$* **працює в часі** *$f(n)$, якщо для довільного вхідного слова $p$ час обчислень, необхідний ДМТ $M$ для розв'язання проблеми $p$, дорівнює $f(l(p))$, де $l(p) = n$ – довжина слова $p$. Функція $f$ називається* **часовим обмеженням** *ДМТ $M$.*

Тепер можна ввести поняття класу складності, використовуючи тристрічкові ДМТ і розв'язувані ними мови.

**Означення 24.** *Нехай* $L \subseteq (X \cup \{\#\})^*$ *– мова. Говоримо, що мова* $L$ *належить* **класу** $\mathbf{TIME}(f(n))$*, якщо існує тристрічкова ДМТ, яка розв'язує мову* $L$ *у часі* $f(n)$*. Множину мов* $\mathbf{TIME}(f(n))$ *називають* **класом часової складності**.

*Якщо мова* $L$ *розв'язується тристрічковою ДМТ в поліноміальному часі, тобто* $f(n) = n^k$*, де* $k$ *– стала величина, то клас таких мов за всіма* $k$ *складає* **поліноміальний клас часової складності** *і позначається* $\mathbf{P}$*. Отже,*

$$\mathbf{P} = \mathbf{TIME}(n^k) = \bigcup_{j>1} \mathbf{TIME}(n^j).$$

*Говоримо, що мова* $L$ *належить до класу* $\mathbf{SPACE}(f(n))$*, якщо існує тристрічкова ДМТ, яка розв'язує мову* $L$ *і використовує не більше* $f(n)$ *клітинок робочої стрічки. Множину мов* $\mathbf{SPACE}(f(n))$ *називають* **класом складності по пам'яті**.

*Якщо мова* $L$ *розв'язується тристрічковою ДМТ при поліноміальній пам'яті, тобто* $f(n) = n^k$*, де* $k$ *– стала величина, то клас таких мов за всіма* $k$ *складає* **поліноміальний клас складності по пам'яті** *і позначається* $\mathbf{PSPACE}$*, тобто*

$$\mathbf{PSPACE} = \mathbf{SPACE}(n^k) = \bigcup_{j>1} \mathbf{SPACE}(n^j).$$

*Говоримо, що мова* $L$ *належить класу* $\mathbf{SPACE}(\log n)$*, якщо існує тристрічкова ДМТ, яка розв'язує мову* $L$ *і використовує не більше* $\log n$ *клітинок робочої стрічки. Множину мов* $\mathbf{SPACE}(\log n)$ *називають* **класом логарифмічної складності по пам'яті** *і позначають через* $\mathbf{L}$ *(або іще* $\mathbf{LOGSPACE}$*), тобто*

$$\mathbf{L} = \mathbf{SPACE}(\log n).$$

Зауважимо, що визначені вище класи складності $\mathbf{P}$, $\mathbf{PSPACE}$, $\mathbf{L}$ є найважливішими і найпопулярнішими класами в теорії складності обчислень. Розглянемо приклади.

**Приклад 2.3.3.** Неважко показати, що однострічкова ДМТ розпізнає паліндроми в часі $f(n) = n^2$. Робота цієї ДМТ виконується у два етапи. На першому етапі за $2n+1$ кроків виконується порівняння першого і останнього символів. Потім повторюється та сама робота над словом довжини $n-2$, далі – над словом довжини $n-4$ і т. д. Отже, загальне число кроків ДМТ у найгіршому випадку буде $f(n) = \frac{(n+1)(n+2)}{2}$. Звідси випливає, що мова паліндромів належить до класу $\mathbf{TIME}(\frac{(n+1)(n+2)}{2}) = O(n^2)$. Слід зазначити, що одержана оцінка є найбільш песимістичною, тому що для слова $01^n$ ця ДМТ за $n+3$ кроків визначає, що дане слово не є паліндромом. Але, обчислюючи $f(n)$, необхідно брати до уваги найгірший із можливих варіантів вхідних даних.

Двострічкова ДМТ з прикладу 2.3.2 розв'язує мову паліндромів у часі $f(n) = 3n+3 = O(n)$, отже, ця мова належить класу складності $\mathbf{TIME}(3n+3)$. ♠

Аналогічно до означення класів **P, PSPACE, L** визначають і нижченаведені класи складності, які часто трапляються у застосуваннях. Ці класи лежать в ієрархії нижче від класу **P** і їхнє означення таке:

– **константа:** існує така константа $k$, що мова $L$ розв'язується ДМТ за $k$ кроків. Клас всіх таких мов позначається $\mathbf{TIME}(k)$;

– **логарифм:** мова $L$ розв'язується ДМТ за $\log n$ кроків. Клас усіх таких мов позначається $\mathbf{TIME}(\log n)$;

– **полілогарифм:** існує така константа $k$, що мова $L$ розв'язується ДМТ за $\log^k n$ кроків. Клас усіх таких мов позначається $\mathbf{TIME}(\log^k n)$;

– **лінійна:** мова $L$ розв'язується ДМТ за $n$ кроків. Клас усіх таких мов позначається $\mathbf{TIME}(n)$.

Аналогічно визначають і класи складності по пам'яті.

Наведена класифікація ґрунтується на такому твердженні.

**Теорема 10.** *Для довільної $k$-стрічкової ДМТ $M$, яка працює в часі $f(n)$, існує однострічкова ДМТ $M'$, що працює в часі $O(f(n)^2)$, така, що для довільного слова $p \in F(X)$ виконується рівність $M(p) = M'(p)$ [15].*

Із цієї теореми випливає, що збільшення кількості стрічок не збільшує обчислювальної потужності багатострічкових ДМТ порівняно з однострічковими. Єдине, до чого приводить збільшення кількості стрічок – це поліноміальне підвищення ефективності такої ДМТ.

### 2.3.4. Недетерміновані МТ. Класи $NP, NSPACE$

Недетермінована МТ визначається аналогічно ДМТ, тільки відношення переходів не є функцією. Це означає, що множина можливих обчислень може розгалужуватися, як дерево. Наведемо формальні означення.

**Означення 25.** *Недетермінованою МТ (НДМТ) називається четвірка $M = (K, X, \Delta, s_0)$, де $K, X, s_0$ – ті самі об'єкти, що і в ДМТ, а $\Delta$ – відношення переходів, яке визначає декілька можливих станів, у які МТ може перейти. Це значить, що НДМТ має можливість вибору між кількома діями і $\Delta$ вже не є функцією, тобто*

$$\Delta \subset (K \times X) \times ((K \cup \{h, \text{“}yes\text{”}, \text{“}no\text{”}\}) \times X \times \{l, r, t\}).$$

Таким чином, для кожної комбінації стану і символу може існувати більше одного наступного допустимого стану або такого стану може не існувати жодного.

**Конфігурацією НДМТ** називається трійка $(s, p', q')$, де $s \in \in K$, а $p'$ і $q'$ – слова, відповідно ліворуч, включаючи і символ, який оглядає головка, і праворуч від головки НДМТ. Говоримо, що НДМТ переходить від конфігурації $(s, p', q')$ до конфігурації $(s', p'', q'')$ за один крок, якщо існує такий рух головки $((s, x), (s', y, d)) \in \Delta$, що

а) $d = r$ і слово $p''$ збігається зі словом $p'$, в якому останній символ $x$ замінено символом $y$ і перший символ слова $q'$ дописано останнім символом до слова $p''$ (можливо, цей символ пустий $\#$, якщо $q' = e$), а слово $q''$ збігається зі словом $q'$, у якого стерто перший символ;

б) $d = l$ і слово $p''$ збігається зі словом $p'$ без останнього символу $x$, який замінено символом $y$, і цей символ $y$ дописується першим символом до слова $q''$;

в) $d = t$ і слово $p''$ збігається зі словом $p'$, в якому останній символ $x$ замінено символом $y$ і $q' = q''$.

Відношення $\overset{NM^k}{\longrightarrow}$ і $\overset{NM^*}{\longrightarrow}$ можна визначити так само, як і для ДМТ, але слід пам'ятати, що $\Delta$ є не функцією, а відношенням.

**Означення 26.** *Нехай NM – деяка НДМТ і L – мова. Говоримо, що NM* **розв'язує мову** *L, якщо для кожного* $p \in X^*$ *виконується:* $p \in L$ *тоді і тільки тоді, коли* $(s_0, \triangleright, p) \overset{NM^*}{\longrightarrow} (\text{“}yes\text{”}, p'q')$, *де* $p', q'$ – *деякі слова.*

Це означення свідчить, що НДМТ мають певні переваги над іншими моделями обчислень. Вхідне слово $p$ сприймається тоді і тільки тоді, коли існує деяка послідовність недетермінованих переходів, яка переводить НДМТ у стан “*yes*”. Інші послідовності переходів можуть не сприймати слово $p$, але для сприйняття $p$ достатньо наявності хоча б однієї такої послідовності. Слово $p$ не сприймається тоді і тільки тоді, коли не існує жодної послідовності переходів, яка переводить НДМТ у стан “*yes*”.

**Означення 27.** *NM розв'язує мову L у часі f(n)* ($f : \mathcal{N} \to \mathcal{N}$), *якщо NM розв'язує L і, крім того, для довільного* $p \in X^*$, *такого, що* $(s_0, \triangleright, p) \overset{NM^k}{\longrightarrow} (\text{“}yes\text{”}, p', q')$, *справедлива нерівність* $k \leq f(l(p))$.

Із цього означення випливає, що $NM$ розв'язує мову $L$ у часі $f(l(p))$, якщо всі послідовності, що переводять $NM$ у стан “*yes*”, не перевищують довжини $f(l(p))$.

Множина мов, що розв'язуються НДМТ, є класом складності $\mathbf{NTIME}(f(n))$. Важливим класом складності є клас **NP**, який означається як теоретико-множинне об'єднання класів $\mathbf{NTIME}(n^k)$, де $k$ – деяке стале число, тобто

$$\mathbf{NP} = \bigcup_{k>1} \mathbf{NTIME}(n^k).$$

Зауважимо, що клас **P** є підкласом класу **NP**, оскільки клас ДМТ є підкласом НДМТ. Іншими важливими класами складності є класи

$$\mathbf{NPSPACE} = \mathbf{NSPACE}(n^k) \text{ і } \mathbf{EXP} = \mathbf{TIME}(2^{n^k}).$$

Між ДМТ і НДМТ існує зв'язок, який виражається таким твердженням [34].

**Теорема 11.** *Нехай деяка НДМТ NM розв'язує мову L у часі f(n), тоді існує тристрічкова ДМТ M, яка розв'язує мову L у часі* $O(c^{f(n)})$*, де c – константа, що залежить від NM, тобто*

$$\mathbf{NTIME} \subseteq \bigcup_{c>1} \mathbf{TIME}(c^{f(n)}).$$

Тепер уточнимо поняття обмеженості за пам'яттю для тристрічкової НДМТ $NM = (K, X, \Delta, s_0)$.

**Означення 28.** *Говоримо, що НДМТ NM розв'язує мову L у пам'яті* $f(n)$*, якщо NM розв'язує мову L і для довільного* $p \in$ $\in (X \setminus \{\#\})^*$*, для якого* $(s_0, \triangleright, p, \triangleright, e, \triangleright, e) \overset{NM^*}{\longrightarrow} (s, p_1, q_1, p_2, q_2, p_3, q_3)$*, справедлива нерівність* $l(p_2 q_2) \leq f(l(p))$.

Це означає, що НДМТ $NM$ під час обчислень жодного разу не потребує клітинок на її робочій стрічці більше, ніж значення функції $f$ на довжині вхідного слова $p$. Зауважимо, що в цьому означенні навіть не вимагається, щоб усі обчислення НДМТ були скінченними. Та все ж, для означення деяких типів проблем і алгоритмів за допомогою НДМТ будемо припускати, що її дерево досяжних конфігурацій є скінченним і включає всі фінальні конфігурації, тобто конфігурації, в яких НДМТ зупиняється, причому глибина цих конфігурацій така ж, як і глибина самого дерева. У цьому разі простором-пам'яттю, що використовується, є максимальна кількість клітинок на робочій стрічці, що відвідуються, за всіма конфігураціями цього дерева.

### 2.3.5. Редукція і повнота

Для порівняння проблем за складністю їхнього розв'язання, користуються поняттям **редукції**. Неформально редукція проблеми $B$ до проблеми $A$ означає існування функції $R$, у результаті якої для довільного окремого випадку проблеми $B$ є еквівалентний йому окремий випадок $R(p)$ проблеми $A$. Під еквівалентністю двох окремих випадків розуміють те, що відповідь на питання, чи є $R(p)$ позитивним прикладом проблеми $A$, буде одночасно відповіддю на

питання, чи є $p$ позитивним прикладом проблеми $B$. Інакше кажучи, розв'язати проблему $B$ для прикладу $p$ – це те саме, що розв'язати проблему $A$ для прикладу $R(p)$ (рис. 2.3.2).

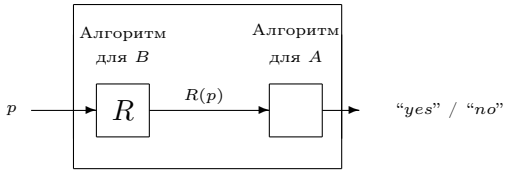



Рис. 2.3.2. Редукція $B$ до $A$

Якщо має місце ситуація, показана на рис. 2.3.2, то говоримо, що проблема $A$ не менш важка, ніж проблема $B$ за однієї умови. Ця умова стосується редукції $R$: *редукція $R$ не повинна бути складною в обчислювальному плані.* Якщо не зробити жодних обмежень відносно цієї складності, то можна дійти до абсурдних наслідків. Формальне означення редукції має вигляд.

**Означення 29.** *Говоримо, що мова $L_1$* **редукується до мови** *$L_2$, якщо існує словарна функція $R$ (функція, визначена на словах і зі значеннями у множині слів), яка обчислюється за допомогою ДМТ в логарифмічній пам'яті (тобто $O(\log n)$), така, що для кожного слова $p$ виконується $p \in L_1$ тоді і тільки тоді, коли $R(p) \in L_2$. Функція $R$ називається* **редукцією** *$L_1$ до $L_2$.*

Позначимо $L \leq_R L'$ той факт, що мова $L$ редукується до мови $L'$ за допомогою редукції $R$.

Звідси безпосередньо випливає, що редукції є поліноміальними алгоритмами, і це стверджує таке твердження.

**Теорема 12.** *а) Якщо $R$ є редукцією, яка обчислюється за допомогою деякої ДМТ $M$, то для кожного вхідного слова $p$ машина $M$ закінчує свою роботу через поліноміальне число кроків [34].*

*б) Нехай $L_1, L_2, L_3$ – алгоритмічно розв'язувані проблеми. Тоді*

*б1) якщо $L_1 \leq_{R_1} L_2$ і $L_2 \leq_{R_2} L_3$, то $L_1 \leq_R L_3$, де $R = R_1 * R_2$;*

*б2) якщо $L_1 \leq_R L_2$ і $L_2 \in P$, то $L_1 \in P$.*

**Означення 30.** *Проблема $L$ називається NP-повною, якщо (1) $L \in NP$ і (2) $\forall L_1 \in NP(L_1 \leq_R L)$.*

Клас $NP$-повних проблем позначають $NPC$.

**Приклади** $NPC$ **проблем**. а) **(Підмножина суми)** *Вхід:* множина невід'ємних цілих чисел $\{a_1, a_2, \ldots, a_n\}$ і натуральне число $S > 0$.

*Проблема:* існує чи ні підмножина чисел $\{a_{i_1}, a_{i_2}, \ldots, a_{i_k}\}$, така що $\sum\limits_{j=1}^{k} a_{i_j} = S$?

Відомо, що ця проблема в загальному випадку належить до класу $NPC$.

б) **(Дискретний логарифм)** *Вхід:* рівняння $y = g^x \pmod p$, де $p$ – просте число.

*Проблема:* обчислити $x$, якщо відомі числа $y, g, p$. Число $x$ називається *дискретним логарифмом.*

Відомо, що ця проблема належить класу $NPC$ і складність розв'язання цієї проблеми еквівалентна складності розв'язання проблеми факторизації натурального числа.

Найкращий алгоритм розв'язання проблеми факторизації натуральних чисел має складність $e^{((\ln\ p)^{1/3}(\ln(\ln\ p))^{2/3}(c+o(1))}$. Для великих чисел це складна задача. Дійсно, нехай комп'ютер обчислює добуток двох чисел із 90 знаками у часі $10^{-14}$ (для сучасних комп'ютерів цей час нереальний) і модуль має від 50 до 100 знаків, тоді час, потрібний комп'ютеру для факторизації: $T = 2 \cdot \log\ p \cdot 10^{-14} = 6 \cdot 10^{-12}$ секунд. А обчислення дискретного логарифма потребує $10^{45} \cdot 10^{-14} = 10^{31}$ секунд $> 10^{22}$ років. ♠

**Теорема 13.** *Нехай $L_1$ і $L_2$ – алгоритмічно розв'язувані проблеми. Тоді, якщо $L_1 \in NP$, $L_2 \in NPC$ і $L_2 \leq_R L_1$, то $L_1 \in NPC$.*

Звідси випливає, що для доведення того, що проблема $L_1$ буде $NP$-повною, потрібно

1) довести, що $L_1 \in NP$;

2) знайти відому проблему $L_2 \in NPC$;

3) довести, що $L_2 \leq_R L_1$.

**Означення 31.** *Алгоритмічно розв'язувана проблема $L$ називається NP-важкою, якщо існує проблема $L_1 \in NPC$ і $L_1 \leq_R L$.*

Описані вище класи складності називають класами складності в моделі Тьюрінга. У практичних застосуваннях користуються іншою моделлю складності, яку називають *арифметичною.* У моделі Тьюрінга для знаходження складності враховується довжина зображення вхідних даних, а в арифметичній моделі – лише кількість елементарних операцій, які виконуються над цими даними.

**Означення 32.** *В арифметичній моделі часовою складністю алгоритму на вхідних даних p називається кількість елементарних операцій, які виконуються алгоритмом для обчислення значень вихідних даних.*

"Елементарні операції" трактують різними способами: операції на бітах, порівняння слів або бітів, елементарні інструкції процесора, додавання або множення бінарних чисел тощо.

Алгоритм називається **строго поліноміальним**, якщо він має часову поліноміальну складність в арифметичній моделі і логарифмічну складність за пам'яттю в моделі Тьюрінга (або, що те саме, поліноміальну часову складність у моделі Тюрінга).

### 2.3.6. Односторонні функції

Односторонні функції відіграють важливу роль у криптографії. Про такого типу функції говорилося в пункті а) властивості 1 з підрозділу 1.1. Теорія складності обчислень дозволяє уточнити поняття односторонньої функції.

**Означення 33.** *Функція f(x) називається односторонньою, якщо*

*а) для кожного значення аргументу x складність обчислення значення y=f(x) належить класу P (поліноміальної складності), а складність обчислення значення $x = f^{-1}(y)$ належить класу NP (практично неможливо обчислити за розумний проміжок часу);*

*б) не існує пари значень $x, x'$ таких, що $x \neq x'$ і $f(x) = f(x')$.*

У криптографічних системах використовують функції, введені Діффі і Хеллманом в 1976 р., які називають *односторонніми функціями із секретом.*

**Означення 34.** *Односторонньою функцією із секретом називається функція $f_k(x) = y$, складність обчислення якої для всіх значень x належить до класу P, але обчислення $x = f_k^{-1}(y)$ майже для всіх значень y належить класу NP. Але, якщо скористатися секретною інформацією k, то для всіх значень y обчислення значення x такого, що $f_k(x) = y$, належить класу P.*

Це поняття є основним у сучасній криптографії. Гарантією існування односторонніх функцій служить гіпотеза, що $P \neq NP$. А питання існування односторонніх функцій взагалі залишається відкритим тому, що нинішній стан наших знань не дає підстав стверджувати, що односторонні функції існують. Але, не дивлячись на це, існують кандидати серед функцій, ефективне обчислення значень яких нам відоме, тоді як жодні ефективні алгоритми обчислення значень обернених функцій невідомі.

Першим прикладом кандидата на односторонню функцію є розкладання цілого числа на прості множники. Найпотужніший комп'ютер із найкращим у його розпорядженні алгоритмом не у змозі швидко (у поліноміальному часі) розкласти на множники тисячозначне ціле число, яке є добутком двох приблизно рівних простих чисел.

Другим важливим кандидатом на односторонню функцію є модульне піднесення до степеня або логарифмування в кільці лишків за модулем $n$ – $Z_n$. Піднесення до степеня обчислюється в цьому кільці швидко, а знаходження значення оберненої функції (логарифма) практично неможливе. Розглянемо приклади.

**Приклади односторонніх функцій**. 1) Нехай $X = \{1, 2, \ldots, 16\}$ і $f(x) = r_x$ для кожного $x \in X$, де $r_x$ – остача від ділення $3^x$ на 17. Тоді маємо

| $x$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $f(x)$ | 3 | 9 | 10 | 13 | 5 | 15 | 11 | 16 | 14 | 8 | 7 | 4 | 12 | 2 | 6 | 1 |

.

Якщо задані число $x \in [1, 16]$ і така таблиця, то відносно легко можна знайти прообраз $x$ для $y = f(x)$. Але коли маємо число (наприклад 7) і такої таблиці немає, то знайти $x$ для якого $f(x) = 7$ не є простою справою.

2) Візьмемо прості числа $p = 48611$, $q = 53993$ і обчислимо $m = pq = 2624653723$. Нехай $X = \{1, 2, \ldots, m-1\}$. Визначимо функцію на множині $X$ за допомогою рівності $f(x) = r_x$ для кожного $x \in X$, де $r_x$ – остача від ділення $x^3$ на $m$. Наприклад, $f(2489991) = 1981394214$, оскільки $2489991^3 = 5881949859 \cdot m + 1981394214$. Обчислення значення $f(x)$ відносно просте, але обчислення значення $x$ для $y = f(x) = 1981394214$ практично неможливе. ♠

Використання функцій із секретом дає можливість досить ефективно знаходити функцію, обернену до даної бієктивної функції.

Наприклад, у пункті 2) попереднього прикладу секретом є розклад числа $m = p \cdot q = 2624653723$. Тобто, числа $p$ і $q$ є секретом.

Як зазначалося, складнісна стійкість криптосистем ґрунтується на гіпотезі $P \neq NP$. Але гіпотеза $P \neq NP$ не є достатньою умовою стійкості криптосистеми навіть, коли вона ґрунтується на розв'язанні $NP$-повної проблеми. Прикладом нестійкої $NP$-повної проблеми є окремий випадок **проблеми упакування рюкзака**.

Існує декілька причин нестійкості.

*Перша причина* полягає в тому, що при встановленні $NP$-повноти розглядається найгірший випадок. Проблема включається до класу $NP$-проблем, якщо для неї існує лише декілька складних варіантів. Але зловмисник може натрапити на варіант проблеми, який не є важким. Отже, аналіз найгірших варіантів непридатний для вимірювання стійкості практичних криптосистем.

*Друга причина* полягає у складності встановлення нової, більш низької верхньої межі складності $NP$-проблеми, навіть якщо проблема є важкою для обчисленнь. Ця нова межа в кращому випадку залишається не доведеною.

*Третя причина* пов'язана з особливостями прикладної криптосистеми. Реальні криптосистеми є результатом компромісів між користувачами такої системи.

## 2.4. Елементи теорії ймовірностей

Нехай $S$ – деяка фіксована $n$-елементна множина. Над елементами із $S$ виконуються експерименти такі, що при кожному з них можливі $n$ несумісних і рівноможливих результатів $E_1, E_2, \ldots, E_n$. Слово "несумісний" означає, що коли в даному експерименті появляється результат $E_i$, то жоден із решти результатів у тому ж експерименті появитися не може. Кожний такий результат $E_i$ називається **елементарною подією**. Крім елементарних подій розглядаються також випадкові події, які складаються з елементарних подій. Наприклад, елементарною подією є результат $E_1 = 3$ під час підкидання грального кості, але нас може цікавити подія $A$ появи числа не більшого 3 на грані кості. Подія $A$ включає елементарні

події $E_1 = 1, E_2 = 2, E_3 = 3$. Отже, $A = \{E_1, E_2, E_3\}$.

**Означення 35.** *Нехай* $|S| = n$, $|A| = m$. *Імовірністю випадкової події* $A$ *називається відношення кількості несумісних рівноможливих елементарних подій, які складають подію* $A$ (*тобто, числа* $m$), *до кількості всіх можливих подій* (*тобто, до числа* $n$). *Ця ймовірність позначається* $p(A)$ *і дорівнює* $\frac{m}{n}$. *Множина* $S$ *називається* **полем подій**.

З означення ймовірності випадкової події $A$ випливає, що

1) $0 \le p(A) \le 1$, 2) $p(S) = 1$ і 3) $p(\emptyset) = 0$.

Подія $\emptyset$ називається **неможливою**, оскільки їй не відповідає жодна елементарна подія, а подія $S$ називається **достовірною**. Якщо $p(A) = \frac{m}{n}$, то ймовірність того, що подія $A$ не наступить, така: $\frac{n-m}{n} = 1 - p(A)$. Подія $\bar{A}$, яка означає, що подія $A$ не наступить, називається *протилежною* до події $A$.

### 2.4.1. Властивості ймовірностей. Випадкові величини

Нехай $A \cup B$ означає появу однієї з подій $A$ або $B$, яку називають *сумою подій*, а $A \cap B$ означає подію появи обох подій $A$ і $B$ одночасно, яку називають *добутком подій*.

**Правило додавання ймовірностей**. Тоді

1) $p(A \cup B) = p(A) + p(B) - p(A \cap B)$;

2) $p(A \cup B) = p(A) + p(B)$, якщо $A \cap B = \emptyset$, тобто випадкові події $A$ і $B$ несумісні;

3) якщо $\bigcup\limits_{k=1}^{n} E_k = S$ і $E_i \cap E_j = \emptyset$ $(i \ne j)$, то $\sum\limits_{k=1}^{n} p(E_k) = 1$.

Правило 1) обґрунтовується тим, що коли $p(A) = \frac{m_1}{n}$, $p(B) = \frac{m_2}{n}$ і $|A \cap B| = k$, то $p(A \cap B) = \frac{k}{n}$. Отже, $p(A \cup B) = \frac{m_1+m_2-k}{n} = p(A) + p(B) - p(A \cap B)$.

**Правило множення ймовірностей**. Нехай подія $A$ появляється $m_1$ разів серед $n_1$ рівноймовірних результатів першого експеримента, а подія $B$ – $m_2$ разів серед $n_2$ рівноймовірних результатів другого експерименту. Тоді $p(A) = \frac{m_1}{n_1}$ і $p(B) = \frac{m_2}{n_2}$. Розглянемо тепер експеримент, який полягає в тому, що одночасно появляються

обидві події. У такому випадку поле $S$ складається з $n_1 n_2$ елементів, а події $A \cap B$ відповідатимуть $m_1 m_2$ результатів. Отже,

$$p(A \cap B) = \frac{m_1 m_2}{n_1 n_2} = \frac{m_1}{n_1} \cdot \frac{m_2}{n_2} = p(A)p(B).$$

Одержана рівність називається **правилом множення ймовірностей**. Це правило узагальнюється таким чином: нехай $A_1, A_2, \ldots, A_k$ – деякі взаємно незалежні події, тобто умови експерименту, з результатом якого пов'язана поява однієї з подій, ніяким чином не залежить від появи інших подій. У такому випадку

$$p(A_1 \cap A_2 \cap \ldots \cap A_k) = p(A_1)p(A_2) \cdots p(Ak).$$

Якщо події не є незалежними, то правило множення може не виконуватися. Нехай $A$ – подія, яка означає, що з урни, в якій міститься $m$ чорних і $n - m$ білих кульок, дістається чорна кулька, а подія $B$ – дістається теж чорна кулька, після того, як з урни дістали одну кульку. Якщо перша кулька була чорною (відбулася подія $A$), то в урні залишилося $m - 1$ чорних кульок і $n - m$ білих. Тоді ймовірність $p(B) = \frac{m-1}{n-1}$. Якщо ж перша кулька була білою (відбулася подія $\bar{A}$), то $p(B) = \frac{m}{n-1}$. Як бачимо ймовірність події $B$ залежить від того відбулася, чи ні подія $A$. Якщо подія $A$ відбулася, то ймовірність події $B$ називається **умовною ймовірністю** події $B$ за умови появи події $A$ і позначається $p(B|A)$. Отже, в нашому прикладі $p(B|A) = \frac{m-1}{n-1}$.

Обчислення значення $p(B|A)$ виконується таким чином. Подія $A$ може наступити в $N = m(n-1)$ випадках (першою була витягнута чорна кулька, а далі одна з $n-1$ кульок). Подія $B$ наступає в $M = m(m-1)$ випадках (обидві кульки були чорними). Тоді подія, яка нас цікавить, може наступити в $\frac{m(m-1)}{m(n-1)} = \frac{m-1}{n-1}$ випадках. Нехай кількість рівноймовірних випадків появи подій $A$ і $B$ дорівнює $K$, а кількість появи і події $A$, і події $B$ дорівнює $M$. Тоді ймовірність події $A \cap B$ – $p(A \cap B)$ – дорівнює $\frac{M}{K}$. Але $\frac{M}{K} = \frac{N}{K} \cdot \frac{M}{N}$, де $N = m(n-1), M = m(m-1)$, і тоді $\frac{M}{N} = p(B|A)$ і $p(A) = \frac{N}{K}$. Отже, $p(A \cap B) = p(A) \cdot p(B|A)$, звідки дістаємо

$$p(B|A) = \frac{p(A \cap B)}{p(A)}.$$

**Властивості умовної ймовірності** випливають з означення цієї величини:

1) $0 \leq p(B|A) \leq 1$;

2) $p(B|A) = 1$, якщо $A \subseteq B$ і, зокрема, коли $B$ достовірна подія;

3) $p(B|A) = 0$, якщо події $A$ і $B$ несумісні і, зокрема, коли $B$ неможлива подія;

4) якщо $C \subset B$, то $p(B|A) \leq p(C|A)$;

5) якщо $B$ і $C$ несумісні події, то $p(B \cup C|A) = p(B|A) + p(C|A)$;

6) якщо $B_1, B_2, \ldots, B_k$ попарно несумісні події, то
$p(B_1 \cup B_2 \cup \cdots \cup B_k|A) = p(B_1|A) + p(B_2|A) + \cdots + p(B_k|A)$;

7) $p(\bar{B}|A) = 1 - p(B|A)$.

Тепер можна дати строге означення незалежних подій.

**Означення 36.** *Події $A$ і $B$ називаються незалежними, якщо* $p(B|A) = p(B)$.

Якщо розглянути задачу про те, скільки викликів надходить від абонентів на телефонну станцію протягом певного проміжку часу, то кількість цих викликів не є сталою величиною. Спостереження показують, що це число має значні коливання. Подібна ситуація спостерігається з кількістю викликів швидкої медичної допомоги і в багатьох інших задачах. Число значень, яких може набувати випадкова величина, може бути скінченним, зліченним або незліченним. Значення можуть розподілятися дискретно або заповнювати інтервал щільно. У криптографії, як правило, застосовується дискретна область, в якій набуває своїх значень випадкова величина. До дискретних областей відносять або скінченні, або нескінченні, але зліченні області. Для того, щоб задавати ймовірності значень випадкової величини, вводять поняття функції розподілу випадкової величини. Якщо $\xi$ – випадкова величина і $x$ – довільне дійсне число, то ймовірність того, що $\xi$ набуде значення $x$, називається функцією розподілу ймовірностей випадкової величини $\xi$. Нехай $O$ – деяка зліченна область.

**Означення 37.** *Функцією розподілу випадкової величини $\xi$ на дискретній області $O$ називається відображення $P : O \to \mathcal{D}$ таке,*

*що* $P[\xi = x_i] = p_i$ $(i = 1, 2, \ldots, |O|)$ *і для якого виконуються такі умови:*

*а)* $p_i \geq 0$; *б)* $\sum\limits_{i=1}^{|O|} p_i = 1$.

Таким чином, випадковою величиною називається величина, значення якої залежить від випадку і для якої визначена функція розподілу ймовірностей.

**Рівномірний розподіл** має вигляд

$$P[\xi = x_i] = \frac{1}{|O|}.$$

**Приклад 2.4.1**. а) Нехай $O = \{a, b, c, \ldots, x, y, \}$ – алфавіт англійської мови. Виберемо випадково довільний елемент $x$ із $O$, дотримуючись рівномірного розподілу. Тоді ймовірність того, що вибраним елементом буде літера $d$, буде $\frac{1}{|O|} = \frac{1}{26}$.

б) Нехай $O$ – множина невід'ємних чисел, які мають не більше $k$ бітів у двійковому зображенні. Виберемо з множини $O$ випадковий елемент, дотримуючись рівномірного розподілу. Покажемо, що ймовірність вибрати число, яке має $k$ бітів, дорівнює $\frac{1}{2}$.

Розіб'ємо множину $O$ на дві підмножини $O_1 = \{0, 1, 2, \ldots, 2^{k-1} - 1\}$ і $O_2 = \{2^{k-1}, 2^{k-1} + 1, \ldots, 2^k - 1\}$. Множина $O_2$ включає всі двійкові числа, які мають $k$ бітів у своєму двійковому зображенні. Оскільки $|O_1| = |O_2| = \frac{|O|}{2}$, то

$$P[x \in O_2] = P[\bigcup_{i=2^{k-1}}^{2^k-1} [x = i]] = \sum_{i=2^{k-1}}^{2^k-1} P[x = i]] = \frac{|O_2|}{|O|} = \frac{1}{2}. \spadesuit$$

**Біноміальний розподіл**. Нехай $\xi$ означає кількість появ події $A$ в послідовності $n$ незалежних дослідів, в кожному з яких її ймовірність стала і дорівнює $p$. Залежно від випадку $\xi$ може набувати всіх цілочисельних значень від 0 до $n$ включно. Яка ймовірність того, що подія $A$ появиться $m$ разів в $n$ дослідах, тобто, чому дорівнює $P[\xi = m]$? Знайти цю ймовірність можна за допомогою таких підрахунків: у послідовності з $n$ елементів нас цікавить $m$ елементів, які означають появу події $A$. Усіх таких випадків буде очевидно $C_n^m$, де $C_n^m$ – біноміальний коефіцієнт. Кожний випадок появи події $A$ має ймовірність $p$, а кожний випадок непояви $A$ (тобто появи події $\bar{A}$) має ймовірність $1 - p$. Отже,

$$P[\xi = m] = C_n^m p^m (1 - p)^{n-m}.$$

Якщо випадкова величина $\xi$ набуває значення з множини $O = \{0, 1, 2, \ldots, n\}$ і для величини $p \in (0, 1)$ виконується отримана вище залежність, то говорять, що випадкова величина $\xi$ має **біноміальний розподіл імовірностей**.

**Приклад 2.4.2**. Розглянемо такі задачі. Нехай під час виконання експерименту ідеальна монета підкидалася 10 разів. Яка ймовірність того, що

а) "орел" випаде п'ять разів?

б) "орел" випаде не більше п'яти разів?

*Розв'язання.* а) Застосовуючи функцію біноміального розподілу ймовірностей, знаходимо: $P[\xi = 5] = C_{10}^5(\frac{1}{2})^5(\frac{1}{2})^5 = C_{10}^5(\frac{1}{2})^{10} = \frac{252}{1024} \approx 0,246$.

б) У цій задачі необхідно знайти суму ймовірностей всіх подій, за яких "орел" випаде не більше п'яти разів. Отже, маємо $P[\xi \leq 5] = (\frac{1}{2})^{10} \sum\limits_{i=0}^{5} C_{10}^i = 0,623$. ♠

**Ланцюги Маркова**. Безпосереднім узагальненням схеми незалежних досліджень є схема ланцюгів Маркова. У цій схемі виконується послідовність експериментів, у кожному з яких може відбутися одна і тільки одна з $k$ несумісних подій $A_1^s, A_2^s, \ldots, A_k^s$ (верхній індекс означає номер експерименту).

**Означення 38.** *Говорять, що послідовність екпериментів утворює простий ланцюг Маркова, якщо умовна ймовірність в $(s+1)$-му експерименті $(s = 1, 2 \ldots)$ відбутися події $A_i^{s+1}$ $(i = 1, \ldots, k)$ залежить тільки від того, яка подія відбулася в $s$-му експерименті і не змінюється від додаткових відомостей про те, які події відбувалися в раніше проведених есперимнтах.*

Далі обмежимося розглядом *однорідних ланцюгів Маркова*, в яких умовна ймовірність події $A_j^{s+1}$ в $(s+1)$-му експерименті, не залежить від номера експерименту. Таку ймовірність називають ймовірністю переходу і позначають $p_{ij}$, де перший індекс означає результат попереднього експерименту, а другий індекс указує, в який стан перейде система в наступний момент часу.

**Означення 39.** *Однорідний ланцюг Маркова – це така послідовність випадкових величин $\{\xi_i\}, i = 1, 2 \ldots$, які набувають своїх значень у дискретній області $Z$, що для будь-якого $n \geq 2$*

$$P(\xi_n = x_n | \xi_1 = x_1, \ldots, \xi_{n-1} = x_{n-1}) = P(\xi_n = x_n | \xi_{n-1} = x_{n-1}) =$$

$$= p(x_n x_{n-1}).$$

(*Для неоднорідного марковського ланцюга ймовірності переходу* $p(x_n x_{n-1})$ *залежали б від* $n$*, а в даному випадку вони залежать лише від станів* $x_{n-1}x_n$).

Повна ймовірнісна ситуація всіх можливих змін, які виконуються під час переходу від одного експерименту до безпосередньо наступного, визначається матрицею

$$\pi_1 = \begin{pmatrix} p_{11} & p_{12} & \dots & p_{1k} \\ p_{21} & p_{22} & \dots & p_{2k} \\ \dots & \dots & \dots & \dots \\ p_{k1} & p_{k2} & \dots & p_{kk} \end{pmatrix},$$

яка складається з імовірностей переходів і яка називається *матрицею переходу.*

**Властивості елементів матриці переходу** зводяться до таких:

а) $0 \le p_{ij} \le 1$; б) $\sum\limits_{j=1}^{k} p_{ij} = 1$ $(i = 1, 2, \dots, k)$.

А для самих матриць переходу виконується таке правило. Якщо

$$\pi_n = \begin{pmatrix} P_{11}(n) & P_{12}(n) & \dots & P_{1k}(n) \\ P_{21}(n) & P_{22}(n) & \dots & P_{2k}(n) \\ \dots & \dots & \dots & \dots \\ P_{k1}(n) & P_{k2})n) & \dots & P_{kk}(n) \end{pmatrix}$$

– матриця переходу після виконання $n$-го експерименту, то $\pi_n = \pi_1^n$.

Розглянемо тепер застосування ймовірносних методів до аналізу властивостей природних мов.

## 2.4.2. Ентропія і інформація

Основною властивістю випадкових подій є відсутність впевненості в тому, що вони відбудуться, а це створює певну невизначеність експериментів, пов'язаних із цими подіями. Зрозуміло, що ступінь такої невизначеності в різних випадках буде різним. Але з практичної точки зору важливо вміти чисельно оцінювати ступінь невизначеності результатів різноманітних експериментів, щоб мати можливість порівнювати їх між собою.

Нехай маємо $k$ рівноймовірних результатів екпериментів. Очевидно, що ступінь невизначеності кожного результату визначається числом $k$: при $k = 1$ результат експерименту не є випадковим, але коли $k$ велике число, то передбачити результат важко. Отже, шукана числова характеристика ступеня невизначеності є функцією $f(k)$ від $k$. Для детальнішого визначення функції $f(k)$ необхідно вимагати від неї виконання певних додаткових умов. Розглянемо два незалежні експерименти $\alpha$ і $\beta$. Нехай експеримент $\alpha$ має $k$ рівноймовірних результатів, а експеримент $\beta$ – $l$ рівноймовірних результатів. Зрозуміло, що результат експерименту $\alpha\beta$ більш невизначений, ніж $\alpha$ і $\beta$ тому, що до невизначеності експерименту $\alpha$ додається невизначеність експерименту $\beta$. Логічно вважати, що ступінь невизначеності експерименту $\alpha\beta$ дорівнює сумі невизначеностей експериментів $\alpha$ і $\beta$, а оскільки експеримент може мати $kl$ рівноймовірних результатів, то приходимо до такої умови, якій повинна задовольняти функція $f(k)$:

$$f(kl) = f(k) + f(l).$$

Функцією, яка має такі властивості, є логарифм, оскільки $\log(kl) = \log k + \log l$, то це означає, що за міру невизначеності експерименту, який має $k$ рівноймовірних результатів, беруть число $\log k$. Таке означення узгоджується з тим, що при $k = 1$ міра невизначеності $\log k = 0$ і що при зростанні $k$ ця міра теж зростає.

Зазначимо, що вибір основи логарифмів несуттєвий, оскільки на підставі відомої формули $\log_b k = \log_a b \log_a k$ перехід від однієї основи до другої зводиться лише до множення функції $f(k) = \log k$ на константу (модуль переходу $\log_b a$). Тому далі буде використовуватися логарифм за основою 2. Це означає, що за одиницю виміру ступеня невизначеності експерименту, який має лише два рівноймовірні результати, беруть **біт**. Але це не обмежує загальності тому, що коли взяти за основу логарифма число 10 (десяткова одиниця або **дит**), то ступінь невизначеності буде приблизно в $\log 10 \approx 3\frac{1}{3}$ рази більшим двійкової одиниці.

Нехай маємо експеримент, який може давати $k$ рівноймовірних результатів $A_1, A_2, \ldots, A_k$ (тобто $p(A_i) = \frac{1}{k}$). Оскільки загальна невизначеність експерименту дорівнює $\log k$, то можна вважати, що

кожний окремий результат, який має ймовірність $\frac{1}{k}$, вносить невизначеність $\frac{1}{k}\log k = -\frac{1}{k}\log\frac{1}{k}$. Аналогічно до цього можна покласти, що в загальному випадку, для експерименту з таблицею ймовірностей

| експеримент | $A_1$ | $A_2$ | $\cdots$ | $A_k$ |
|---|---|---|---|---|
| імовірність | $p(A_1)$ | $p(A_2)$ | $\cdots$ | $p(A_k)$ |

міра невизначеності дорівнює

$$H(\alpha) = -p(A_1)\log p(A_1) - p(A_2)\log p(A_2) - \ldots - p(A_k)\log p(A_k).$$

**Означення 40.** *Число $H(\alpha)$ називається ентропією експерименту $\alpha$.*

**Властивості ентропії**.

а) $H(\alpha) \geq 0$, оскільки $p(A_i) \geq 0$ і $-p(A_i)\log p(A_i) \geq 0$;

б) $H(\alpha) = 0$ тоді, коли $p(A_i) = 1$ $(i = 1, 2, \ldots, k)$;

в) якщо $p(A_i)$ мале число, то і $H(\alpha)$ мале число;

г) максимальна ентропія $H(\alpha)$ досягається за рівномірного розподілу ймовірностей, тобто

$$H(\alpha) = -\tfrac{1}{k}\log\tfrac{1}{k} - \tfrac{1}{k}\log\tfrac{1}{k} - \ldots - \tfrac{1}{k}\log\tfrac{1}{k} = \log k.$$

**Умовна ентропія**. У розглянутому вище випадку вважалося, що експерименти незалежні. Але в дійсності це не завжди так. Якщо експерименти залежні, то вже не можна бути певним того, що ентропія експерименту $\alpha\beta$ дорівнюватиме сумі $H(\alpha) + H(\beta)$.

Припустимо, що експерименти $\alpha$ і $\beta$ не є незалежними. Розглянемо ентропію $H(\alpha\beta)$:

$$\begin{aligned} &H(\alpha\beta) = -p(A_1B_1)\log p(A_1B_1) - \ldots - p(A_1B_l)\log p(A_1B_l) - \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &-p(A_kB_1)\log p(A_kB_1) - \ldots - p(A_kB_l)\log p(A_kB_l), \end{aligned} \qquad (2.1*)$$

де $A_1, A_2, \ldots, A_k, B_1, B_2, \ldots, B_l$ означають можливі результати експериментів $\alpha$ і $\beta$. Тепер $p(A_1B_1) = p(A_1)p(B_1|A_1)$, де $p(B_1|A_1)$ – умовна ймовірність події $B_1$ за умови, що відбулася подія $A_1$. Отже,

$$H(\alpha\beta) = -p(A_1)p(B_1|A_1)(\log p(A_1) + \log p(B_1|A_1)) -$$
$$-p(A_1)p(B_2|A_1)(\log p(A_1) + \log p(B_2|A_1)) - \ldots -$$
$$-p(A_1)(p(B_l|A_1)(\log p(A_1) + \log p(B_l|A_1)) =$$
$$= -p(A_1)(p(B_1|A_1) + p(B_2|A_1) + \ldots + p(B_l|A_1)) \log p(A_1) +$$
$$+p(A_1)(-p(B_1|A_1) \log p(B_1|A_1) - p(B_2|A_1) \log p(B_2|A_1) - \ldots$$
$$\ldots - p(B_l|A_1) \log p(B_l|A_1)).$$

Але $p(B_1|A_1) + p(B_2|A_1) + \ldots + p(B_l|A_1) = 1$ (оскільки подія $B_1$, $B_2, \ldots, B_l$ достовірна) і на підставі того, що

$$-p(B_1|A_1) \log p(B_1|A_1) - \ldots - p(B_l|A_1) \log p(B_l|A_1) = H_{A_1}(\beta),$$

за умови, що відбулася подія $A_1$. Таким чином, перший рядок (2.1*) набуває вигляду: $-p(A_1) \log p(A_1) + p(A_1)H_{A_1}(\beta)$. Аналогічно отримуємо вирази для решти рядків із (2.1*). У результаті приходимо до виразу

$$H(\alpha\beta) = H(\alpha) + p(A_1)H_{A_1}(\beta) + p(A_2)H_{A_2}(\beta) + \ldots + p(A_k)H_{A_k}(\beta).$$

Покладаючи $H_\alpha(\beta) = p(A_1)H_{A_1}(\beta) + p(A_2)H_{A_2}(\beta) + \ldots + p(A_k)H_{A_k}(\beta)$, дістаємо

$$H(\alpha\beta) = H(\alpha) + H_\alpha(\beta).$$

Отриманий вираз називається **правилом додавання ентропій**.

**Властивості умовної ентропії**. Умовна ентропія має такі властивості:

1) $H_\alpha(\beta) \geq 0$;

2) $H(\alpha\beta) = H(\alpha) + H_\alpha(\beta) = H(\beta) + H_\beta(\alpha)$.

Інтуїтивно це означає, що невизначеність експериментів $\alpha$ і $\beta$ дорівнює невизначеності $\alpha$ плюс невизначеність $\beta$, після того, як результат експерименту $\alpha$ став відомим. Друга рівність у 2) справедлива на підставі того, що $\alpha$ і $\beta$ входять у формулу симетрично.

3) $H(\alpha) \geq H_\alpha(\beta) = H_\alpha(\beta\gamma)$.

Ця властивість означає, що додаткові знання не можуть збільшити невизначеність.

**Елементи теорії інформації**. Розглянемо спочатку моделі джерел відкритого тексту, які необхідні для математичного аналізу тексту та його криптографічних перетворень.

Під текстом будемо розуміти послідовність літер деякого скінченного алфавіту $X = \{x_1, x_2, \ldots, x_m\}$, де $x_i$ – літери або символи цього алфавіту. Елементами алфавіту можуть бути власне літери; літери та цифри; літери, цифри та знаки пунктуації, взагалі скінченний набір будь-яких символів. Як правило, ми будемо розглядати український, російський чи латинський алфавіти (малі літери) зі знаком пропуску, що вважається літерою, або без нього, або ж двійковий алфавіт, що складається з двох символів: 0 та 1.

Відкритий текст (ВТ) – це текст, що підлягає шифруванню, шифрований текст (ШТ) – це текст, що утворюється в результаті шифрування. Відкритий і шифрований тексти можуть бути записані як у одному й тому ж, так і в різних алфавітах.

Нагадаємо, що словом довжини $n$ ($n$-грамою) називається слово в алфавіті $X$, яке сладається з $n$ символів цього алфавіту. При $n = 2$ це біграма, при $n = 3$ – триграма.

Будь-який текст має певну статистичну структуру. Для опису цієї структури користуються різноманітними ймовірнісними моделями мови.

Нехай маємо деяке *джерело*, яке генерує *відкритий текст*. Це джерело генерує послідовність символів алфавіту $x_1, x_2, \ldots, x_n, \ldots$ випадковим чином. Воно визначається алфавітом та ймовірностями появи $n$-грам: $P(\xi_{i+1} = x_1, \xi_{i+2} = x_2, \ldots, \xi_{i+n} = x_n)$ для будь-яких цілих $n \geq 1, i \geq 0$ (тут $\xi_{i+1}, \xi_{i+2}, \ldots, \xi_{i+n}, \ldots$ – випадкові величини, а $x_1, x_2, \ldots, x_n$ – літери алфавіту $X$), які мають задовольняти умовам:

$$\sum_{x_1,\ldots,x_n \in X} P(\xi_{i+1} = x_1, \xi_{i+2} = x_2, \ldots, \xi_{i+n} = x_n) = 1,$$

а також для будь-якого цілого $s \geq 1$

$$\sum_{x_1,\ldots,x_n \in X} P(\xi_{i+1} = x_1, \ldots, \xi_{i+n} = x_n, \xi_{i+n+1} = x_{n+1}, \ldots, \xi_{i+n+s} = x_{n+s}) =$$
$$= P(\xi_{i+1} = x_1, \ldots, \xi_{i+n} = x_n).$$

*Джерело* називають *стаціонарним*, якщо для будь-яких цілих $n \geq 1, 1 \leq i_1 < \ldots < i_n, j \geq 0$, і будь-якого набору літер алфавіту $x_1, \ldots, x_n$ виконується рівність:

$$P(\xi_{i_1+j} = x_1, \ldots, \xi_{i_n+j} = x_n) = P(\xi_{i_1} = x_1, \ldots, \xi_{i_n} = x_n).$$

Далі будемо розглядати лише стаціонарні джерела, тобто такі,

у яких немає залежності від зсуву $j$. Для стаціонарних джерел достатньо задати ймовірності $P(\xi_1 = x_1, \xi_2 = x_2, \ldots, \xi_n = x_n)$ для $n \geq 1$.

Залежно від властивостей сумісних розподілів $P(\xi_1 = x_1,\ \xi_2 = = x_2, \ldots, \xi_n = x_n)$ для $n \geq 1$, можна побудувати різні моделі джерела ВТ.

Найбільш уживаними є описані нижче чотири моделі, з яких кожна наступна більш адекватно, ніж попередня, відображає структуру мови [9]. Перша з них є простою й менш за всі враховує реальні статистичні властивості мови. Цю модель називають М0.

**М0**. У цій моделі джерело у кожен момент часу генерує символи з $X$ незалежно та рівноймовірно:

$$P(\xi_i = x) = \frac{1}{m},\ i = 1, 2, \ldots, m, x \in X.$$

Усі $n$-грами в моделі М0 є рівноймовірними:

$$P(\xi_i = x_1, \ldots, \xi_n = x_n) = \frac{1}{m^n}$$

для будь-яких цілих $n \geq 1$ та $x_1, x_2, \ldots, x_n \in X$.

Модель М0 має допоміжний характер, оскільки вона не враховує навіть найпростіших властивостей мови. Наступна модель ВТ враховує частоти, з якими окремі літери алфавіту зустрічаються у мові.

**М1**. Символи тексту $x_1, x_2, \ldots, x_n \in X$ є незалежними, але вони генеруються з різними ймовірностями $P(\xi_i = x) = p(x), i = = 1, 2, \ldots, x \in X$, де розподіл $p(x)$ відомий.

Розподіл імовірностей $p(x)$ відповідає частотам появи літер $x \in \in X$ у мові. У цій моделі ймовірність появи $n$-грами має вигляд

$$P(\xi_i = x_1, \ldots, \xi_n = x_n) = \prod_{i=1}^{n} p(x_i).$$

Зазначимо, що ймовірності появи літер у природних мовах значно відрізняються. Наприклад, найбільшу частоту в українській і російській мовах має літера “о”, в англійській – літера “е”. У російській мові літера “о” зустрічається майже в 50 разів частіше літери “ф”, яка має найменшу частоту.

Наступна модель враховує мовну залежність між двома літерами, що стоять поряд.

**М2**. Джерело генерує біграми $x_1x_2, x_3x_4, x_5x_6\ldots$ незалежно одну від одної. Тобто на множині всіх біграм задано розподіл імовірностей $p(x_i, x_j)$ $i, j = 1, \ldots, m$, і кожна нова біграма джерела генерується незалежно від інших. У цій моделі ймовірність появи біграм має вигляд: $p(x_1x_2)p(x_3x_4)\ldots p(x_{n-1}x_n)$.

Складніші залежності мови враховують за допомогою марковської моделі.

**М3**. У цій моделі ВТ $x_1, x_2, \ldots$ утворює стаціонарний ланцюг Маркова. Для задання такого ланцюга достатньо задати розподіл початкових станів $p_0(x_i)$, $x_i \in X$, та перехідні ймовірності $p(x_ix_j) = = P(\xi_{n+1} = x_j|\xi_n = x_i), x_i, x_j \in X$, які на підставі однорідності не залежать від $n$.

За накладання деяких умов на ланцюг Маркова (які не суперечать властивостям природних мов) існує граничний розподіл $\pi_{x_i} = \lim\limits_{n\to\infty} P(\xi_n = x_i|\xi_{n-1} = x_j)$, що не залежить від початкового стану $p_0(x_j)$. Він називається стаціонарним розподілом імовірностей марковського ланцюга. У цій моделі ймовірність появи послідовності $x_1x_2\ldots x_n$ має вигляд $p(x_1)p(x_2|x_1)\ldots p(x_n|x_{n-1})$. Граничні розподіли ймовірності задовольняють таким рівностям:

$$\begin{cases} \sum\limits_{x_i\in X} \pi_{x_i} = 1; \\ \sum\limits_{x_i\in X} \pi_{x_i} p(x_i, x_j) = \pi_{x_j}, \ x_j \in X. \end{cases}$$

Імовірність $n$-грами у моделі М3 у стаціонарному режимі можна записати у вигляді

$$P(\xi_1 = x_1, \ldots, \xi_n = x_n) = \pi_{x_1} p(x_1, x_2)p(x_2, x_3)\cdots p(x_{n-1}x_n).$$

Нехай $M = \{q_1, q_2, \ldots, q_m\}$ – скінченна множина, на якій задано розподіл імовірностей $P = \{p(q_1), p(q_2), \ldots, p(q_m)\}$. Пара $(M, P)$ в теорії інформації називається **скінченним ансамблем**. Елементи $q_i \in M$ будемо називати **повідомленнями**. Часто говорять просто про ансамбль $M$, розуміючи під цим пару $(M, P)$.

Інтуїтивно зрозуміло, що малоймовірне повідомлення несе в собі більше інформації, ніж більш імовірне. К. Шеннон запропонував для виміру кількості інформації застосовувати функцію, яка відповідає цьому інтуїтивному уявленню, і яка зручна для обчислень.

**Означення 41.** *Власною інформацією повідомлення $q_i$ називається величина $I(q_i) = -\log p(q_i) \geq 0$, а ентропією ансамблю $(M, P)$ за всіма повідомленнями – величина*

$$H(M) = -\sum_{i=1}^{m} p(q_i) \log p(q_i).$$

Як зазначалося вище, величина $H(M)$ інтерпретується як невизначеність експерименту, в якому з ансаблю $M$ вибирається одне повідомлення $q_i$, імовірність якого $p(q_i), i = 1, 2, \ldots, m$.

Розглянемо декартів добуток скінченних множин $M$ та $M_1$, тобто множину пар $(q, r)$, де $q \in M, r \in M_1$. Нехай на множині пар задано розподіл імовірностей $p(q, r)$. Тоді говорять, що ансамблі $M$ та $M_1$ задані сукупно. Розподіл $p(q, r)$ індукує розподіли на $M$ та $M_1$:

$$p(q) = \sum_{r_j \in M_1} p(q, r_j) \text{ і } p(r) = \sum_{q_i \in M} p(q_i, r).$$

Ці розподіли дають можливість розглядати $M$ і $M_1$ як окремі ансамблі.

**Означення 42.** *Сукупною ентропією ансамблів $M$ і $M_1$ називається величина $H(MM_1) = \sum_{q,r} p(q, r) \log p(q, r)$.*

*Сукупно задані ансамблі $M$ і $M_1$ називаються незалежними, якщо $\forall(q, r)(p(q, r) = p(q)p(r))$.*

Звідси випливає, що коли $M$ і $M_1$ – незалежні, то $H(MM_1) = = H(M) + H(M_1)$.

Нехай відомий результат $k$-го експерименту $M$. Тоді умовна ентропія буде така:

$$H_M(r) = -\sum_{q \in M} p(r|q) \log p(r|q),$$

а умовна ентропія ансамблю $M_1$ відносно ансамблю $M$

$$H_M(M_1) = -\sum_{q\in M} p(q) \sum_{r\in M_1} p(r|q)\log p(r|q) = \\ = -\sum_{q\in M, r\in M_1} p(q,r)\log p(q,r).$$

**Означення 43.** *Взаємною інформацією ансамблів $M$ та $M_1$ називається величина $I(M;M_1) = H(M) - H_M(M_1)$. При незалежних ансамблях $M$ та $M_1$ $I(M;M_1)) = 0$.*

**Ентропія на символ джерела**. Якщо $X$ – алфавіт, то $n$-грама $(x_1,\dots,x_n) \in X^n$ і $n$ ансамблів задані сукупно розподілом $n$-грам $P(\xi_1 = x_1,\dots,\xi_n = x_n)$ (джерело стаціонарне: немає залежності від розташування $n$-грами в тексті). Ентропія $n$-грами

$$H(X^n) = -\sum_{x_j\in X} p(x_1,x_2,\dots x_n)\log p(x_1,x_2,\dots,x_n). \tag{2.1}$$

Середня ентропія на один символ $n$-грами буде дорівнювати $H_n = \frac{H(X^n)}{n}$. Для стаціонарних джерел існує границя $H_\infty$ цієї величини:

$$\lim_{n\to\infty} \frac{H(X^n)}{n} = H_\infty,$$

і ця границя називається **ентропією на символ джерела**.

Розглянемо, чому дорівнюватиме ентропія на символ джерела для різних моделей ВТ, які були введені раніше.

**М0**. $H(X^n) = nH(X) = n\log m$ (третя та четверта властивості ентропії).

**М1**. Унаслідок незалежності літер у тексті
$H_\infty = \lim\limits_{\infty} \frac{H(X^n)}{n} = \frac{nH(X)}{n} = H(X) = -\sum\limits_{x\in X} p(x)\log p(x)$.

**М2**. Розглядаються тексти довжиною $2n$:
$H_\infty = \lim\limits_{\infty} \frac{H(X^{2n})}{2n} = \frac{nH(X^2)}{2n} = \frac{H(X^2)}{2} = H_2$ (оскільки біграми незалежні).

**М3**. Можна довести, що для джерела, яке описується однорідним ланцюгом Маркова зі стаціонарним розподілом $\pi_{x_i}$ та ймовірностями переходу $p(x_ix_j)$, $x_i, x_j \in X$,

$$H_\infty = -\sum_{x_i,x_j\in X} \pi_{x_i} \log p(x_i, x_j).$$

Величина $H_n$ є $n$-м наближенням до $H_\infty$. Зазначимо, що перші наближення $H_1, H_2, H_3$ ще дуже відрізняються від $H_\infty$ (див. наведену нижче таблицю). У той же час обчислити $H_n$ для великих значень $n$ практично неможливо через величезну кількість можливих $n$-грам. Але можна розглянути умовну ентропію $n$-го символу тексту при умові, що відомі результати $n-1$ попередніх експериментів, тоді $H_n = H(x_n|x_1, x_2, \ldots, x_{n-1})$. У теорії інформації доводиться, що послідовність $H_n$ має границю при $n \to \infty$ і ця границя збігається з границею послідовності $H_n$. Отже,

$$H_\infty = \lim_{n\to\infty} H(x_n|x_1 \ldots, x_{n-1}).$$

Ця рівність служить **другим означенням ентропії на символ джерела**.

| Мова | $H_0$ | $H_1$ | $H_2$ | $H_3$ | $H_5$ | $H_8$ |
|---|---|---|---|---|---|---|
| Англійська | 4.76 | 4.03 | 3.32 | 3.10 | 2.1 | 1.9 |
| Російська | 5 | 4.35 | 3.52 | 3.01 | – | – |

Друге означення ентропії на символ джерела використовують для експериментального оцінювання $H_\infty$ шляхом вгадування людиною наступної літери тексту.

Використовуючи друге означення $H_\infty$, А. Н. Колмогоров експериментально оцінив для російської мови $H_n$ для великих значень $n$. Виявилося, що після $H_{30}$ значення $H_n$ вже практично не змінюються, тобто дорівнюють $H_\infty$, у той час як $H_{15}$ ще істотно відрізняється від $H_\infty$. Аналогічні результати були отримані і для інших європейських мов (роботи К. Шеннона та ін.).

**Означення 44.** *Надлишковістю на символ джерела називається величина* $R = 1 - \frac{H_\infty}{H_0}$, *де* $H_0 = \log|X| = \log m$.

$H_0$ дорівнює максимальній кількості інформації, яку може нести в собі один символ джерела, а $H_\infty$ – кількість інформації, яку насправді несе в собі один символ. Таким чином, для достатньо великих $n$ величина $nR$ є середньою кількістю "зайвих" символів у

тексті довжиною $n$, після втрати яких теоретично можна відновити текст. Надлишковість європейських мов знаходиться десь на рівні 60-80 %. Але це не означає, що після випадкового видалення 60 % символів (літер) завжди залишиться можливість відновити текст. Відкидати літери потрібно вибірково, використовуючи всі закономірності мови, і відновлювати також, застосовуючи всі ці закономірності. Але практично врахувати всі закономірності неможливо. Експериментально встановлено: тільки до 25 % літер можна видалити випадковим чином, щоб розглянутий текст залишився придатним для відновлення.

Якби літери в тексті були незалежними та рівномірно розподіленими, то $H_\infty$ дорівнювало б $H_0$, а надлишковість $R$ була б нульовою. Але тоді будь-яка послідовність літер була б змістовим текстом, і навіть найменша помилка при передачі повідомлень призводила б до іншого змістового тексту й не могла б бути визначена. В усному мовленні ми "ковтаємо" частину звуків, або виголошуємо їх нечітко, на письмі інколи робимо орфографічні помилки, та завдяки надлишковості мови все одно розуміємо один одного. Тож надлишковість – це природний механізм, що сприяє розумінню та протидії помилкам.

### 2.4.3. Цілком таємна криптосистема за Шенноном

Розглянемо основні положення, закладені К. Шенноном, відносно криптографічних систем. Ці положення були сформульовані ще в 40-х р. минулого століття і деякі з них уже застаріли, але деякі не втратили свого значення і нині.

Нехай $m, m_1$ – повідомлення, тобто слова в деякому алфавіті; $c, c_1$ – криптограми, що теж, як правило, є словами в деякому алфавіті; $k \in K$ – ключ; $E_k, D_k$ – конкретні перетворення (алгоритми) шифрування і розшифрування відповідно з ключем $k$.

У загальному випадку шифр – це відображення $f : M \to C$, де $(\forall k \in K, \forall m \in M)\ D_k(E_k(m)) = m$. Відображення $f$ має бути ін'єктивним, що дає можливість зашифрувати будь-який ВТ за допомогою ключа $k$.

У теорії Шеннона сформульовано такі припущення:

1. Криптоаналітику відомий тільки шифрований текст, тобто атака здійснюється на основі шифротексту.

2. Ключ і рандомізатор (засіб зрівнювання частотних характеристик тексту) використовують для шифрування тільки один раз (тобто криптоаналіз здійснюється тільки по одній криптограмі).

3. На декартовому добутку задано ймовірнісний розподіл.

Деякі припущення Шеннон використовував неявно, зокрема, такі: канал передачі без спотворень, перетворення інформації без помилок, відсутній зворотний зв'язок. Хоча вже розроблені більш загальні теорії, але результати Шеннона заклали наріжний камінь криптографії як науки.

Крім того, припускається, що всі простори криптосистеми $\mathcal{S}$ та розподіл імовірностей на $M \times K$ відомі криптоаналітику.

Нехай $p(m,k)$ – розподіл імовірностей на декартовому добутку $M \times K$, де $\sum\limits_{m,k} p(m,k) = 1$. Зокрема, якщо відкритий текст і ключ, що генеруються, незалежні (а найчастіше саме так на практиці і відбувається), то $p(m,k) = p(m)p(k)$. Розподіл $p(m,k)$ індукує розподіли ймовірностей на інших множинах системи. На множині алгоритмів шифрування $E$, алгоритмів розшифрування $D$ та множині криптограм $C$ розподіли задають формулами:

$$(\forall k \in K)p(E_k) = p(D_k) = p(k);$$
$$(\forall c \in C)p(c) = \sum_{\forall (m,k), E_k(m)=c} p(m,k),$$

де останнє підсумовування здійснюється за всіма парами $(M,k)$, для яких $E_k(m) = c$.

Сумісний розподіл імовірностей криптограм і ключів на $C \times K$ задається формулою

$$(\forall c,k)p(c,k) = \sum_{\forall m, E_k(m)=c} p(m,k).$$

Сумісний розподіл імовірностей криптограм і відкритих текстів на $C \times M$ індукується такими співвідношеннями:

$$(\forall c,m)p(c,m) = p(m,c) = \sum_{\forall k, E_k(m)=c} p(m,k).$$

Імовірнісні розподіли на вказаних декартових добутках дають можливість знайти умовні розподіли:

$$p(m|c) = \frac{p(m,c)}{p(c)},\ p(c|m) = \frac{p(m,c)}{p(m)},\ p(k|c) = \frac{p(k,c)}{p(c)}.$$

Після побудови вищезазначених розподілів можна розглядати також ентропію множин з імовірнісною мірою і побудувати математичну модель криптосистеми. Поняття цілком таємної криптосистеми формалізує властивість теоретичної стійкості у теоретико-інформаційному розумінні.

**Означення 45.** *Цілком таємною криптосистемою $\mathcal{S}$ називається криптосистема, для якої виконується одна з умов:*

*1.* $(\forall(m,c))\ p(m|c) = p(m)$*;*

*2.* $H(m|c) = H(m)$*, де* $H(m)$ *і* $H(m|c)$ *– ентропія і умовна ентропія відповідно.*

Умови 1 і 2 еквівалентні. Умова 1 означає, що ВТ і ШТ статистично незалежні. Умову 2 можна інтерпретувати як відсутність у ШТ інформації відносно ВТ, тобто взаємна інформація $I(m,C) = H(m) - H(m|c) = 0$.

**Теорема 14.** *Для того, щоб криптографічна система була цілком таємною, необхідно і достатньо щоб виконувалась одна з умов:*

*1.* $(\forall(m,c))\ p(c|m) = p(c)$*;*

*2.* $H(c|m) = H(c)$*.*

*Доведення.* Необхідність. Якщо система цілком таємна, то з $p(m) = p(m|c) = \frac{p(m,c)}{p(c)}$ випливає $\frac{p(m,c)}{p(m)} = p(c) = p(c|m)$.

Достатність доводиться аналогічно. ■

Користуючись цим критерієм, Шеннон показав, що цілком таємні криптосистеми існують і такою системою є нижченаведений шифр Вернама.

***Шифр Вернама.*** У цьому шифрі способом оборони перед атаками криптоаналізу є вибір ключа такої самої довжини, як і довжина відкритого тексту. Ключ генерується як випадкова послідовність $n$ незалежних рівноймовірних випадкових бітів з імовірністю

$2^{-n}$ незалежно від ВТ. Таку систему запропонував інженер Гільберт Вернам в 1918 р. Цей шифр найкраще служить для шифрування бінарних даних і описується таким чином:

$$c_i = p_i \oplus k_i,$$

де

$p_i$ – $i$-та бінарна цифра відкритого тексту;

$k_i$ – $i$-та бінарна цифра ключа;

$c_i$ – $i$-та бінарна цифра криптограми;

$\oplus$ – операція симетричної різниці (XOR – операція додавання за модулем 2).

Криптограма генерується за допомогою операції XOR і відкритого тексту. У зв'язку з властивостями операції XOR, розшифрування ґрунтується на тій самій операції: $p_i = c_i \oplus k_i$.

Цей шифр використовується дотепер на окремих важливих напрямках зв'язку. Головним недоліком шифру Вернама є велика довжина ключа, який потрібно попередньо передавати закритим каналом, і цей ключ у кожному сеансі зв'язку повинен генеруватися заново (*одноразовий ключ*). Дійсно, у шифрі Вернама ВТ кодується двійковим словом в алфавіті $X = \{0, 1\}$, ключ та ШТ також є словами в цьому ж алфавіті і мають однакові довжини. ШТ знаходиться з ВТ і ключа операцією XOR. Як ключ беруть "ідеальну" випадкову послідовність незалежних рівноймовірних випадкових бітів, тобто кожна реалізація довжини $n$ з'являється з імовірністю $2^{-n}$ незалежно від ВТ. Наприклад, якщо шифрується слово $m = 011011011100101$, то ймовірність появи будь-якої ключової послідовності, а також будь-якої криптограми має ймовірність $2^{-n}$.

**Теорема 15.** *Шифр Вернама – цілком таємна криптосистема.*

*Доведення.* Маємо відкритий текст: $m = m_1 m_2 \dots m_n$, ключ $k = k_1 k_2 \dots k_n$, криптограму $c = c_1 c_2 \dots c_n$, $m_i, k_i, c_i \in \{0, 1\}$, $c = m_1 \oplus k_1, \dots, m_n \oplus k_n$. На підставі теореми 14, безпосередньо знаходимо, що ймовірність появи будь-якої криптограми

$$(\forall c) p(c) = \sum_{\forall (m,k), E_k(m)=c} p(m,k) = 2^{-n} \sum_m p(m) = 2^{-n}$$

тому, що $p(m,k) = p(m)p(k) = 2^{-n}p(m)$, а умовна ймовірність

$$p(c|m) = \frac{p(m,c)}{p(m)},\ p(c,m) = p(m,c) = \sum_{(\forall k), E_k(m)=c} p(m,k).$$

Звідки $p(c|m) = \frac{p(m)2^{-n}}{p(m)} = 2^{-n}$. Отже, на підставі теореми 14, шифр Вернама – цілком таємна криптосистема. ■

Шифр Вернама з одноразовим ключем називається **одноразовим блокнотом** і застосовуєься і в наш час на окремих важливих напрямках зв'язку. За допомогою побудованої теорії Шеннон довів, що цей шифр є цілком таємним, тобто, маючи тільки криптограму, неможливо відтворити не тільки відкритий текст, а і будь-яку інформацію щодо нього. Головним недоліком шифру Вернама, як зазначалося у його розгляді, є велика довжина ключа, який потрібно попередньо передавати закритим каналом.

**Теорема 16.** *Якщо криптосистема цілком таємна, то* $H(m) \leq H(k)$ (*межа Шеннона*).

*Доведення.* Оскільки система цілком таємна, то

$$H(m) = H(m|c) \leq H(m,k|c) =$$
$$= H(k|c) + H(m|k,c) = H(k|c) \leq H(k).$$

Тут $H(m|k,c) = 0$ тому, що при відомих ШТ і ключах ВТ однозначно відновлюється. Таким чином, $H(m) \leq H(k)$. ■

Зазначимо, що межа Шеннона є лише необхідною, але не достатньою умовою цілковитої таємності криптосистеми.

Якщо ключі вибирають рівноймовірно, то ентропія ключів максимальна і дорівнює довжині ключа в бітах (даного двійковою послідовністю). Для розумного тексту природною мовою, заданого у двійковому алфавіті, ентропія дорівнює приблизно чверті його довжини в бітах. Якби надлишковість дорівнювала нулю, ентропія такого тексту збігалася б із кількістю двійкових символів у ньому. З вищенаведених теорем випливає, що у цілком таємної системи при шифруванні тексту великого розміру таємний ключ теж повинен бути великого розміру і при зростанні довжини ВТ довжина ключа

має зростати пропорційно. Таким чином, криптосистема з фіксованим ключем у загальному випадку не може бути цілком таємною. Для цілковитої таємності секретні ключі для більшості практичних застосувань мають бути занадто великими. Тому майже всі криптосистеми, що використовуються на практиці, загалом не є цілком таємними. Для таких криптосистем характеристикою їхньої надійності є практична стійкість. Для не цілком таємних систем Шеннон розглянув питання теоретико-інформаційної ненадійності, а також необхідного обсягу криптограми для зламування її шифру.

Якщо шифр забезпечує незалежність між розподілом початкових і зашифрованих текстів, то такий шифр називають стійким в **інформаційно-теоретичному сенсі**. На відміну від стійкості в сенсі обчислювальної стійкості, яка розглядалася вище, стійкість в інформаційно-теоретичному сенсі є **безумовною** і не піддається жодному з методів криптоаналізу. А протоколи, які стійкі в інформаційно-теоретичному сенсі, називаються **протоколами з нульовим розголошенням**.

Розглядаючи поняття інформаційно-теоретичної стійкості, Шеннон сформулював дві умови такої стійкості для класичних шифрів:

1. $|K| \geq |M|$, тобто довжина ключа має бути не менша довжини повідомлення;

2. $k \in K$ і використовується лише один раз для шифрування.

### 2.4.4. Частотна характеристика символів мови

Як зазначалося вище (модель **M1**), важливе місце в характеристиці природної мови займає частота появи літер в її словах. Не всі літери алфавіту тієї, чи іншої природної мови появляються в словах з однаковою частотою. Одна літера появляється частіше за інші, а друга літера зрідка появляється в тексті повідомлення. Аналогічна ситуація справедлива і для частоти появи двознаків, тризнаків і т. д. Наприклад, серед двознаків в англійській мові частіше за інші зустрічаються

$$th, an, to, it, do, en.$$

Ця обставина досить важлива і відіграє велику роль у криптоаналізі. Якщо текст зашифровано у відомому алфавіті, то частота появи символів у зашифрованому тексті несе певну інформацію про відкритий текст. Маючи в розпорядженні частоту появи символів алфавіту даної мови, криптоаналітик має можливість відтворити відкритий текст.

Частоту появи літер англійської мови наведено в табл. 2.4.1.

**Таблиця 2.4.1.** Частотна характеристика літер алфавіту англійської мови.

| Символ | Очікувана | Обчислена | Символ | Очікувана | Обчислена |
|---|---|---|---|---|---|
| A | 7,9 | 7,5 | N | 7,0 | 7,0 |
| B | 1,5 | 1,4 | O | 7,4 | 7,5 |
| C | 3,0 | 4,1 | P | 3,0 | 3,0 |
| D | 4,0 | 3,2 | Q | 0,2 | 0,2 |
| E | 12,8 | 12,6 | R | 6,5 | 6,7 |
| F | 2,0 | 2,3 | S | 6,0 | 7,3 |
| G | 1,5 | 1,9 | T | 8,9 | 9,2 |
| H | 6,0 | 3,8 | U | 3,0 | 2,8 |
| I | 6,5 | 7,7 | V | 1,0 | 1,0 |
| J | 0,5 | 0,2 | W | 1,5 | 1,4 |
| K | 0,4 | 0,4 | X | 0,5 | 0,3 |
| L | 3,7 | 3,8 | Y | 2,0 | 1,6 |
| M | 3,0 | 3,0 | Z | 0,2 | 0,1 |

**Приклад застосування частотного криптоаналізу**. Нехай зашифрований текст в алфавіті англійської мови має такий вигляд [40]:

UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZVUEPHZ
HMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSXEPYEPOPDZSZUFPOMBZ
WPFUPZHMDJUDTMOHMQ

На першому етапі можна обчислити відносну частоту появи літер алфавіту англійської мови на підставі табл. 2.4.1.

Якщо повідомлення має велику довжину, то застосування частотного аналізу є ефективним методом для криптоаналітика.

Але оскільки маємо в розпорядженні текст невеликої довжини, то не може бути впевненості в точному розв'язанні задачі криптоаналізу. Відносна частота появи літер у вказаній шифрограмі (у відсотках) наведена у табл. 2.4.2.

**Таблиця 2.4.2.** Відносна частота появи літер у шифрограмі

| Літера | Частота | Літера | Частота | Літера | Частота |
|---|---|---|---|---|---|
| P | 13,13 | E | 5,00 | B | 1,67 |
| Z | 11,67 | V | 4,17 | G | 1,67 |
| S | 8,33 | X | 4,17 | Y | 1,67 |
| U | 8,33 | F | 3,33 | I | 0,83 |
| O | 7,50 | W | 3,33 | J | 0,83 |
| M | 6,67 | Q | 2,50 | C | 0,00 |
| H | 5,83 | T | 2,50 | K | 0,00 |
| D | 5,00 | A | 1,67 | L,N,R | 0,00 |

Порівнюючи ці дані з даними табл. 2.4.1, правдоподібним видається, що літери P і Z в шифрограмі відповідають літерам $e$ і $t$ в явному тексті, але у нас не має впевненості в тому, яка літера відповідає якій. Літери S, U, O, M і H виступають відносно часто і правдоподібною є їхня відповідність літерам явного тексту із множини $\{r, n, i, o, a, s\}$.

Літери з найменшою частотою, тобто A,B,G,Y,I,L, правдоподібно належать до підмножини літер $\{w, v, b, k, x, q, j, \}$. У цьому місці маємо декілька можливих шляхів відтворення відкритого тексту.

Можемо продовжувати розпізнавати літери відкритого тексту, щоб вияснити, чи стає він деяким сенсовим текстом оригінального повідомлення. Більш систематичним методом є пошук чергових регулярностей. Наприклад, можливо певні слова знайдуться в тексті. Але можна шукати послідовності літер у шифрограмі, які повторюються, і намагатися здогадатися про їхні відповідники у відкритому тексті.

Ефективним способом аналізу є частота появи дволітерових комбінацій. Можна побудувати таблицю, подібну до табл. 2.4.1, яка включає частоту появи в текстах дволітерових комбінацій. Найчастіше в англомовних текстах появляється комбінація *th*. У тексті нашої шифрограми найчастіше виступає комбінація ZW, яка появляється 3 рази. Тоді припускаємо, що літері Z відповідає літера $t$, а літері W – літера $h$. Прямуючи далі таким шляхом, припускаємо, що літері P відповідає літера $e$. У шифрограмі виступає комбінація літер ZWP, яку можемо перекласти як *the*. Ця комбінація найчастіше появляється в англомовних текстах, і це показує, що ми прямуємо правильним шляхом.

Звернемо увагу на комбінацію літер WS у першому рядку. Не відомо, чи ці літери складають одне слово, чи ні. Якщо це так, то воно має вигляд (наприклад that), а звідси випливає, що літері S відповідає літера $a$. Отже, маємо

```
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZVUEPHZ
 t a       e  e te  a that e e a      a   t   e t
HMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSXEPYEPOPDZSZUFPOMBZ
   ta t ha e ee  a e  th    t  a  e  e e tat  e   t
WPFUPZHMDJUDTMOHMQ
he  et
```

Зідентифікували ми тільки чотири літери, але маємо в розпорядженні не такий малий фрагмент відтвореного тексту. Продовження аналізу в такому напрямку шляхом проб і помилок приводить до повного відтворення відкритого тексту шифрограми. Повний відкритий текст, із доданими пропусками між словами, є таким:

it was disclosed yesterday that several informal but direct
contacts have been made with political representatives of
the viet cong in moscow. ♠

**Контрольні запитання**

1. Що таке порядок зростання функції? Які вам відомі оцінки порядку зростання функцій? Навести означення ін'єктивної, сюр'єктивної та бієктивної функцій.

2. Які вам відомі класи часової складності алгоритмів?

3. Яка функція називається односторонньою та односторонньою із секретом?

4. Упорядкувати функції за порядком зростання:

а) $2^n$, $\log_2 \log_2 n$, $n^2 + \log_2 n$, $\log_2 n$,

б) $n - n^2 + 6n^3$, $2^{n-1}$, $n^2$, $n^3$, $n \log_2 n$,

в) $\sqrt{n}$, $8$, $n$, $n!$, $(3/2)^n$, $(\log_2 n)^2$.

5. Для кожної пари нижченаведених функцій $f$ і $g$ виконується тільки одна з рівностей $f = O(g)$ або $g = O(f)$. Визначити, який із випадків має місце.

а) $f(n) = (n^2 - n)/2$, $g(n) = 6n$; б) $f(n) = n + 2\sqrt{n}$, $g(n) = n^2$;

в) $f(n) = n + n \log_2 n$, $g(n) = n\sqrt{n}$; г) $f(n) = n^2 + 3n + 4$, $g(n) = n^3$;

д) $f(n) = n \log_2 n$, $g(n) = n\sqrt{n}/2$; е) $f(n) = 2 \log_2 n^2$, $g(n) = \log_2 n + 1$.

6. Навести приклади односторонніх функцій і функцій із класів $NP$ і $NPC$.

7. Що таке джерело відкритого тексту?

8. Чому розглядаються стаціонарні джерела і що це означає?

9. Які вам відомі моделі відкритого тексту?

10. Дайте визначення ентропії ансамблю.

11. Які вам відомі означення ентропії на символ джерела?

12. Скільки доданків у правій частині формули (2.1)?

13. Навести означення кількості інформації, інформаційного вмісту мови та ентропії.

14. Що таке надлишковість на символ джерела? Чому приблизно дорівнює надлишковість європейських мов?

15. Чим відрізняються процедури зашифрування та розшифрування?

16. Як визначається поняття теоретичної та практичної стійкості криптосистеми?

17. Які припущення зробив Шеннон у запропонованій ним моделі випадкового шифру?

18. Навести означення цілком таємної системи в теорії Шеннона.

19. Що таке границя Шеннона?

20. Чи виконання нерівності Шеннона є достатньою умовою для цілковитої таємності?

21. Чи існують цілком таємні криптосистеми?

## 2.5. Елементи теорії чисел

Майже у всіх криптографічних системах існує потреба у простих числах, або взаємно простих числах, або потреба у знаходженні розв'язків порівнянь за заданим модулем. Отже, основне завдання цього підрозділу – це розгляд указаних і супутних їм питань.

### 2.5.1. Основні означення

Уведемо формальні означення основних понять.

**Означення 46.** *Нехай $m, n \in \mathcal{Z}$. Число $m$ називається дільником числа $n$ або кажуть, що число $n$ кратне $m$ (позначення $m|n$), якщо існує таке число $k \in \mathcal{Z}$, що $n = m \cdot k$. Зокрема, число $n$ називається парним, якщо $n = 2k$.*

Із цього означення випливають такі прості наслідки.

а) Якщо $n \neq 0$ і $n = m \cdot k$, то $m \leq n$ і $k \leq n$.

б) Якщо $n = 0$, то 0 кратний довільному $m \in \mathcal{Z}$, $m \neq 0$, оскільки $0 = m \cdot 0$.

в) Оскільки операція множення у множині цілих чисел комутативна, то коли $m$ дільник $n$, то $k$ теж є дільником $n$.

г) Довільне число $n \in \mathcal{Z}, n \neq 0$ ділиться на $n$ і на 1, оскільки $n = n \cdot 1$. Ці два дільники називаються *тривіальними*. Зокрема, число $n$ називається **простим**, якщо воно має лише тривіальні дільники. Ціле число називається **складеним**, якщо воно має нетривіальні дільники.

Звідси випливає, що число 1 не належить ні до простих, ні до складених чисел. Отже, кожне ціле число або просте, або складене, або дорівнює одиниці.

д) Якщо $m, n \in \mathcal{N}$ і $m|n$ та $n|m$, то $m = n$. Дійсно, з умови а) випливає $m \leq n$ і $n \leq m$ і тому $m = n$.

є) Якщо $m, n \in \mathcal{Z}$ і $m|n$, то $\pm m|\pm n$.

Властивість є) дає змогу в питаннях подільності обмежуватися лише додатними числами, оскільки дільники $m$ і $-m$ відіграють одну й ту саму роль і такі дільники називають *асоційованими*. Тому далі розглядатимуться лише додатні дільники.

ж) Якщо $m, n, k \in \mathcal{Z}$ і $m|n$ та $k|m$, то $k|n$. Справді, з умови випливає, що $n = m \cdot l$ і $m = k \cdot s$, де $l, s \in \mathcal{Z}$. Але тоді $n = m \cdot l = k(s \cdot l)$, що означає $k|n$.

к) Якщо $m, n, k \in \mathcal{Z}$ і $m, n$ кратні $k$, то $k|m \pm n$. Дійсно, з умови випливає, що $m = k \cdot l$ і $n = k \cdot s$. Але тоді $m \pm n = k(l \pm s)$.

л) Якщо $k|m \pm n$ і $k|n$, то $k|m$. Дійсно, з умови випливає $m \pm n = k \cdot l$ і $m = k \cdot s$. Але тоді $n = k \cdot l \mp k \cdot s = k(l \mp s)$, що означає $k|n$.

Підсумовуючи, зазначимо, що відношення подільності на множині $\mathcal{N}$ є відношенням часткового порядку, а на множині $\mathcal{Z}$ – відношенням квазіпорядку. Це випливає з д) і є) відповідно.

**Теорема 17 (про ділення з остачею).** *Якщо $m, n \in \mathcal{Z}$, де $m > 0$, то завжди знайдуться такі цілі числа $q$ і $r$, які задовольняють рівність: $n = m \cdot q + r$, де $0 \leq r < m$.*

*Доведення.* Розглянемо вираз $m \cdot x$, в якому $x$ набуває довільних цілих значень як додатних, так і від'ємних. Зі зростанням $x$ величина $m \cdot x$ також зростає, тому для деякого цілого $x = q$ матимемо $m \cdot q \leq n$, але вже $m(q+1) > n$. Позначимо різницю $n - m \cdot q$ через $r$. Тоді з попередніх нерівностей отримуємо, що $n - m \cdot q = r \geq 0$ і $m > n - m \cdot q = r$, тобто $n = m \cdot q + r$ і $0 \leq r < m$.

Покажемо тепер єдиність частки й остачі. Припустимо, що існують, крім $q$ і $r$, інші частка $q_1$ і остача $r_1$, які задовольняють умову теореми. Тоді маємо

$$\begin{aligned} n &= m \cdot q + r, \quad 0 \leq r < m, \\ n &= m \cdot q_1 + r_1, \quad 0 \leq r_1 < m. \end{aligned}$$

Віднімаючи почленно ці рівності, дістаємо: $m(q - q_1) = r_1 - r$. А це означає, що $r_1 - r$ ділиться на $m$, проте це можливо лише тоді, коли $r = r_1$, тому що $(r_1 - r) < m$. Але звідси дістаємо $q = q_1$. ■

Отже, число $n$ кратне $m \neq 0$ тоді і тільки тоді, коли остача $r = 0$. Дійсно, в такому випадку маємо $n = m \cdot q$, а це означає, що $n$ кратне $m$.

Числа $q$ і $r$, про які йшлося в теоремі 17, називають відповідно **неповною часткою** та **остачею від ділення** $n$ на $m$. Оскільки $q$ – це найбільше ціле число, яке не перевищує частки від ділення

$n$ на $m$, то звідси випливає, що $q = \lfloor \frac{n}{m} \rfloor$ і $r = rest(n, m)$. Отже, отримуємо справедливість такої рівності:

$$n = m\lfloor \frac{n}{m} \rfloor + rest(n, m). \tag{2.2}$$

На цій формулі ґрунтується алгоритм переходу від десяткового запису цілого числа $n$ до запису його в системі за основою $m$.

**Приклад 2.5.1.** Нехай $m = 2$, тоді $n = 2\lfloor \frac{n}{2} \rfloor + rest(n, 2)$. Застосовуючи цю формулу для знаходження числа 53 в системі числення за основою 2, знаходимо:

$$53 = 2 \cdot 26 + 1,$$
$$26 = 2 \cdot 13 + 0,$$
$$13 = 2 \cdot 6 + 1,$$
$$6 = 2 \cdot 3 + 0,$$
$$3 = 2 \cdot 1 + 1,$$
$$1 = 2 \cdot 0 + 1.$$

Отже, $53 = 110101_2$.

Те саме число 53 в системі числення за основою $m = 5$ матиме розклад:

$$53 = 5 \cdot 10 + 3,$$
$$10 = 5 \cdot 2 + 0,$$
$$2 = 5 \cdot 0 + 2.$$

Отже, $53 = 203_5$. ♠

### 2.5.2. Найбільший спільний дільник чисел

Нехай $m, n \in \mathcal{N}$ і існує $x \in \mathcal{N}$ таке, що $x|m$ і $x|n$. У цьому випадку число $x$ називається **спільним дільником** чисел $m$ і $n$. Найбільше серед спільних дільників число $k$ називається **найбільшим спільним дільником (НСД)** чисел $m$ і $n$ ($k$=НСД$(m, n)$). Якщо НСД$(m, n)$ =1, то числа $m$ і $n$ називаються **взаємно простими**.

Важливу властивість НСД дає теорема 18.

**Теорема 18.** *Якщо m,n,q,r – цілі числа, пов'язані співвідношенням $n = m \cdot q + r$, то сукупність спільних дільників чисел $m$ і $n$ збігається із сукупністю спільних дільників чисел $m$ і $r$. Зокрема,* НСД$(m, n)$ = НСД$(m, r)$.

*Доведення.* Довільний спільний дільник чисел $m$ і $n$ на підставі пункту л) попереднього підрозділу буде також дільником числа $r$.

Навпаки, довільний спільний дільник чисел $m$ і $r$ буде дільником числа $n$ і, отже, є спільним дільником чисел $m$ і $n$. Таким чином, спільні дільники чисел $m$ і $n$ збігаються зі спільними дільниками чисел $m$ і $r$, зокрема НСД$(m, n)$ = НСД$(m, r)$. ■

Ця теорема дає простий спосіб обчислення НСД двох чисел, відомий під назвою **алгоритму Евкліда**. Він полягає в такому: припустимо, що $m$ і $n$ – два цілих числа, причому $m > 0$ і $n > 0$. Поділимо $n$ на $m$, і частку позначимо через $q_0$, а остачу через $r_1$, де $0 \le r_1 < m$. Далі поділимо $m$ на $r_1$ і частку позначимо через $q_1$, а остачу через $r_2$, де $0 \le r_2 < r_1$ і т. д.

Очевидно, що такий процес скінченний і він завершується, коли остача дорівнює нулю. Дійсно, внаслідок означення остачі маємо спадний ряд цілих чисел: $m > r_1 > r_2 > \ldots > r_k$, який може мати не більше $m$ членів. Отже,

$$
\begin{aligned}
& n = m \cdot q_0 + r_1, \qquad 0 \le r_1 < m; \\
& m = r_1 \cdot q_1 + r_2, \qquad 0 \le r_2 < r_1; \\
& r_1 = r_2 \cdot q_2 + r_3, \qquad 0 \le r_3 < r_2; \\
& \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
& r_{k-2} = r_{k-1} \cdot q_{k-1} + r_k, \qquad 0 \le r_k < r_{k-1}; \\
& r_{k-1} = r_k \cdot q_k.
\end{aligned}
$$

На підставі теореми 18 і наведених рівностей отримуємо, що НСД$(m, n)$ = НСД$(m, r_1)$ = НСД$(r_1, r_2)$ і т. д. Отже, НСД$(m, n)$ дорівнює $r_k$.

**Приклад 2.5.2.** Знайти НСД чисел 816 і 323. Послідовно знаходимо:

$$
\begin{aligned}
& 816 = 323 \cdot 2 + 170, \\
& 323 = 170 \cdot 1 + 153, \\
& 170 = 153 \cdot 1 + 17, \\
& 153 = 17 \cdot 9 + 0.
\end{aligned}
$$

Отже, НСД(816, 323) = 17. ♠

З наведених властивостей НСД випливає таке рекурсивне означення функції НСД:

$$
\text{НСД}(m, n) = \begin{cases} n, & \text{якщо } m = n, \\ \text{НСД}(m, rest(n, m)), & \text{якщо } m < n, \\ \text{НСД}(n, rest(m, n)), & \text{якщо } n < m. \end{cases}
$$

Звідси випливає такий очевидний алгоритм обчислення НСД:

НСД$(m,n)$ = if $m = n$ then return $(n)$ else
if $m < n)$ then НСД$(m, rest(n,m))$ else НСД$(n, rest(m,n))$.

**Приклад 2.5.3.** Знайти НСД(30,21) і НСД(5,0).
*Розв'язання.* НСД(30,21) = НСД(21,9) = НСД(9,3) = 3, оскільки $rest(9,3) = 0$.
НСД(5,0) = НСД(0,5) = 5, оскільки $rest(5,0) = 0$ і $rest(0,5) = 0$. ♠

Безпосередньо з алгоритму Евкліда отримуємо такі твердження.

**Теорема 19.** *Якщо* $d =$ НСД$(m,n)$*, то завжди знайдуться такі цілі числа* $a$ *і* $b$*, що буде справедлива рівність* $d = a \cdot n + b \cdot m$.

*Доведення.* З рівностей, які фігурували в алгоритмі Евкліда, отримуємо

$$r_1 = n - mq_0 = n \cdot 1 + (-q_0) \cdot m = n \cdot a_1 + m \cdot b_1,$$

де $a_1 = 1, b_1 = -q_0$ – цілі числа. З наступної рівності, записаної у вигляді $r_2 = m - r_1 q_1$, випливає, що

$$r_2 = m - (a_1 n + b_1 m)q_1 = n \cdot (-a_1 q_1) + m \cdot (1 - q_1 b_1) = n \cdot a_2 + m \cdot b_2,$$

де $a_2 = -a_1 q_1, b_2 = 1 - q_1 b_1$ – цілі числа. Переходячи від рівності до рівності згори вниз, дістаємо, що

$$r_k = d = na_k + m \cdot b_k, \text{ де } a_k, b_k \text{ – цілі числа.}$$

Позначаючи $a = a_k, b = b_k$, отримуємо рівність $d = n \cdot a + m \cdot b$. ■

**Приклад 2.5.4.** Знайти розклад НСД чисел 816 і 323 у вигляді $816 \cdot x + 323 \cdot y$.
*Розв'язання.* З прикладу 2.5.2 послідовно знаходимо:

$$170 = 816 - 323 \cdot (2),$$
$$153 = 323 - 170 = -816 + 323 \cdot (3),$$
$$17 = 170 - 153 = 816 \cdot (2) + 323 \cdot (-5).$$

Отже, НСД(816, 323) $= 816 \cdot (2) + 323 \cdot (-5) = 17$. ♠

**Наслідок 1.** а) *Якщо* $m$ *і* $n$ *взаємно прості числа, то* НСД$(m,n) = an + bm = 1$.

б) *Якщо добуток* $m \cdot n$ *ділиться на* $k$ *і* НСД$(m,k) = 1$*, то* $n$ *кратне* $k$.

в) *Якщо* $n$ *кратне* $m$ *та* $k$ *і* НСД$(m,k)=1$, *то* $n$ *кратне* $m\cdot k$.

г) *Якщо* НСД$(n,k)=$ НСД$(m,k)=1$, *то* НСД$(m\cdot n,k)=1$.

д) *Якщо* НСД$(m,n)=1$, *то* НСД$(m^k,n^k)=1$.

*Доведення.* а) випливає безпосередньо з теореми 19.

б) Оскільки НСД$(m,k)=1$, то $m\cdot a+k\cdot y=1$. Помноживши цю рівність на $n$, дістаємо $nma+nky=n$. Але ліва частина цієї рівності ділиться на $k$, отже, і права частина теж повинна ділитися на $k$.

в) Справді, із $m|n$ випливає, що $n=m\cdot a_1$, де $a_1$ – ціле. Далі, з $k|n$ випливає, що $n=k\cdot a_2$ і $k|m\cdot a_1$. Але оскільки НСД$(m,k)$=1, то на підставі теореми 19 отримуємо, що $a_1=k\cdot a_3$. Отже, $n=$ $=(mk)\cdot a_3$, а це означає, що $mk|n$.

г) З умови маємо $n\cdot a+k\cdot b=1$. Помноживши цю рівність на $m$, дістаємо $mna+mkb=m$. Припустимо, що НСД$(m\cdot n,k)=$ $=d>1$, тоді ліва частина рівності $mna+mkb=m$ ділиться на $d$ і тому $d|m$. Але зі зробленого припущення випливає, що й $d|k$, тобто НСД$(m,k)=d>1$. А це суперечить умові.

д) Доведення пропонується як вправа. ■

Маючи розклад НСД двох чисел, який дає теорема 19, можна знаходити розв'язки в цілих числах лінійного рівняння вигляду $ax+$ $+by=c$, де $a,b,c\in\mathcal{Z}$. Критерій існування розв'язків таких рівнянь випливає з наступної теореми.

**Теорема 20.** *Рівняння* $ax+by=c$, *де* $a,b,c\in\mathcal{Z}$, *має розв'язки в цілих числах тоді і тільки тоді, коли* НСД$(a,b)|c$. *Якщо* НСД$(a,b)=d$ *і* НСД$(a,b)|c$, *то ці розв'язки мають вигляд*

$$x=\tfrac{u\cdot c}{d},\quad y=\tfrac{v\cdot c}{d},$$

*де* $u,v$ *– довільний розв'язок рівняння* НСД$(a,b)=au+bv$.

*Доведення.* Нехай НСД$(a,b)=au+bv=d$ і НСД$(a,b)|c$, тоді $c=e\cdot d$ для деякого цілого числа $e$. Отже, $aue+bve=de=c$, це означає, що $x=ue$ і $y=ve$ є розв'язками рівняння $ax+by=c$.

Навпаки, нехай існують такі $x$ і $y$, що $ax+by=c$. Тоді на підставі того, що НСД$(a,b)$ ділить як $a$, так і $b$, дістаємо, що

НСД$(a,b)|ax+by$ і НСД$(a,b)|c$. ■

**Приклад 2.5.5.** Знайти розв'язки рівняння.

а) $816\cdot x+323\cdot y=51$; б) $252\cdot x+580\cdot y=20$; в) $252\cdot x+580\cdot y=15$.

*Розв'язання.* а) З прикладу 2.5.4 маємо НСД(816,323) $=17=816\cdot 2+323\cdot(-5)$. Отже,

$$x=\tfrac{uc}{17}=\tfrac{2\cdot 51}{17}=6,\quad y=\tfrac{vc}{17}=\tfrac{(-5)\cdot 51}{17}=-15.$$

б) Оскільки НСД(252, 580) $=4=252\cdot(-23)+580\cdot(10)$, то, домножуючи обидві частини рівняння на $e=5$, дістаємо $252\cdot(-115)+580\cdot(50)=20$. Отже, розв'язком даного рівняння будуть числа $(x,y)=(-115,50)$.

в) Рівняння розв'язків не має, оскільки 15 не кратне НСД(252, 580) $=4$. ♠

Зі сказаного вище випливає спосіб знаходження окремого розв'язку рівняння $ax+by=c$. Теорема 21 установлює загальний вигляд усіх розв'язків такого рівняння.

**Теорема 21.** *Якщо $a$ і $b$ – ненульові цілі числа і $(x_0,y_0)$ – розв'язок рівняння $ax+by=c$, то довільний розв'язок $(x,y)$ цього рівняння має вигляд*

$$x=x_0+\tfrac{b}{d}t,\quad y=y_0-\tfrac{a}{d}t,$$

*де $t$ – довільне ціле число, а $d=$* НСД$(a,b)$.

*Доведення.* Нехай $(x,y)$ і $(x_0,y_0)$ – розв'язки рівняння $ax+by=$ $=c$. Тоді $ax+by=ax_0+by_0$ і $a(x-x_0)=-b(y-y_0)$. Поділивши обидві частини на НСД$(a,b)=d$, дістаємо $\frac{a}{d}(x-x_0)=-\frac{b}{d}(y-y_0)$. Оскільки НСД$(\frac{a}{d},\frac{b}{d})=1$, то покладаючи $x-x_0=u\frac{b}{d}, y-y_0=-v\frac{a}{d}$, отримуємо $(\frac{a}{d})u(\frac{b}{d})=(\frac{b}{d})v(\frac{a}{d})$. Звідки дістаємо $u=v$. Покладаючи $t=u=v$, отримуємо $x=x_0+\frac{b}{d}t,\ \ y=y_0-\frac{a}{d}t$.

Покажемо, що пара $(x_0+\frac{b}{d}t, y_0-\frac{a}{d}t)$ є розв'язком рівняння $ax+$ $+by=c$. Дійсно,

$$a(x_0+\tfrac{b}{d}t)+b(y_0-\tfrac{a}{d}t)=ax_0+a\tfrac{b}{d}t+by_0-b\tfrac{a}{d}t=ax_0+by_0=c.\ \blacksquare$$

Наприклад, загальний розв'язок рівняння $816x+323y=51$ (див. приклад 2.5.5 а)) матиме вигляд: $x=6+19t,\ y=-15-48t$.

Задача знаходження НСД кількох чисел $m_1,m_2,\ldots,m_k$ зводиться до задачі обчислення НСД двох чисел. Справді, з алгоритму

Евкліда випливає, що сукупність спільних дільників чисел $m_1$ і $m_2$ збігається із сукупністю дільників числа $d = \text{НСД}(m_1, m_2)$. А звідси отримуємо

$$\text{НСД}(m_1, m_2, \ldots, m_k) = \text{НСД}(\ldots \text{НСД}(m_1, m_2), \ldots, m_k).$$

### 2.5.3. Факторизація цілих чисел

Зі сказаного випливає, що довільне ціле число можна подати у вигляді добутку степенів простих чисел. Побудова такого добутку для заданого числа називається його **факторизацією**. У зв'язку із цим розглянемо деякі властивості простих чисел і встановимо єдиність розкладу довільного натурального числа $m > 1$ на прості множники.

**Твердження 2.** *Нехай $p$ – просте число і $m \in \mathcal{N}$, $m \neq 1$, тоді*

а) *якщо $p$ ділиться на $m$, то $m = p$;*

б) *кожне натуральне число $m$ або ділиться на $p$, або з ним взаємно просте;*

в) *добуток двох (або кількох) натуральних чисел ділиться на $p$ тоді і тільки тоді, коли принаймні один із співмножників ділиться на $p$;*

г) *найменший відмінний від одиниці дільник числа $m > 1$ є числом простим;*

д) *простих чисел нескінченно багато;*

є) *найменший простий дільник складеного числа $m$ не більший за $\sqrt{m}$.*

*Доведення.* а) Якщо $m \neq p$, то $p$ має три дільники $1, m, p$ і тому не може бути простим числом.

б) На підставі простоти $p$ $\text{НСД}(m, p)$ дорівнює або 1, або $p$. У першому випадку $m$ і $p$ – взаємно прості, а в другому випадку $m$ кратне $p$.

в) Ця властивість випливає з того, що коли добуток $m \cdot n$ ділиться на $p$ і $\text{НСД}(n, p) = 1$, то $m$ повинно ділитися на $p$.

г) Нехай $q$ – найменший дільник числа $m > 1$, тобто $m = m_1 \cdot q$. Якби $q$ було складеним, то $q = q_1 \cdot q_2$, де $q_1 < q, q_2 < q$, і тоді

$m = m_1 q_1 q_2$. Оскільки $q_1$ і $q_2$ менші $q$, то це суперечить вибору числа $q$.

д) Припустимо, що простих чисел скінченне число і дорівнює $n$. Позначимо їх $p_1, p_2, \ldots, p_n$ і розглянемо число $m = p_1 p_2 \ldots p_n + 1$. Число $m$ або саме просте і більше від будь-якого з $p_i$, або має ділитися на деяке просте число, відмінне від $p_1, p_2, \ldots, p_n$. Отже, існує принаймні одне просте число, відмінне від $p_1, p_2, \ldots, p_n$, а це суперечить нашому припущенню.

є) Нехай $q$ – цей дільник, тоді $m = m_1 \cdot q$ і $m_1 \geq q$. Звідси, домножаючи на $q$, дістаємо $m = m_1 \cdot q \geq q^2$ і $q \leq \sqrt{m}$. ■

Остання властивість дає змогу зрозуміти є чи ні дане число $m$ простим і будувати розклад $m$ у випадку, коли воно складене. Дійсно, якщо $m$ не ділиться на жодне з простих чисел, менших $\sqrt{m}$, то $m$ – просте. Інакше, знаходимо найменше $p_1$, яке є дільником числа $m$, тобто $m = p_1 \cdot m_1$, де $m_1 \leq m$. Далі цей спосіб застосовується до числа $m_1$ і т. д.

**Теорема 22 (основна).** *Кожне натуральне число $m > 1$ єдиним способом можна подати у вигляді добутку простих чисел із точністю до порядку співмножників.*

*Доведення.* Існування такого подання легко доводиться методом математичної індукції, на якому зупинятися не будемо. Доведемо його єдиність.

Припустимо, що $m$ має два розклади

$$m = p_1 \cdot p_2 \cdots p_k = q_1 \cdot q_2 \cdots q_r$$

у вигляді добутку простих чисел. Права частина цієї рівності ділиться на $q_1$ і на підставі твердження 2 (пункт в)) один із співмножників лівої частини ділиться на $q_1$. Не обмежуючи загальності, можемо вважати, що цей співмножник дорівнює $p_1$. Оскільки $p_1$ – просте число, то $p_1 = q_1$. Скорочуючи обидві частини на $q_1$, дістаємо $p_2 \cdot p_3 \cdots p_k = q_2 \cdot q_3 \cdots q_r$. Повторюючи попередні міркування щодо цієї рівності, дістаємо $p_3 \cdots p_k = q_3 \cdots q_r$ і т. д. Якщо в одній із частин, наприклад лівій, скоротяться всі $p_i$, то і у правій частині

всі $q_i$ скоротяться тому, що рівність $1 = q_{r-k} \cdot q_{r-k+1} \cdots q_r$ можлива лише тоді, коли $q_{r-k} = q_{r-k+1} = \cdots = q_r = 1$.

Отже, перший розклад збігається з другим і єдиність розкладу $m$ на прості множники доведено. ■

У розкладі числа $m$ на прості множники деякі з них можуть повторюватися і тоді розклад матиме вигляд

$$m = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}. \tag{2.3}$$

Цей розклад називають **канонічним розкладом числа** $m \in \mathcal{N}$.

**Приклад 2.5.6.** Знайти канонічний розклад чисел 144 і 997.

*Розв'язання.* На підставі твердження 2 пункт е) слід перевірити всі прості числа менші $12 = \sqrt{144}$. Звідси знаходимо, що $144 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 3 = 2^4 \cdot 3^2$.

А для числа 997 необхідно перевірити всі прості числа, менші 31, оскільки $31 < \sqrt{997} < 32$. Жодне із цих чисел не є дільником 997, то 997 – просте число. ♠

З канонічного розкладу (2.3) і основного правила комбінаторики отримуємо формулу для підрахунку кількості дільників числа $m \in \in \mathcal{N}$:

$$nd(m) = (\alpha_1 + 1)(\alpha_2 + 1) \cdots (\alpha_k + 1).$$

Наприклад, число $144 = 2^4 \cdot 3^2$ матиме $5 \cdot 3 = 15$ дільників. Дійсно, цими дільниками будуть числа: 1, 2, 4, 8, 16, 3, 9, 6, 12, 24, 48, 18, 36, 72, 144.

Сума дільників числа $m \in \mathcal{N}$, канонічний розклад якого має вигляд (2.3), обчислюється за такою формулою:

$$\sigma(m) = \frac{p_1^{\alpha_1+1}-1}{p_1-1} \frac{p_2^{\alpha_2+1}-1}{p_2-1} \cdots \frac{p_k^{\alpha_k+1}-1}{p_k-1}.$$

У зв'язку з важливістю задачі факторизації розглянемо ще одну спеціальну можливість побудови канонічного розкладу натурального числа.

**Твердження 3.** а) *Якщо $x > 0$ – деяке дійсне число і $d$ – натуральне число, то кількість натуральних чисел, які не більші за $x$ і діляться на $d$, дорівнює $\lfloor \frac{x}{d} \rfloor$.*

б) *Якщо $x > 0$ – довільне дійсне число і $d$ – натуральне число, то* $\lfloor \frac{\lfloor x \rfloor}{d} \rfloor = \lfloor \frac{x}{d} \rfloor$.

*Доведення.* а) Розглянемо послідовність чисел $d, 2d, 3d, \ldots, sd$ таких, що $sd \le x < (s+1)d$. Тоді $s \le \frac{x}{d} < s+1$. Але це означає, що $s = \lfloor \frac{x}{d} \rfloor$.

б) Оскільки між $\lfloor x \rfloor$ і $x$ немає натуральних чисел, то кількість чисел, кратних $d$, і таких, що не перевищують $x$ і $\lfloor x \rfloor$, буде однаковою. На підставі пункту а) цього твердження їх буде $\lfloor \frac{x}{d} \rfloor$. Отже, $\lfloor \frac{\lfloor x \rfloor}{d} \rfloor = \lfloor \frac{x}{d} \rfloor$. ■

**Теорема 23.** *Показник, з яким дане просте число $p$ входить до канонічного розкладу числа $n!$, дорівнює*

$$\lfloor \frac{n}{p} \rfloor + \lfloor \frac{n}{p^2} \rfloor + \cdots + \lfloor \frac{n}{p^k} \rfloor, \tag{2.4}$$

*де $p^k \le n$, а $p^{k+1} > n$.*

*Доведення.* На підставі твердження 3, кількість співмножників добутку $n!$, кратних $p$, дорівнюватиме $\lfloor \frac{n}{p} \rfloor$. Цими співмножниками будуть числа $p, 2p, \ldots, \lfloor \frac{n}{p} \rfloor p$. Решта чисел цього добутку на $p$ не діляться. Отже, поява $p$ в канонічному розкладі $n!$ визначається добутком

$$M = p \cdot 2p \cdots \lfloor \tfrac{n}{p} \rfloor p = p^{\lfloor \frac{n}{p} \rfloor} \cdot \lfloor \tfrac{n}{p} \rfloor !.$$

Позначимо $\lfloor \frac{n}{p} \rfloor = n_1$, тоді $M = p^{n_1} n_1!$. Серед множників $1, 2, \ldots, n_1$ можуть бути числа, які кратні $p$: $p, 2p, \ldots, \lfloor \frac{n_1}{p} \rfloor p$. Їхній добуток дорівнює $p^{\lfloor \frac{n_1}{p} \rfloor} \lfloor \frac{n_1}{p} \rfloor !$, або позначаючи через $n_2 = \lfloor \frac{n_1}{p} \rfloor = \lfloor \frac{\lfloor \frac{n}{p} \rfloor}{p} \rfloor = \lfloor \frac{n}{p^2} \rfloor$ (на підставі твердження 3), дістаємо

$$M = M_1 \cdot p^{n_1 + n_2} n_2!,$$

де $M_1$ – добуток множників, що не діляться на $p$. Якщо $n_2 < p$, то процес закінчено, а якщо $n_2 \ge p$, то продовжуємо його далі.

Очевидно, що цей процес скінченний, бо $n > n_1 > n_2 > \ldots$, і при деякому $k$ виявиться, що $n_k < p$ і $\lfloor \frac{n_k}{p} \rfloor = \lfloor \frac{n}{p^{k+1}} \rfloor = 0$.

Отже,

$$M = M_{k-1} \cdot p^{n_1+n_2+\cdots+n_k} n_k!.$$

Серед множників $n_k!$ немає чисел, кратних $p$, тому що $n_k < p$. $M_{k-1}$ також не містить множників, кратних $p$, і, отже, до канонічного розкладу $n!$ просте число $p$ увійде з показником

$$n_1 + n_2 + \cdots + n_k = \lfloor \tfrac{n}{p} \rfloor + \lfloor \tfrac{n}{p^2} \rfloor + \cdots + \lfloor \tfrac{n}{p^k} \rfloor. \blacksquare$$

**Приклад 2.5.7.** а) Знайти показник степеня, з яким число 3 входить до розкладу 1000!.

б) Знайти канонічний розклад 30!.

*Розв'язання.* а) На підставі формули (2.4) отримаємо

$$\alpha = \lfloor \tfrac{1000}{3} \rfloor + \lfloor \tfrac{1000}{9} \rfloor + \lfloor \tfrac{1000}{27} \rfloor + \lfloor \tfrac{1000}{81} \rfloor + \lfloor \tfrac{1000}{243} \rfloor + \lfloor \tfrac{1000}{729} \rfloor = 333+111+37+12+4+1 = 498.$$

б) Для знаходження цього розкладу слід знайти всі степені простих чисел 2, 3, 5, 7, 11, 13, 17, 19, 23, 29. Отже,

$\lfloor \tfrac{30}{2} \rfloor = 15$, $\lfloor \tfrac{15}{2} \rfloor = 7$, $\lfloor \tfrac{7}{2} \rfloor = 3$, $\lfloor \tfrac{3}{2} \rfloor = 1$; $\lfloor \tfrac{30}{3} \rfloor = 10$, $\lfloor \tfrac{10}{3} \rfloor = 3$, $\lfloor \tfrac{3}{3} \rfloor = 1$;
$\lfloor \tfrac{30}{5} \rfloor = 6$, $\lfloor \tfrac{6}{5} \rfloor = 1$; $\lfloor \tfrac{30}{7} \rfloor = 4$, $\lfloor \tfrac{4}{7} \rfloor = 0$; $\lfloor \tfrac{30}{11} \rfloor = 2$, $\lfloor \tfrac{2}{11} \rfloor = 0$ і т. д.

У результаті канонічний розклад числа 30! має вигляд

$$2^{15+7+3+1}3^{10+3+1}5^{6+1}7^4 11^2 13^2 \cdot 17 \cdot 19 \cdot 23 \cdot 29 = 2^{26}3^{14}5^7 7^4 11^2 13^2 \cdot 17 \cdot 19 \cdot 23 \cdot 29. \spadesuit$$

Слід зазначити, що описані вище способи факторизації ефективно діють для відносно невеликих чисел.

### 2.5.4. Найпростіші методи факторизації

У зв'язку з важливістю проблеми факторизації не тільки в арифметиці, а і в криптографії, наведемо прості алгоритми факторизації.

**Метод пробного ділення (МПД)** ґрунтується на твердженні 2 є). Для заданого числа $n \in \mathcal{N}$ необхідно перевірити всі прості числа, які є його дільниками і не перевищують величини $\lfloor \sqrt{n} \rfloor$ (див. приклад 2.5.6). Алгоритм такої перевірки має вигляд

**АПД** $(n)$
*Вхід:* число $n \in \mathcal{N}$.
*Вихід:* розклад $n$ на прості множники.
*Метод:*
1. $t := 0$; $k := 0$;
2. if $n = 1$ then STOP;

3. $q := \lfloor \frac{n}{d_k} \rfloor; r := n \bmod d_k$;
4. if $r \neq 0$ then goto 6;
5. if $r = 0$ then $(t := t + 1; p_t := d_k; n := q; print(d_k)$; goto 3);
6. if $q > d_k$ then $(k := k + 1$; goto 3);
7. $t := t + 1; p_t := n; print(n$ is prime); STOP.

Складність цього алгоритму $O(\sqrt{n} \log^2 n)$ і його використання має сенс лише у випадку, коли $n$ ділиться на невелекі прості дільники. Для великих чисел $n$ алгоритм потребує великої кількості обчислень тому, що в інтервалі $[2, \sqrt{n}]$ кількість простих чисел асимптотично дорівнює $2(\sqrt{n}/\ln n)$. Для числа $n$, у десятковому записі якого міститься 100 цифр, знайдеться приблизно $4 \cdot 10^{42}$ простих чисел, на які потрібно ділити число $n$ для отримання його розкладу.

Алгоритм МПД працює краще, якщо в пам'яті комп'ютера є таблиця простих чисел. Але зберігання такої таблиці в пам'яті комп'ютера займає багато місця.

**Метод Ферма** на відміну від МПД дає можливість визначати великі дільники числа $n$, які не обов'язково прості. Цей метод Ферма описав ще у 1643 р. (рік народження І. Ньютона). Суть цього методу полягає в тому, що непарне число $n \in \mathcal{N}$, яке факторизується, записується у вигляді добутку $n = uv$ двох непарних чисел $u$ і $v$, де $u < v$. А далі метод визначає невід'ємні цілі числа $x$ і $y$, які задовольняють рівність $n = x^2 - y^2 = (x - y)(x + y)$. Тоді $u = x - y, v = x + y$ – дільники числа $n$. Звідси визначаємо $x = \frac{u+v}{2}, y = \frac{v-u}{2}$, де $0 \leq y < x \leq n$.

Якщо число $n$ просте, то єдині можливі значення $u = 1, v = n$, а звідси $x = \frac{n+1}{2}$ – найменше значення $x$, за якого $y = \sqrt{x^2 - n}$ буде цілим числом. Якщо $n = r^2$, то $x = r, y = 0$, а коли $n$ складене число і $n \neq r^2$ при $y \neq 0$, то $\sqrt{n + y^2} > n$. Можна показати, що існує натуральне число $x$ таке, що $\lfloor \sqrt{n} \rfloor < x < \frac{n+1}{2}$ і $\sqrt{x^2 - n}$ – ціле число.

**АФ($n$)**
*Вхід:* число $n \in \mathcal{N}$.
*Вихід:* розклад $n$ на прості множники.
*Метод:*

1. $x := \lfloor\sqrt{n}\rfloor$;
2. if $n = x^2$ then (print($x$); STOP) else ($x := x + 1$; goto 3);
3. if $x = \frac{n+1}{2}$ then (print($n$ is prime); STOP) else $y := \sqrt{x^2 - n}$;
4. if $y^2 = x^2 - n$ then (print $(x - y, x + y)$; STOP);
5. $x := x + 1$; goto 2.

**Приклад застосування АФ**. Знайти розклад числа $n = 1342127$ алгоритмом АФ.

*Розв'язання.* $x = \lfloor\sqrt{n}\rfloor = \lfloor\sqrt{1342127}\rfloor = 1158$. Оскільки $x^2 = 1158^2 = 1340964 \neq$ $\neq n$, то покладаємо $x = 1159$ і обчислюємо $\frac{n+1}{2} = 671064$. Оскільки $x \neq 671064$, то обчислюємо $y = \sqrt{x^2 - n} = \sqrt{1159^2 - 1342127} = 33,97...$ – не ціле. Покладаємо $x =$ $= 1160 \neq 671064$ і знаходимо $y = \sqrt{1160^2 - 1342127} = 58,93...$ – не ціле число.

Алгоритм працюватиме доки число $y = \sqrt{x^2 - n}$ не стане цілим, або не буде виконана рівність $x = \frac{n+1}{2} = 671064$.

Процес обчислень на кожній ітерації наведено нижче в таблиці.

| $x$ | $y = \sqrt{x^2 - n}$ |
|---|---|
| 1159 | 33,97... |
| 1160 | 58,93... |
| 1161 | 76,11... |
| 1162 | 50,09... |
| 1163 | 102,18... |
| 1164 | 113 |

.

У результаті виконання АФ знаходимо $x = 1164, y = 113$, а звідси $u = 1051, v = 1277$. ♠

Сучасні алгоритми факторизації будуть розглядатися в наступних розділах.

### 2.5.5. Функція Ойлера $\varphi(n)$

Функція Ойлера $\varphi(n)$ визначається для всіх натуральних чисел $n$, значенням якої є кількість натуральних чисел, менших від $n$ і взаємно простих з $n$. Для $n = 1$ покладають $\varphi(1) = 1$.

Для невеликих значень $n$ значення $\varphi(n)$ можна знайти простим підрахунком. Наприклад,

| $n$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\varphi(n)$ | 1 | 1 | 2 | 2 | 4 | 2 | 6 | 4 | 6 | 4 | 10 | 4 | 12 | 6 | 8 |

Для обчислення значення $\varphi(n)$ для довільного $n$ існує формула, за якою ці значення знаходять. Це випливає з такого твердження.

**Теорема 24.** *Нехай $x>0$ – довільне дійсне число, $p_1, p_2, \ldots, p_k$ – різні прості числа, а $B(x; p_1, p_2, \ldots, p_k)$ означає кількість цілих додатних чисел, не більших $x$ і не кратних жодному із чисел $p_1, p_2, \ldots, p_k$. Тоді*

$$B(x;, p_1, p_2, \ldots, p_k) = \lfloor x \rfloor - \lfloor \tfrac{x}{p_1} \rfloor - \lfloor \tfrac{x}{p_2} \rfloor - \cdots - \lfloor \tfrac{x}{p_k} \rfloor + \lfloor \tfrac{x}{p_1 p_2} \rfloor + \cdots + \\ + \lfloor \tfrac{x}{p_{k-1} p_k} \rfloor - \lfloor \tfrac{x}{p_1 p_2 p_3} \rfloor - \cdots - \lfloor \tfrac{x}{p_{k-2} p_{k-1} p_k} \rfloor + \cdots + (-1)^k \lfloor \tfrac{x}{p_1 p_2 \cdots p_k} \rfloor.$$

Користуючись цією формулою у випадку, коли $x = n$ – натуральне число, а $p_1, p_2, \ldots, p_k$ – прості числа, і тим, що кожне натуральне число взаємно просте з $n$ тоді і тільки тоді, коли воно не ділиться на жодне з простих множників числа $n$, знаходимо: $\varphi(n) = B(n; p_1, p_2, \ldots, p_k)$, тобто

$$\begin{aligned} \varphi(n) &= n - \tfrac{n}{p_1} - \tfrac{n}{p_2} - \cdots - \tfrac{n}{p_k} + \tfrac{n}{p_1 p_2} + \cdots + \tfrac{n}{p_{k-1} p_k} - \\ &- \tfrac{n}{p_1 p_2 p_3} - \cdots - \tfrac{n}{p_{k-2} p_{k-1} p_k} + \cdots + (-1)^k \tfrac{n}{p_1 p_2 \cdots p_k} = \\ &= n[1 - \tfrac{1}{p_1} - \tfrac{n}{p_2} - \cdots - \tfrac{n}{p_k} + \tfrac{n}{p_1 p_2} + \cdots + \tfrac{n}{p_{k-1} p_k} - \\ &- \tfrac{n}{p_1 p_2 p_3} - \cdots - \tfrac{n}{p_{k-2} p_{k-1} p_k} + \cdots + (-1)^k \tfrac{n}{p_1 p_2 \cdots p_k}]. \end{aligned}$$

Вираз, який міститься у квадратних дужках, запишемо як

$$\varphi(n) = \frac{n}{p_1 p_2 \cdots p_k}(p_1 - 1)(p_2 - 1) \cdots (p_k - 1) = n(1 - \frac{1}{p_1})(1 - \frac{1}{p_2}) \cdots (1 - \frac{1}{p_k}), \quad (2.5)$$

що випливає з правила множення біномів, які відрізняються лише другими членами.

**Приклад 2.5.8.** Знайти $\varphi(1620)$.

*Розв'язання.* Канонічний розклад числа 1620 має вигляд $1620 = 2^2 3^4 5$. Отже, $\varphi(1620) = 1620(1 - \frac{1}{2})(1 - \frac{1}{3})(1 - \frac{1}{5}) = 54 \cdot 8 = 432$. ♠

З формули 2.5 випливають такі властивості функції Ойлера:

а) якщо $n = p^k$, то $\varphi(n) = p^k(1 - \frac{1}{p}) = p^{k-1}(p - 1)$;

б) якщо $n = p$ – просте число, то $\varphi(n) = n - 1$;

в) $\varphi(m \cdot n) = \varphi(m)\varphi(n)$.

Доведемо пункт в), оскільки справедливість пунктів а) і б) очевидна.

Нехай канонічні розклади чисел $m$ і $n$ мають вигляд

$$m = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k} \text{ і } n = q_1^{\beta_1} q_2^{\beta_2} \cdots q_l^{\beta_l}.$$

З формули 2.5 дістаємо

$$\varphi(m \cdot n) = m \cdot n(1 - \tfrac{1}{p_1})(1 - \tfrac{1}{p_2}) \cdots (1 - \tfrac{1}{p_k})(1 - \tfrac{1}{q_1})(1 - \tfrac{1}{q_2}) \cdots (1 - \tfrac{1}{q_l}) =$$
$$= m(1 - \tfrac{1}{p_1})(1 - \tfrac{1}{p_2}) \cdots (1 - \tfrac{1}{p_k}) \cdot n(1 - \tfrac{1}{q_1})(1 - \tfrac{1}{q_2}) \cdots (1 - \tfrac{1}{q_l}) = \varphi(m)\varphi(n).$$

Доведена властивість в) для функції Ойлера називається мультиплікативністю. У загальному випадку означення мультиплікативної функції має вигляд.

**Означення 47.** *Функція $f(n)$ називається мультиплікативною, якщо*

а) *вона визначена для довільного $n \in \mathcal{N}$,*

б) *$f(n) \neq 0$ принаймні для одного $n \in \mathcal{N}$,*

в) *$f(m \cdot n) = f(m) \cdot f(n)$, де $m, n \in \mathcal{N}$ – взаємно прості числа.*

Простим прикладом мультиплікативної функції є $f(n) = n^s$. Дійсно,

$$f(m \cdot n) = (m \cdot n)^s = m^s n^s = f(m)f(n).$$

Найпростіші властивості мультиплікативних функцій дає теорема 25.

**Теорема 25.** *Якщо $f, f_1, f_2$ – мультиплікативні функції, то*

а) *$f(1) = 1$,*

б) *добуток $f_1(n) \cdot f_2(n)$ – мультиплікативна функція,*

в) *коли $n_1, n_2, \ldots, n_s$ – попарно взаємно прості числа, то*
$$f(n_1 \cdot n_2 \cdots n_s) = f(n_1)f(n_2) \cdots f(n_s),$$

г) *коли $n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$ – канонічний розклад числа $n$, а $\sum\limits_{d|n}$ означає суму за дільниками $d$ числа $n$, то*

$$\sum_{d|n} f(d) = [1 + f(p_1) + \cdots + f(p_1^{\alpha_1})] \cdots [1 + f(p_k) + \cdots + f(p_k^{\alpha_k})]. \quad (2.6)$$

*Доведення.* а) Дійсно, якщо $f(n) \neq 0$, то

$$f(n) = f(n \cdot 1) = f(n)f(1).$$

Отже, $f(1) = 1$.

б) Позначимо $f(n) = f_1(n) \cdot f_2(n)$, тоді отримуємо

$$f(1) = f_1(1)f_2(1) = 1.$$

Далі, якщо НСД$(n_1, n_2) = 1$, то

$$f(n_1n_2) = f_1(n_1n_2)f_2(n_1n_2) = f_1(n_1)f_1(n_2)f_2(n_1)f_2(n_2) = \\ = [f_1(n_1)f_2(n_1)][f_1(n_2)f_2(n_2)] = f(n_1)f(n_2).$$

А це означає мультиплікативність функції $f$.

в) Доведення методом математичної індукції за числом $s$.

Для $s = 1, 2$ справедливість твердження випливає з попередніх пунктів а) та б).

Припустимо, що твердження справедливе для $s - 1$ і доведемо його для $s$.

Оскільки НСД$(n_i, n_j) = 1$ для всіх $i \neq j$ за умовою, то НСД$(n_1n_2 \cdots n_{s-1}, n_s) = 1$. За означенням мультиплікативної функції маємо

$$f(n_1n_2 \cdots n_{s-1}n_s) = f(n_1n_2 \cdots n_{s-1})f(n_s).$$

На підставі припущення індукції, дістаємо

$$f(n_1n_2 \cdots n_{s-1}) = f(n_1)f(n_2) \cdots f(n_{s-1}),$$

а звідси очевидним чином випливає справедливість усього твердження.

г) Розкриємо дужки у правій частині (2.6). Дістаємо суму доданків, які мають вигляд

$$f(p_1^{\beta_1})f(p_2^{\beta_2}) \cdots f(p_k^{\beta_k}),$$

де $0 \leq \beta_i \leq \alpha_i$, $i = 1, 2, \ldots, k$. На підставі мультиплікативності функції $f$ маємо

$$f(p_1^{\beta_1}p_2^{\beta_2} \cdots p_k^{\beta_k}) = f(d),$$

оскільки $p_1^{\beta_1}p_2^{\beta_2} \cdots p_k^{\beta_k}$ – дільники числа $n$. З правила множення полінома на поліном випливає, що жоден такий доданок не буде пропущений і не повториться більше одного разу. Але саме це значення й буде тим, що стоїть у лівій частині тотожності (2.6). ■

### 2.5.6. Порівняння за модулем

Нехай $a, b, m \in \mathcal{Z}$, $m > 0$.

**Означення 48.** *Числа $a$ і $b$ називаються такими, що порівнюються або конгруентні за модулем $m$, якщо їхні остачі від ділення на $m$ рівні між собою.*

Коректність такого означення випливає з теореми про ділення з остачею (теорема 17). Якщо числа $a$ і $b$ порівнюються за модулем $m$, то це позначають $a \equiv b \pmod m$ або стисліше $a \equiv b\ (m)$.

Безпосередньо із цього означення випливають такі властивості відношення порівнювання.

**Теорема 26.** а) *$a \equiv b\ (m)$ тоді і тільки тоді, коли $a = b + mt$, де $t$ – ціле число.*

б) *$a \equiv b\ (m)$ тоді і тільки тоді, коли $a - b$ кратне $m$.*

в) *Відношення $\equiv$ є відношенням еквівалентності.*

г) *Якщо $a \equiv b\ (m)$ і $c \equiv d\ (m)$, то $a \pm c \equiv b \pm d\ (m)$.*

*Доведення.* а) Якщо $a \equiv b\ (m)$, то на підставі теореми про ділення з остачею дістаємо $a = mq + r$ і $b = mq_1 + r$, де $0 \le r < m$. Звідси знаходимо $a - b = m(q - q_1)$ і $a = b + mt$, де $t = q - q_1$ – ціле число.

Навпаки, нехай $a = b + mt$, де $t$ – ціле число. Подамо $b$ у вигляді $b = mq_1 + r$, де $0 \le r < m$, звідки дістаємо $a = m(q_1 + t) + r = mq + r$, де $q = q_1 + t$ – ціле число. А це означає, що $a \equiv b\ (m)$.

б) Якщо $a = b + mt$, де $t$ – ціле число, то звідси дістаємо $a - b = mt$, тобто $a - b$ кратне $m$. Якщо $a - b$ кратне $m$, то очевидно, що $a \equiv b\ (m)$.

в) Покажемо рефлексивність, симетричність і транзитивність відношення $\equiv$.

Рефлексивність очевидна, оскільки $a - a = 0$ завжди кратне $m$.

Симетричність випливає з того, що коли $a - b$ кратне $m$, то і $b - a$ теж кратне $m$.

Якщо $a \equiv b\ (m)$ і $b \equiv c\ (m)$, то $a - b$ і $b - c$ кратні $m$. А звідси отримуємо, що і $(a - b) + (b - c) = a - c$ має бути кратним $m$.

г) Нехай дано

$$a \equiv b\ (m), \quad c \equiv d\ (m). \tag{2.7}$$

Тоді, на підставі пункту б) цієї теореми можемо написати рівності:

$$a = b + mt_1, \quad c = d + mt_2, \tag{2.8}$$

де $t_1, t_2$ – цілі числа.

Додаючи почленно ці рівності, дістаємо

$$a + c = b + d + m(t_1 + t_2) = b + d + mt,$$

де $t = t_1 + t_2$ – ціле число. Але це означає, що $a + c \equiv b + d\ (m)$.

Аналогічне доведення і для різниці. ■

Звідси випливає, що відношення $\equiv$ є конгруентністю відносно операцій додавання і віднімання.

**Наслідок 2.** а) *Якщо* $a \equiv b\ (m)$, *то* $(a \pm c) \equiv (b \pm c)\ (m)$.

б) *Якщо* $a \equiv b\ (m)$, *то* $(a \pm mk) \equiv b\ (m)$.

*Доведення.* а) Якщо $a \equiv b\ (m)$, то додаючи до цієї конгруентності очевидну конгруентність $\pm c \equiv \pm c\ (m)$, отримуємо $(a \pm c) \equiv (b \pm \pm c)\ (m)$ на підставі теореми 26.

б) Випливає з означення конгруентності і теореми 26. ■

Доведена теорема та її наслідок описують адитивні властивості конгруентностей. Наступна теорема описує їхні мультиплікативні властивості.

**Теорема 27.** а) *Якщо* $a \equiv b(m)$ *і* $c \equiv d(m)$, *то* $(a \cdot c) \equiv (b \cdot d)(m)$.

б) *Якщо* $a \equiv b\ (m)$ *і* $n \in \mathcal{N}$, *то* $a^n \equiv b^n\ (m)$.

в) *Обидві частини конгруентності* $a \equiv b\ (m)$ *можна поділити на їхній спільний дільник, якщо він взаємно простий з* $m$.

г) *Обидві частини конгруентності* $a \equiv b\ (m)$ *і модуль* $m$ *можна помножити на одне й те саме натуральне число.*

д) *Обидві частини конгруентності* $a \equiv b\ (m)$ *і модуль* $m$ *можна поділити на будь-який їхній спільний дільник.*

*Доведення.* а) Розглянемо конгруентності (2.7) і рівності (2.8), які з них випливають. Перемножуючи почленно рівності (2.8), дістаємо $a \cdot c = b \cdot d + mt$, де $t$ – ціле число. На підставі теореми 26 маємо: $a \cdot c \equiv b \cdot d\ (m)$.

б) З попереднього пункту отримуємо, що коли $a \equiv b\ (m)$, то $a^2 \equiv b^2\ (m)$. А звідси отримуємо $a^n \equiv b^n\ (m)$ для довільного натурального $n$.

в) Нехай $a \equiv b\ (m)$, $a = a_1 d$, $b = b_1 d$ і НСД$(m, d) = 1$, тоді $a - b = d(a_1 - b_1)$ кратне $m$ і, оскільки НСД$(m, d)$=1, то $a_1 - b_1$ буде кратне $m$. А це означає, що $a_1 \equiv b_1\ (m)$.

г) З конгруентності $a \equiv b\ (m)$ випливає рівність $a = b + mt$, де $t$ – ціле число. Помножимо цю рівність на ціле $k > 0$: $ak = bk + mk \cdot t$. Із цієї рівності випливає, що $ak \equiv bk\ (mk)$.

д) Нехай $a \equiv b\ (m)$, $a = a_1 d$, $b = b_1 d$, $m = m_1 d$, тоді маємо $a = b + mt$, або $a_1 d = b_1 d + m_1 t d$. Звідси, скорочуючи обидві частини на $d$, дістанемо: $a_1 = b_1 + m_1 t$, а це означає $a_1 \equiv b_1\ (m_1)$. ■

Розглянемо ще деякі властивості конгруентностей.

**Теорема 28.** а) *Якщо* $a \equiv b\ (m_1)$ *і* $a \equiv b\ (m_2)$, *то* $a \equiv b\ (m)$, *де* $m =$ НСК$(m_1, m_2)$ – *найменше спільне кратне чисел* $m_1$ *і* $m_2$.

б) *Якщо* $a \equiv b\ (m)$, *то* $a \equiv b\ (d)$, *де* $d$ – *дільник модуля* $m$.

в) *Якщо* $a \equiv b\ (m)$ *і* $a$ *та* $m$ *кратні* $d$, *то і* $b$ *кратне* $d$;

г) *Якщо* $a \equiv b\ (m)$, *то* НСД$(a, m) =$ НСД$(b, m)$.

*Доведення.* а) З умови випливає, що $a - b$ кратне $m_1$ і $m_2$ і тому $a - b$ повинно бути кратним $m =$ НСК$(m_1, m_2)$. А це означає, що $a \equiv b\ (m)$.

б) Дійсно, з умови отримуємо $a - b$ кратне $m$ і тому $a - b$ кратне довільному дільнику $d$ модуля $m$. Отже, $a \equiv b\ (d)$.

в) Справді, з умови випливає, що $a = b + mt$, де $t$ – ціле число, і якщо $a$ і $m$ діляться на $d$, то і $b$ повинно ділитися на $d$.

г) Якщо $a \equiv b\ (m)$, то $a = b + mt$. На підставі теореми 18 множина спільних дільників $a$ і $m$ збігається з множиною спільних дільників $b$ і $m$. Зокрема, НСД$(a, m) =$ НСД$(b, m)$. ■

Звідси випливає, що відношення $\equiv$ є конгруентністю і відносно операцій множення і ділення.

**Класи за даним модулем**. Оскільки відношення конгруентності за модулем $m$ є відношенням еквівалентності, то множина цілих чисел $\mathcal{Z}$ розбивається на класи еквівалентності $C_0, C_1, \ldots, C_{m-1}$. Ці

класи для простоти позначатимемо $0, 1, \ldots, m-1$ відповідно. Отже, клас $k$ включає всі цілі числа, які порівнюються з $k$ за модулем $m$, тобто $n \equiv k\ (m)$, $k = 0, 1, \ldots, m-1$.

Оскільки значення операцій на класах не залежить від вибору представників, то визначимо операції на цих класах таким чином: $x \oplus y = rest(x + y, m)$ і $x \odot y = rest(x \cdot y, m)$. Звідси, з урахуванням властивостей операцій додавання і множення, випливає, що множина класів відносно введених операцій $\oplus$ і $\odot$ утворює асоціативно-комутативне кільце, яке позначають $Z_m$. У такому кільці можуть існувати ненульові елементи, добуток яких дорівнює нулю. Такі елементи називають *дільниками нуля*, а кільце, що має такі елементи, називають *кільцем із дільниками нуля*. Якщо кільце не має дільників нуля, то воно називається областю цілісності або кільцем без дільників нуля.

Якщо $m$ – просте число, то $Z_m$ є полем. Дійсно, нехай $a$ і $b$ – будь-які числа з множини $\{0, 1, \ldots, m-1\}$, такі, що $a \not\equiv 0\ (m)$ і $b \not\equiv 0\ (m)$. Тоді $ab \not\equiv 0\ (m)$ на підставі того, що НСД$(a, m)$ = НСД$(b, m)$=1.

Якщо $ab \equiv 0\ (m)$, то це означає, що $ab$ кратне $m$. Але це можливо лише тоді, коли або $a = 0$, або $b = 0$ на підставі простоти числа $m$ і того, що $a < m$ і $b < m$.

Звідси випливає, що $Z_m$ є областю цілісності, тобто є кільцем без дільників нуля. Оскільки $Z_m$ – скінченне, то воно є полем [14].

**Приклад 2.5.9.** Таблиці Келі поля $Z_5$ набувають вигляду

Таблиця 2.5.1

| $\oplus$ | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 |
| 1 | 1 | 2 | 3 | 4 | 0 |
| 2 | 2 | 3 | 4 | 0 | 1 |
| 3 | 3 | 4 | 0 | 1 | 2 |
| 4 | 4 | 0 | 1 | 2 | 3 |

,

Таблиця 2.5.2

| $\odot$ | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 |
| 2 | 0 | 2 | 4 | 1 | 3 |
| 3 | 0 | 3 | 1 | 4 | 2 |
| 4 | 0 | 4 | 3 | 2 | 1 |

.

За цими таблицями легко знаходять розв'язки рівняння $a \odot x = b$, наприклад, $3 \odot x = 4$ має розв'язок $x = 3$, а $4 \odot x = 1$ має розв'язок $x = 4$. ♠

Якщо модуль є складеним числом, тобто $m = k \cdot n$, де $k, n > 1$ і $k < m, n < m$, то $Z_m$ має дільники нуля. Дійсно, числа $k$ і $n$ відмінні від нуля, але оскільки $k \cdot n = m$, то $k \odot n = rest(k \cdot n, m) = 0$.

**Повні та зведені системи лишків**. Візьмемо з кожного класу лишків за модулем $m$ по одному представнику. У результаті дістаємо так звану *повну систему лишків* за модулем $m$. Як правило, за повну систему лишків беруть найменші невід'ємні числа $0, 1, \ldots, m-1$. Повна система лишків має такі основні властивості.

**Теорема 29.** а) *Довільні $m$ чисел, які попарно не конгруентні між собою за модулем $m$, утворюють повну систему лишків за цим модулем.*

б) *Якщо* НСД$(a, m) = 1$ *і $x$ пробігає повну систему лишків за модулем $m$, то вираз $ax + b$ пробігатиме повну систему лишків за цим модулем.*

*Доведення.* а) На підставі того, що всі $m$ чисел попарно не конгруентні між собою за модулем $m$, то вони є представниками різних класів за цим модулем. Оскільки цих класів $m$ і всіх представників теж $m$, то ці числа і складатимуть повну систему лишків.

б) Справді, значень виразу $ax + b$ буде стільки, скільки існує значень числа $x$, тобто $m$. На підставі попереднього пункту цієї теореми залишається показати, що будь-які два числа $ax_1 + b$ і $ax_2 + b$, що відповідають таким числам $x_1$ і $x_2$, що $x_1 \not\equiv x_2\ (m)$, самі будуть такими, що $ax_1 + b \not\equiv ax_2 + b\ (m)$. Припустимо супротивне, тобто що $ax_1 + b \equiv ax_2 + b\ (m)$, тоді отримуємо $ax_1 \equiv ax_2\ (m)$. За умовою НСД$(a, m) = 1$ і тому, поділивши обидві частини конгруентності на $a$, дістаємо: $x_1 \equiv x_2\ (m)$. А це суперечить тому, що $x_1 \not\equiv x_2\ (m)$. ■

**Приклад 2.5.10.** Нехай $x$ пробігає повну систему лишків за модулем 8. Знайти невід'ємні лишки виразу $3x + 2$.

*Розв'язання.* Маємо $x = 0, 1, 2, 3, 4, 5, 6, 7$. Тоді $3x + 2 = 2, 5, 0, 3, 6, 1, 4, 7$. ♠

Важливими класами лишків є такі, які включають числа, взаємно прості з модулем $m$. Очевидно, що серед повної системи лишків за модулем $m$ таких класів буде $\varphi(m)$, тобто кількість чисел, менших $m$ і взаємно простих з $m$.

Узявши по одному представнику з таких класів, дістаємо так звану *зведену систему лишків* за модулем $m$. Зведену систему лишків, як правило, отримують із повної системи лишків: $0, 1, \ldots, m-1$.

**Приклад 2.5.11.** Зведена система лишків за модулем 20 складатиметься з $\varphi(20) = 8$ лишків, взаємно простих із 20: 1,3,7,9,11,13,17,19. ♠

Зведена система лишків має такі основні властивості.

**Теорема 30.** а) *Довільні* $\varphi(m)$ *чисел, які взаємно прості з* $m$ *і попарно не конгруентні між собою за модулем* $m$*, утворюють зведену систему лишків за цим модулем.*

б) *Якщо* НСД$(a, m) = 1$ *і* $x$ *пробігає зведену систему лишків за модулем* $m$*, то вираз* $ax$ *теж пробігатиме зведену систему лишків за цим модулем.*

*Доведення.* а) Оскільки всі $\varphi(m)$ чисел попарно не конгруентні між собою за модулем $m$ і взаємно прості з $m$, то вони є представниками різних класів за цим модулем. Цих класів $\varphi(m)$ і всіх представників теж $\varphi(m)$, тому ці числа і складатимуть зведену систему лишків.

б) Справді, значень виразу $ax$ буде стільки, скільки існує значень числа $x$, тобто $\varphi(m)$. На підставі попереднього пункту цієї теореми, залишається показати, що будь-які два числа $ax_1$ і $ax_2$ неконгруентні між собою і взаємно прості з $m$. За умовою НСД$(a, m) =$ $= 1$ і НСД$(x, m)$=1 (оскільки $x$ пробігає зведену систему лишків за модулем $m$). Отже, НСД$(ax, m) = 1$. Коли б $ax_1 \equiv ax_2\ (m)$, то скорочуючи на $a$, дістали б $x_1 \equiv x_2\ (m)$. Але це неможливо, бо $x_1$ і $x_2$ належать до різних класів лишків за модулем $m$. ■

Із цієї теореми випливає, що зведена система лишків відносно операції множення $a \odot b = rest(ab, m)$ утворює абелеву групу. Одиницею групи буде клас 1. А з пункту б) попередньої теореми отримуємо, що кожний елемент $a$ зведеної системи лишків має обернений, тобто такий елемент $b$, що $ab = 1$.

Як застосування властивостей зведеної системи лишків, доведемо теорему Ойлера і малу теорему Ферма.

**Теорема 31 (Ойлера).** *Якщо* $m$ *– натуральне число і* $a$ *– ціле число, що* НСД$(a, m) = 1$*, то* $a^{\varphi(m)} \equiv 1\ (m)$*, де* $\varphi(m)$ *– функція Ойлера.*

*Доведення.* Припустимо, що $x$ пробігає зведену систему найменших невід'ємних лишків за модулем $m$: $r_1, r_2, \ldots, r_{\varphi(m)}$.

Тоді за властивістю б) теореми 30 значення $ax$ теж пробігатимуть зведену систему лишків за цим модулем. Отже, можна написати таку систему конгруентностей:

$$ar_1 \equiv c_1 \ (m), \quad ar_2 \equiv c_2 \ (m), \quad \ldots, \quad ar_{\varphi(m)} \equiv c_{\varphi(m)} \ (m),$$

де $c_1, c_2, \ldots, c_{\varphi(m)}$ є також найменші невід'ємні лишки, взаємно прості з $m$, тобто ті самі лишки, тільки розміщені в іншому порядку. Перемножуючи почленно ці конгруентності, матимемо

$$a^{\varphi(m)} r_1 r_2 \cdots r_{\varphi(m)} \equiv c_1 c_2 \cdots c_{\varphi(m)} \ (m).$$

Скорочуючи обидві частини на число $r_1 r_2 \cdots r_{\varphi(m)}$, взаємно просте з $m$ (бо всі $r_i$ і $c_i$ взаємно прості з $m$ за умовою), дістаємо $a^{\varphi(m)} \equiv \equiv 1 \ (m)$. ■

**Теорема 32 (Ферма мала).** *Якщо $p$ – просте число і $a$ – таке ціле число, що не ділиться на $p$, то $a^{p-1} \equiv 1 \ (p)$.*

*Доведення* випливає безпосередньо з теореми Ойлера, оскільки при простому $m = p$ значення $\varphi(p) = p - 1$. ■

З теореми Ойлера випливає такий важливий наслідок.

**Наслідок 3.** *Якщо $n = pq$, де $p$, $q$ – різні прості числа, а $e$ – взаємно просте число з $\varphi(n)$, то відображення $E(e, n) : x \to x^e \ (n)$ є бієкцією на множині лишків $Z_n = \{0, 1, \ldots, n-1\}$.*

*Доведення* пропонується виконати самостійно.

### 2.5.7. Лінійні конгруентності з невідомим

Лінійну конгруентність, яка містить невідоме $x$, завжди можна подати у такому вигляді:

$$ax \equiv c \ (m), \quad a \not\equiv 0 \ (m). \tag{2.9}$$

З'ясуємо, за яких умов ця конгруентність має розв'язки і якщо має, то скільки.

**Теорема 33.** *Якщо* НСД$(a, m) = 1$, *то конгруентність* (2.9) *має єдиний розв'язок.*

*Доведення.* Якщо НСД$(a, m) = 1$, то на підставі теореми 29 (пункт б)) можна стверджувати, що коли $x$ пробігає повну систему лишків за модулем $m$, то $ax$ теж пробігатиме повну систему лишків за цим модулем. Отже, тільки для одного значення $x = b$ з повної системи лишків матимемо конгруентність $ab \equiv c\ (m)$. ■

**Теорема 34.** *Якщо* НСД$(a, m) = d > 1$ *і* $c$ *кратне* $d$, *то конгруентність* (2.9) *має* $d$ *розв'язків. Якщо ж* $c$ *не ділиться на* $d$, *то конгруентність* (2.9) *не має розв'язків.*

*Доведення.* Якщо НСД$(a, m) = d$, тоді $a = a_1 d$, $m = m_1 d$ і НСД$(a_1, m_1) = 1$. Оскільки в конгруентності (2.9) $a$ і $m$ кратні $d$, то для того, щоб конгруентність виконувалася, число $c$ повинно ділитися на $d$ (теорема 28, пункт в)). Отже, $c = c_1 d$. Якщо ж $c$ не ділиться на $d$, то конгруентність не матиме розв'язків.

Нехай $c$ ділиться на $d$, тоді, скорочуючи обидві частини конгруентності і модуль на $d$, дістаємо нову конгруентність

$$a_1 x \equiv c_1\ (m_1), \tag{2.10}$$

в якій НСД$(a_1, m_1) = 1$ і яка еквівалентна (2.9).

Дійсно, нехай $s$ – довільний розв'язок конгруентності (2.9), тоді отримуємо тотожну конгруентніть $as \equiv c\ (m)$. Скорочуючи обидві частини і модуль цієї конгруентності на $d$, дістаємо нову тотожну конгруентність $a_1 s \equiv c_1\ (m_1)$, де НСД$(a_1, m_1) = 1$. Але це означає, що $s$ є розв'язком конгруентності (2.10).

Навпаки, нехай $t$ – довільний розв'язок конгруентності (2.10), тобто $a_1 t \equiv c_1\ (m_1)$. Помноживши цю тотожну конгруентність і модуль на $d$, дістаємо $at \equiv c\ (m)$. А це означає, що $t$ є розв'язком конгруентності (2.9).

Далі, на підставі попередньої теореми конгруентність (2.10) має єдиний розв'язок за модулем $m_1$. Нехай цим розв'язком буде клас $x \equiv b\ (m_1)$, де $b$ – найменший невід'ємний лишок цього класу.

Оскільки конгруентності (2.9) і (2.10) еквівалентні, то цей клас буде розв'язком конгруентності (2.9) за модулем $m$. Але за модулем $m = m_1 d$ він розпадеться на $d$ класів:

$$b, b + m_1, b + 2m_1, \ldots, b + (d-1)m_1,$$

тому, що $0 \leq b < m_1$. Отже, (2.9) матиме $d$ розв'язків. ■

З наведених теорем випливає, що розв'язання конгруентності (2.9) зводиться до розв'язання конгруентності (2.10), коли НСД$(a_1, m_1) = 1$.

**Приклад 2.5.12.** Знайти розв'язки конгруентностей:
а) $13x \equiv 4$ (24); б) $105x \equiv 75$ (125).

*Розв'язання.* а) Оскільки НСД(13,24) = 1, то конгруентність має єдиний розв'язок. Для знаходження цього розв'язку скористаємося теоремою Ойлера. Для цього домножимо обидві частини конгруентності на $13^{\varphi(24)-1}$: $13^{\varphi(24)}x \equiv 4 \cdot 13^{\varphi(24)-1}$ (24). На підставі того, що $13^{\varphi(24)} \equiv 1$ (24), дістаємо

$$x \equiv 4 \cdot 13^{\varphi(24)-1} \equiv 4 \cdot 13^{8-1} \equiv 4 \cdot 13 \equiv 4 \ (24).$$

Отже, $x \equiv 4$ (24) – шуканий розв'язок даної конгруентності.

б) Скорочуючи обидві частини конгруентності і модуль на НСД(105,25) = 5, дістаємо конгруентність $21x \equiv 15$ (25). Число $c = 75$ кратне 5, і тому конгруентність матиме 5 розв'язків. Цими розв'язками будуть числа $x \equiv 15, 40, 65, 90, 115$ (125). Розв'язок $x \equiv 15$ (25) знаходиться тим же способом, що і в попередньому пункті, а інші розв'язки знаходяться із цього розв'язку шляхом додавання числа 25. ♠

Отже, нам відомі умови існування розв'язків лінійної конгруентності (2.9). Інколи необхідно шукати розв'язки системи конгруентностей за різними модулями:

$$\begin{cases} a_1 x & \equiv & b_1 \ (n_1), \\ a_2 x & \equiv & b_2 \ (n_2), \\ \ldots & \ldots & \ldots \\ a_r x & \equiv & b_r \ (n_r), \end{cases} \tag{2.11}$$

де $a_i, b_i \in \mathcal{Z},\ a_i \neq 0,\ i = 1, 2, \ldots, r$.

Зрозуміло, що система (2.11) матиме розв'язки тільки тоді, коли кожна з конгруентностей їх має. Нехай $d_i$ = НСД$(a_i, n_i)$, де $i =$

$= 1, \ldots, r$. Тоді на підставі попередньої теореми необхідно, щоб $d_i | b_i$, а це дає змогу редукувати систему (2.11) до простішого вигляду:

$$\begin{cases} x & \equiv & c_1 \ (m_1), \\ x & \equiv & c_2 \ (m_2), \\ \ldots & \ldots & \ldots \\ x & \equiv & c_r \ (m_r), \end{cases} \tag{2.12}$$

де $m_i = n_i/d_i, \ i = 1, 2 \ldots, r$ і $c_i = (b_i/d_i) \cdot (a_i/d_i)^{-1} \pmod{m_i}$.

Таким чином, досить уміти розв'язувати систему конгруентностей (2.12). Покажемо, що коли система (2.12) сумісна, то вона має єдиний розв'язок за модулем $M$, що дорівнює найменшому спільному кратному чисел $m_1, m_2, \ldots, m_r$.

Справді, припустимо, що $x$ і $y$ – довільні розв'язки системи (2.12), тоді $x \equiv c_i (m_i)$ і $y \equiv c_i (m_i), \ i = 1, 2, \ldots, r$. Звідси випливає конгруентність $x \equiv y \ (m_i)$, тобто число $x - y$ ділиться на $m_1, m_2, \ldots, m_r$, а тому ділиться і на їхнє найменше спільне кратне $M$. Отже, розв'язки $x$ і $y$ збігаються.

Розглянемо окремий випадок, коли модулі $m_1, m_2, \ldots, m_r$ попарно взаємно прості.

**Теорема 35 (китайська про остачі).** *Якщо в системі* (2.12) *модулі* $m_1, m_2, \ldots, m_r$ *попарно взаємно прості, то ця система конгруентностей має єдиний розв'язок* $x \equiv x_0 \ (m_1 m_2 \cdots m_r)$, *де*

$$x_0 = M_1 y_1 c_1 + M_2 y_2 c_2 + \cdots + M_r y_r c_r, \tag{2.13}$$

*а числа* $M_i$ *і* $y_i$ *визначаються з умов*

$$M_i = \frac{m_1 m_2 \cdots m_r}{m_i}, \ M_i y_i \equiv 1 \ (m_i), \ i = 1, 2 \ldots, r. \tag{2.14}$$

*Доведення.* Підставляючи значення $x_0$ в конгруентності системи (2.12) $x \equiv c_i \ (m_i)$ і беручи до уваги те, що всі $M_j \neq M_i$ на підставі (2.14) діляться на $m_i$ та конгруентність $M_i y_i \equiv 1 \ (m_i)$ має єдиний розв'язок (тому, що НСД$(M_i, m_i) = 1$), дістаємо

$$x_0 \equiv M_i y_i c_i \equiv c_i \ (m_i).$$

А це означає, що $x_0$ задовольняє будь-які конгруентності системи (2.12), а тому і всю систему (2.12). Таким чином, показано, що $x_0$ – єдиний розв'язок системи (2.12) за модулем $M = m_1 m_2 \cdots m_r$, бо найменше спільне кратне попарно взаємно простих чисел дорівнює їхньому добутку. ■

Формула (2.13), за допомогою якої знаходиться розв'язок $x_0$ системи конгруентностей, називається *алгоритмом Гаусса* розв'язання системи конгруентностей (2.12). Складність цього алгоритму має оцінку $O(\log^2 n)$.

**Приклад 2.5.13.** Знайти розв'язок системи

$$\begin{cases} x & \equiv & 15\ (17), \\ x & \equiv & 11\ (13), \\ x & \equiv & 3\ (10). \end{cases}$$

*Розв'язання.* Маємо $M = 17 \cdot 13 \cdot 10 = 2210$, $M_1 = \frac{2210}{17} = 130, M_2 = \frac{2210}{13} = 170$, $M_3 = \frac{2210}{10} = 221$.

Розв'язуючи конгруентності $130y_1 \equiv 1\ (17), 170y_2 \equiv 1\ (13), 221y_3 \equiv 1\ (10)$, дістаємо $y_1 = 14, y_2 = 1, y_3 = 1$. Отже, за теоремою 35 знаходимо

$$x = x_0 = 130 \cdot 14 \cdot 15 + 170 \cdot 1 \cdot 11 + 221 \cdot 1 \cdot 3 = 29833 \equiv 1103 \pmod{2210}.$$ ♠

У загальному випадку, коли модулі $m_1, m_2, \ldots, m_r$ можуть і не бути попарно взаємно простими числами, систему (2.12) можна розв'язати так.

З першої конгруентності системи (2.12) знаходимо $x = c_1 + m_1 t_1$, де $t_1$ пробігає всі цілі числа. Щоб вибрати з них значення $x$, які б задовольняли другу конгруентність, визначимо $t_1$ з умови, що

$$c_1 + m_1 t_1 \equiv c_2 \pmod{m_2}, \text{ або } m_1 t_1 \equiv c_2 - \mathrm{c}_1 \pmod{m_2}.$$

Ця конгруентність першого степеня відносно $t_1$ матиме розв'язки, якщо $c_2 - c_1$ кратне НСД$(m_1, m_2)$, інакше ця конгруентність не має розв'язків, а тому і вся система (2.12) несумісна. Нехай $c_2 - c_1$ ділиться на НСД$(m_1, m_2)$. Розв'язуючи цю конгруентність, дістанемо

$$t_1 \equiv t_1' \pmod{\tfrac{m_2}{\text{НСД}(m_1, m_2)}}.$$

Тоді сукупність усіх значень $t_1$, що задовольняють другу конгруентність, буде

$$t_1 = t_1' + \frac{m_2}{\text{НСД}(m_1,m_2)}t_2,$$

де $t_2$ пробігає всі цілі числа. Звідси дістаємо

$$x = c_1 + m_1t_1' + \frac{m_1m_2}{\text{НСД}(m_1,m_2)}t_2 = y_2 + \text{НСК}(m_1,m_2)t_2, \qquad (2.15)$$

де $y_2 = c_1 + m_1t_1'$ задовольняє перші дві конгруентності. Тепер із чисел (2.15) аналогічним чином вибираються ті, які задовольняють третю конгруентність. У результаті або знайдемо, що вона несумісна, або знайдемо значення

$$x = y_3 + \text{НСК}(m_1,m_2,m_3)t_3,$$

де $t_3$ – ціле число, або

$$x \equiv y_3 \pmod{\text{НСК}(m_1,m_2,m_3)}$$

задовольнятиме перші три конгруентності системи (2.12). Продовжуючи далі застосовувати цей спосіб, прийдемо або до несумісності системи, або до її остаточних розв'язків.

**Приклад 2.5.14.** Розв'язати систему конгруентностей

$$\begin{cases} x \equiv 2\ (15), \\ x \equiv 7\ (20), \\ x \equiv 12\ (35). \end{cases}$$

*Розв'язання.* З першої конгруентності маємо $x = 2 + 15t_1$, де $t_1$ – ціле число. Щоб визначити $t_1$, підставимо значення $x$ у другу конгруентність, тоді матимемо

$$15t_1 \equiv 5\ (20),$$

або, скорочуючи на 5, знайдемо $3t_1 \equiv 1\ (4)$. Звідси $t_1 \equiv 3\ (4)$ або $t_1 = 3 + 4t_2$, де $t_2$ – довільне ціле число. Тепер

$$x = 2 + 15t_1 = 47 + 60t_2,$$

що задовольняє перші дві конгруентності. Із цих чисел виберемо ті, які задовольняють третю конгруентність. Для цього визначимо $t_2$ з умови

$$47 + 60t_2 \equiv 12 \pmod{35},$$

звідки випливає

$$25t_2 \equiv 0\ (35),\ 5t_2 \equiv 0\ (7), t_2 \equiv 0\ (7),$$

або $t_2 = 7t$, де $t$ – довільне ціле число.

Отже, розв'язками системи буде $x = 47 + 420t$ або $x \equiv 47\ (\mathrm{mod}\ 420)$. ♠

Лінійні конгруентності використовують в алгоритмах шифрування, де елементами перетворення виступають окремі літери відкритого тексту (як у шифрі Цезаря, який розглядається в наступному розділі). Якщо використовують блокові алгоритми шифрування, в яких відкритий текст розбивається на блоки по дві літери (**біграми**) і кожний такий блок перетворюється окремо, то застосовуються матриці розмірності $2\times2$. Розглянемо такі матриці над кільцем лишків $\mathcal{Z}_q$, де $q > 1, q \in \mathcal{N}$.

Позначимо $M_2(\mathcal{Z}_q)$ множину всіх квадратних матриць розмірності $2\times2$ над кільцем $\mathcal{Z}_q$. Множина $M_2(\mathcal{Z}_q)$ також буде асоціативним, але не комутативним кільцем над $\mathcal{Z}_q$. Матриця

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

має обернену матрицю тоді і тільки тоді, коли її визначник $D = ad - bc \neq 0$. У цьому випадку обернена матриця до матриці $A$ має вигляд

$$A^{-1} = \begin{pmatrix} D^{-1}d & -D^{-1}b \\ -D^{-1}c & D^{-1}a \end{pmatrix}.$$

Дійсно,

$$A \cdot A^{-1} = \begin{pmatrix} adD^{-1} - bcD^{-1} & -abD^{-1} + abD^{-1} \\ cdD^{-1} - cdD^{-1} & -bcD^{-1} + adD^{-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Зауважимо, що для того, щоб матриця $A$ мала обернену, необхідно щоб обернений елемент $D^{-1}$ існував у кільці $\mathcal{Z}_q$. Оскільки кільце $\mathcal{Z}_q$ може мати дільники нуля, то для існування оберненого елемента $D^{-1}$ повинна виконуватися умова НСД$(D, q) = 1$ (див. теорему 33).

**Приклад 2.5.15.** Знайти обернену матрицю до матриці

$$A = \begin{pmatrix} 2 & 3 \\ 7 & 8 \end{pmatrix}$$

в кільці лишків за модулем 26.

*Розв'язання.* Знаходимо $D = 2 \cdot 8 - 3 \cdot 7 = -5 = 21\ (26)$. Оскільки НСД(21,26) дорівнює 1, то для $D$ існує обернений елемент $D^{-1} = 5\ (26)$. Отже,

$$A^{-1} = \begin{pmatrix} 5 \cdot 8 & -5 \cdot 3 \\ -5 \cdot 7 & 5 \cdot 2 \end{pmatrix} = \begin{pmatrix} 40 & -15 \\ -35 & 10 \end{pmatrix} = \begin{pmatrix} 14 & 11 \\ 17 & 10 \end{pmatrix}.$$

Дійсно,

$$A \cdot A^{-1} = \begin{pmatrix} 2 & 3 \\ 7 & 8 \end{pmatrix} \cdot \begin{pmatrix} 14 & 11 \\ 17 & 10 \end{pmatrix} = \begin{pmatrix} 79 & 52 \\ 234 & 157 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \spadesuit$$

З лінійної алгебри відомо, що довільна матриця відповідає деякому лінійному відображенню векторного простору над $\mathcal{Z}_q$. У нашому випадку відображення $\varphi : \mathcal{Z}_q^2 \to \mathcal{Z}_q^2$. Справедливе таке твердження.

**Твердження 4.** *Нехай $\mathcal{Z}_q$ – кільце лишків за модулем $q$ і*

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

*– матриця над цим кільцем, а $D = ad - bc$ – її визначник. Тоді такі умови еквівалентні:*

а) *НСД(D,q) =1;*

б) *матриця $A$ має обернену матрицю;*

в) *якщо принаймні один з елементів $x, y \in \mathcal{Z}_q$ відмінний від нуля, то $A \cdot (x, y)^T \neq (0, 0)^T$ (індекс $T$ означає транспонування);*

г) *матриця $A$ задає бієктивне відображення $\mathcal{Z}_q^2$ на $\mathcal{Z}_q^2$.*

*Доведення.* Вище було показано, що з умови а) випливає умова б). Доведемо імплікації: б) $\to$ в) $\to$ г) $\to$ а).

Нехай виконується умова б), тоді виконується умова г), оскільки $A^{-1}$ визначає обернене відображення $\mathcal{Z}_q^2$ на $\mathcal{Z}_q^2$. Якщо виконана умова г), тоді із $(x, y) \neq (0, 0)$ випливає $A \cdot (x, y)^T \neq A \cdot (0, 0)^T =$

$= (0,0)^T$, а звідси випливає умова в). Нарешті, нехай виконується умова в), покажемо методом від супротивного, що виконується умова а).

Припустимо, що а) хибна. Покладемо $m = \text{НСД}(D,q) > 1$ і $m' = q/m$. Можливі такі три випадки.

*Випадок 1.* Якщо всі чотири елементи матриці $A$ діляться на $m$, то візьмемо $(x,y) = (m',m')$. А це приводить до суперечності з умовою в).

*Випадок 2.* Якщо хоча б один з елементів $a$ і $b$ не ділиться на $m$, то візьмемо $(x,y) = (-bm', am')$. Тоді

$$A \cdot (x,y)^T = \begin{pmatrix} -abm' + abm' \\ -cbm' + dam' \end{pmatrix} = \begin{pmatrix} 0 \\ Dm' \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

оскільки $m|D$ і, отже, $q = mm'|Dm'$.

*Випадок 3.* Якщо хоча б один з елементів $c$ і $d$ не ділиться на $m$, то візьмемо $(x,y) = (dm', -cm')$ і доведення точно повторює те, яке було в попередньому випадку.

Таким чином, у всіх трьох випадках дістаємо справедливість умови а), а разом із нею і всього твердження. ■

**Приклад 2.5.16.** Знайти розв'язки систем конгруентностей:

$$\text{а)} \begin{cases} 2x + 3y \equiv 1 \ (26), \\ 7x + 8y \equiv 2 \ (26); \end{cases} \qquad \text{б)} \begin{cases} x + 3y \equiv 1 \ (26), \\ 7x + 9y \equiv 2 \ (26); \end{cases}$$

$$\text{в)} \begin{cases} x + 3y \equiv 1 \ (26), \\ 7x + 9y \equiv 1 \ (26). \end{cases}$$

*Розв'язання.* а) У матричному вигляді маємо

$$A = \begin{pmatrix} 2 & 3 \\ 7 & 8 \end{pmatrix},$$

$X = (x,y)^T$, $b = (1,2)^T$. Для матриці $A$ була знайдена обернена матриця $A^{-1}$ у попередньому прикладі і оскільки визначник $D \neq 0$ (26), то система має єдиний розв'язок:

$$X = A^{-1}b = \begin{pmatrix} 14 & 11 \\ 17 & 10 \end{pmatrix} \cdot (1,2)^T = (10,11)^T \ (26).$$

б) Матриця системи не має оберненої, тому що її визначник $D = 14$ не взаємно простий з модулем 26 (НСД(14,26)=2). Але можна шукати розв'язки за модулем 13,

а потім перевірити, чи не є вони розв'язками системи за модулем 26. За модулем 13 знаходимо

$$X = \begin{pmatrix} 9 & 10 \\ 6 & 1 \end{pmatrix} \cdot (e, f)^T,$$

де $(e, f)^T = (1, 2)^T$ для системи б) і $(e, f)^T = (1, 1)^T$ для системи в). Це дає $(x, y) = (3, 8)$ і $(x, y) = (6, 7)$ відповідно.

Перевірка показує, що система б) несумісна, а система в) має два розв'язки (6,7) і (19,20) = (6+13,7+13) за модулем 26. ♠

### 2.5.8. Конгруентності другого степеня

Покажемо, що дослідження і розв'язування конгруентностей довільного степеня з одним невідомим за складеним модулем, зводиться до конгруентностей з одним невідомим за простим модулем.

Розглянемо конгруентність довільного степеня з одним невідомим за модулем простого числа $p$, тобто конгруентність

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_x + a_0 \equiv 0(p). \tag{2.16}$$

Справедливе таке твердження.

**Твердження 5.** *Конгруентність (2.16) еквівалентна конгруентності, степінь якої не більший* $p - 1$.

*Доведення.* Поділивши поліном $f(x)$ на $x^p - x$, дістаємо

$$f(x) = (x^p - x)g(x) + r(x),$$

де степінь полінома $r(x)$ не більший $p - 1$. А на підставі теореми Ферма $x^p - x \equiv 0\ (p)$, отримуємо $f(x) \equiv r(x)\ (p)$. ■

**Теорема 36.** *Якщо* $m_1, m_2, \ldots m_k$ *попарно взаємно прості числа, то конгруентність*

$$f(x) \equiv 0\ (m_1 m_2 \cdots m_k) \tag{2.17}$$

*еквівалентна системі конгруентностей*

$$\begin{cases} f(x) \equiv 0\ (m_1), \\ f(x) \equiv 0\ (m_2), \\ \ldots, \\ f(x) \equiv 0\ (m_k). \end{cases} \tag{2.18}$$

*Якщо $t_1, t_2, \ldots, t_k$ – кількість розв'язків окремих конгруентностей цієї системи за відповідними модулями і $t$ – кількість розв'язків конгруентності (2.17), то $t = t_1 t_2 \cdots t_k$.*

*Доведення.* Перша частина твердження випливає з пунктів а) і б) теореми 28. Справді, нехай $d$ – розв'язок конгруентності (2.17), тобто $f(d) \equiv 0\ (m_1 m_2 \cdots m_k)$, тоді звідси й поготів буде справедлива конгруентність $f(d) \equiv 0\ (m_i)$ для всіх $i = 1, 2, \ldots, k$. А це означає, що $d$ – розв'язок системи конгруентностей (2.18).

Навпаки, якщо $d$ – розв'язок системи (2.18), то виконуються конгруентності

$$f(d) \equiv 0\ (m_i), \text{ де } i = 1, 2, \ldots, k.$$

Але тоді на підставі пункту а) теореми 28 справедлива буде і конгруентність за модулем, який є найменшим спільним кратним модулів $m_1, m_2, \ldots, m_k$. А оскільки ці модулі попарно взаємно прості, то і за модулем $m_1 m_2 \cdots m_k$, тобто

$$f(d) \equiv 0\ (m_1 m_2 \cdots m_k).$$

А це означає, що $d$ – розв'язок конгруентності (2.17).

Друга частина твердження випливає з таких міркувань. Припустимо, що $x \equiv b_i\ (m_i)$ – довільний розв'язок конгруентності $f(x) \equiv 0\ (m_i)$, тоді на підставі китайської теореми про лишки, завжди можна знайти єдине число $x_1$, яке буде розв'язком системи лінійних конгруентностей

$$\begin{cases} x \equiv b_1\ (m_1), \\ x \equiv b_2\ (m_2), \\ \ldots, \\ x \equiv b_k\ (m_k). \end{cases} \tag{2.19}$$

Число $x_1$ визначається за модулем $m_1 m_2 \cdots m_k$ і буде розв'язком системи (2.18), а також і конгруентності (2.16). Тоді, комбінуючи кожний розв'язок однієї конгруентності із системи (2.18) з кожним розв'язком решти конгруентностей, дістаємо число $t = t_1 t_2 \cdots t_k$ лінійних систем конгруентностей типу (2.19). Оскільки кожна така

система має єдиний розв'язок, то на цьому доведення теореми закінчується. ■

**Наслідок 4.** *а) Якщо хоч одна з конгруентностей системи (2.18) не має розв'язків, то і конгруентність (2.16) також не матиме розв'язків.*

*б) Розв'язування конгруентності* $f(x) \equiv 0 \pmod{m}$, де $m = p_1^{s_1} p_2^{s_2} \cdots p_k^{s_k}$ – канонічний розклад модуля m, зводиться до дослідження і розв'язування конгруентностей

$$f(x) \equiv 0 \ (p_i^{s_i}), \tag{2.20}$$

де $i = 1, 2, \ldots, k$.

Покажемо, що дослідження і розв'язування конгруентностей типу

$$f(x) \equiv 0 \ (p^s). \tag{2.21}$$

зводиться до дослідження і розв'язування конгруентностей типу

$$f(x) \equiv 0 \ (p). \tag{2.22}$$

Бачимо, що довільне $x$, яке є розв'язком (2.21), буде розв'язком (2.22).

Нехай $x \equiv x_1 \ (p)$ – довільний розв'язок (2.22). Тоді $x = x_1 + pt_1$, де $t_1$ – ціле число. Підставляючи $x$ у конгруентність $f(x) \equiv 0 \ (p^2)$ і розкладаючи $f(x)$ у ряд Тейлора, дістаємо (на підставі того, що $\frac{1}{k!} f^{(k)}(x_1)$ ціле число, і відкидання членів, кратних $p^2$)

$$f(x_1) + pt_1 f'(x_1) \equiv 0 \ (p^2) \text{ або } \frac{f(x_1)}{p} + t_1 f'(x_1) \equiv 0 \ (p).$$

Обмежуючись випадком, коли $f'(x_1)$ не кратне $p$, отримуємо єдиний розв'язок:

$$t_1 \equiv t_1' \ (p) \text{ або } t_1 = t_1' + pt_2.$$

Вираз для $x$ набуває вигляду

$$x = x_1 + pt_1' + p^2 t_2 = x_2 + p^2 t_2.$$

Підставляючи його в конгруентність $f(x) \equiv 0\ (p^3)$, дістаємо

$$f(x_2) + p^2 t_2 f'(x_2) \equiv 0\ (p^3) \text{ або } \tfrac{f(x_2)}{p^2} + t_2 f'(x_2) \equiv 0\ (p).$$

Тут теж $f'(x_2)$ не кратне $p$, оскільки

$$x_2 \equiv x_1\ (p) \text{ і } f'(x_2) \equiv f'(x_1)\ (p),$$

і тому

$$t_2 \equiv t_2'\ (p) \text{ і } t_2 = t_2' + pt_3.$$

Вираз для $x$ набуває вигляду

$$x = x_2 + p^2 t_2' + p^3 t_3 = x_3 + p^3 t_3 \text{ і т. д.}$$

Таким чином, за заданим розв'язком конгруентності (2.22) поступово знайдемо розв'язок конгруентності (2.21). Отже, довільний розв'язок $x \equiv x_1\ (p)$ конгрунтності (2.22) за умови, що $f'(x_1)$ не кратне $p$, дає єдиний розв'язок конгруентності (2.21):

$$x = x_c + p^c t_c, \text{ де } x \equiv x_c\ (p^c). \blacksquare$$

**Приклад 2.5.17.** Розв'язати конгруентність $f(x) = x^4 + 7x + 4 \equiv 0\ (27)$.

*Розв'язання.* Конгруенція $f(x) \equiv 0\ (3)$ має один розв'язок $x \equiv 1\ (3)$ і при цьому $f'(1) \equiv 2\ (3)$. Отже, $f'(1)$ не кратне 3 і тоді знаходимо

$$x = 1 + 3t_1,\ f(1) + 3t_1 f'(1) \equiv 0\ (9) \text{ або } 3 + 3t_1 \cdot 2 \equiv 0\ (9),$$
$$2t_1 + 1 \equiv 0\ (3),\ t_1 \equiv 1\ (3),\ t_1 = 1 + 3t_2 \text{ і } x = 4 + 9t_2.$$
$$f(4) + 9t_2 f'(4) \equiv 0\ (27),\ 18 + 9t_2 \cdot 2 \equiv 0\ (27),\ 2t_2 + 2 \equiv 0\ (3),$$
$$t_2 \equiv 2\ (3),\ t_2 = 2 + 3t_3 \text{ і } x = 22 + 27t_3.$$

Отже, початкова конгруентність має єдиний розв'язок $x \equiv 22\ (27)$. ♠

Розглянемо **конгруентності другого степеня**, які мають вигляд

$$ax^2 + bx + c \equiv 0\ (m). \tag{2.23}$$

Домножуючи обидві частини (2.23) і модуль на $4a$, дістаємо конгруентність

$$4a^2x^2 + 4abx + 4ac \equiv 0\ (4am), \tag{2.24}$$

яка перетворюється до вигляду $(2ax+b)^2 \equiv b^2-4ac\ (4am)$, або, поклавши $2ax+b=y, b^2-4ac=d$, отримуємо конгруентність

$$y^2 \equiv d\ (4am). \tag{2.25}$$

Неважко показати, що коли відомий розв'язок конгруентності (2.23), завжди можна знайти розв'язок конгруентності (2.25) і навпаки. Дійсно, якщо відомий розв'язок (2.25) $y \equiv y_0\ (4am)$, то з конгруентності $2ax+b \equiv y_0\ (4am)$ знаходимо розв'язок (2.24) або (2.23) за модулем $m$.

Узагалі з рівності $2ax+b=y$ маємо $x=\frac{y-b}{2a}$, і якщо $y-b$ кратне $2a$, то дістаємо розв'язок $x$ конгруентності (2.23). Отже, серед розв'язків $y$ конгруентності (2.25) є такі, яким відповідають розв'язки $x$ конгруентності (2.23), і такі, яким не відповідають розв'язки $x$. Може статися, що кільком різним розв'язкам $y$ за модулем $4am$ відповідає один розв'язок $x$ за модулем $m$. Таким чином справедлива така теорема.

**Теорема 37.** *Конгруенцію другого степеня вигляду (2.23) завжди можна звести до двочленної конгруентності (2.25).*

З попереднього підрозділу випливає, що конгруентність вигляду (2.23) за складеним модулем завжди можна звести до такої за простим модулем. Тому можна обмежитися конгруентністю (2.23), де $m=p$ – просте число. А таку конгруентність можна подати у вигляді

$$x^2+2lx+c \equiv 0\ (p).$$

А ця конгруентність перетворюється до вигляду $y^2 \equiv d\ (p)$, де $y=x+l, d=l^2-c$.

**Приклад 2.5.18.** Звести конгруентність $4x^2-11x-8 \equiv 0 \pmod{13}$ до двочленної.

*Розв'язання.* Оскільки модуль просте число, то $\mathcal{Z}_{13}$ є полем і для елемента 4 існує обернений елемент 10, який неважко знайти розширеним алгоритмом Евкліда. Домножуючи обидві частини, конгруентність і модуль, на 10, дістаємо

$$x^2-6x-2 \equiv 0\ (13).$$

Отже, $l = -3, c = -2, d = 9 + 2 = 11$, і дістаємо $y^2 \equiv 11\ (13)$, де $y = x - 3$. ♠

Розглянемо умови існування розв'язку конгруентності

$$x^2 \equiv a\ (p), \tag{2.26}$$

де $p$ – непарне просте число і НСД$(a, p)$=1. Випадок $p = 2$ тривіальний і тому він не розглядається. Якщо $a \equiv 0\ (p)$, то очевидно єдиним її розв'язком буде $x \equiv 0\ (p)$. Тому далі $p$ буде непарним простим числом і $a$ – цілим числом, взаємно простим із $p$.

**Означення 49.** *Якщо конгруентність (2.26) має хоча б один розв'язок, то число $a$ називається* **квадратичним лишком**, *інакше $a$ називається* **квадратичним нелишком**.

Коли ж двочленна конгруентність має розв'язки? Відповідь на це питання дає теорема 38.

**Теорема 38 (критерій Ойлера).** *При простому $p$ і $a$, не кратному $p$, $a$ є квадратичним лишком тоді і тільки тоді, коли виконується конгруентність $a^{\frac{p-1}{2}} \equiv 1\ (p)$ і буде квадратичним нелишком тоді і тільки тоді, коли $a^{\frac{p-1}{2}} \equiv -1\ (p)$ [5].*

**Приклад 2.5.19.** Знайти розв'язок конгруентності $x^2 \equiv 12\ (19)$.
*Розв'язання.* Знаходимо

$$12^{\frac{19-1}{2}} = 12^9 = 12(12^2)^4 = 12 \cdot 11^4 = 12(11^2)^2 \equiv 12 \cdot 7^2 \equiv 12 \cdot 11 \equiv -1\ (\mathrm{mod}\ 19).$$

Отже, 12 – квадратичний нелишок і тому дана конгруентність не має розв'язку.

А конгруентність $x^2 \equiv 7\ (\mathrm{mod}\ 19)$ має розв'язок $x = 11$. ♠

### 2.5.9. Алгоритми обчислення функцій

**Алгоритми обчислення НСД**. Найбільш відомим алгоритмом обчислення НСД є алгоритм Евкліда. Цей алгоритм розглядався вище і тут подамо лише його варіації.

### Бінарний НСД-алгоритм

*Вхід:* натуральні числа $x, y$, де $x \geq y$.
*Вихід:* НСД$(x, y)$.
*Метод:*
1. $d := 1$;
2. while $x$ і $y$ are even do $(x := x/2;\ y := y/2; d := 2d)$;
3. while $x \neq 0$ do
   3.1. while $x$ is even do $x := x/2$;
   3.2. while $y$ is even do $y := y/2$;
   3.3. $t := |x - y|/2$;
   3.4. if $x \geq y$ then $x := t$ else $y := t$;
4. return $(d \cdot y)$.

**Приклад 2.5.20.** Застосуємо цей алгоритм до чисел $x = 1764$ і $y = 868$. Дістаємо таку послідовність для $x, y$ і $d$:

| Кроки | $x$ | $y$ | $d$ |
|---|---|---|---|
| крок 0 | 1764 | 868 | 1 |
| крок 1 | 441 | 217 | 4 |
| крок 2 | 112 | 217 | 4 |
| крок 3 | 7 | 217 | 4 |
| крок 4 | 7 | 105 | 4 |
| крок 5 | 7 | 49 | 4 |
| крок 6 | 7 | 21 | 4 |
| крок 7 | 7 | 7 | 4 |
| крок 8 | 0 | 7 | 4 |

Вихід: НСД(1764, 686) $= 4 \cdot 7 = 28$.

### Розширений бінарний НСД-алгоритм

знаходить числа $a, b, v$ такі, що $v =$ НСД$(x, y)$ і $v = ax + by$, де $a, b \in \mathcal{Z},\ v, x, y \in \mathcal{N}$.

*Вхід:* натуральні числа $x, y$.
*Вихід:* числа $a, b \in \mathcal{Z}$ такі, що $ax + by = v$, де $v =$НСД$(x, y)$.
*Метод:*
1. $d := 1$;
2. while $x$ and $y$ are even do
   $(x := x/2;\ y := y/2;\ d := 2d)$;

3. $u := x;\ v := y;\ A := 1;\ B := 0;\ C := 0;\ D := 1;$
4. while $u$ is even do
   4.1. $u := u/2;$
   4.2. if $A \equiv B \equiv 0 \pmod 2$ then $(A := A/2;\ B := B/2)$
   else $(A := (A + y)/2;\ B := (B - x)/2);$
5. while $v$ is even do
   5.1. $v := v/2;$
   5.2. if $C \equiv D \equiv 0 \pmod 2$ then $(C := C/2;\ D := D/2)$
   else $(C := (C + y)/2;\ D := (D - x)/2);$
6. if $u \geq v$ then $(u := u - v;\ A := A - C;\ B := B - D)$
   else $(v := v - u;\ C := C - A;\ D := D - B);$
7. if $u = 0$ then $(a := C;\ b := D;\ \text{return}(a, b, d \cdot v))$
   else goto 4.

Зауважимо, що обидва алгоритми мають складність в арифметичній моделі $O(\log^2 n)$ [12, 24].

**Приклад 2.5.21.** Застосуємо цей алгоритм до чисел $x = 693$ і $y = 609$. Маємо таку послідовність значень для $u, v, A, B, C$ і $D$:

| Кроки | $u$ | $v$ | $A$ | $B$ | $C$ | $D$ |
|---|---|---|---|---|---|---|
| крок 0 | 693 | 609 | 1 | 0 | 0 | 1 |
| крок 1 | 84 | 609 | 1 | -1 | 0 | 1 |
| крок 2 | 42 | 609 | 305 | -347 | 0 | 1 |
| крок 3 | 21 | 609 | 457 | -520 | 0 | 1 |
| крок 4 | 21 | 588 | 457 | -520 | -457 | 521 |
| крок 5 | 21 | 294 | 457 | -520 | 76 | -86 |
| крок 6 | 21 | 147 | 457 | -520 | 38 | -43 |
| крок 7 | 21 | 126 | 457 | -520 | -419 | 477 |
| крок 8 | 21 | 63 | 457 | -520 | 95 | -108 |
| крок 9 | 21 | 42 | 457 | -520 | -362 | 412 |
| крок 10 | 21 | 21 | 457 | -520 | -181 | 206 |
| крок 11 | 0 | 21 | 638 | -726 | -181 | 206 |

Вихід: $v = 21 = \text{НСД}(693, 609)$, $a = -181$, $b = 206$. ♠

### Алгоритм обчислення $a^d \pmod m$

*Вхід*: числа $a, d, m$.
*Вихід*: число $y = a^d \pmod m$.
*Метод*:
1. Записати число $d$ у двійковій системі числення $d = d_0 d_1 \cdots d_{r-1} d_r$;

2. $y := 1;\ s := a$;
3. for $i = 0, 1, \ldots, r$ do
   3.1. if $d_i = 1$ then $y := y \cdot s \pmod m$;
   3.2. $s := s \cdot s \pmod m$;
4. return($y$).

Доведення правильності цього алгоритму легко встановити за допомогою математичної індукції.

Оскільки кожне обчислення на кроці 3 потребує не більше трьох множень за модулем $m$ і цей крок виконується $r \le \log m$ разів, то складність алгоритму виражається величиною $O(\log^3 m)$.

**Приклад 2.5.22.** Результати проміжних обчислень показано в таблиці для $d = 100 = (1100100)_2$.

| $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| $d_i$ | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| $y$ | 1 | 1 | $a^4$ | $a^4$ | $a^4$ | $a^{36}$ | $a^{100}$ |
| $s$ | $a^2$ | $a^4$ | $a^8$ | $a^{16}$ | $a^{32}$ | $a^{64}$ | $a^{128}$ |

. ♠

У деяких випадках ефективнішим є алгоритм, в якому піднесення до степеня відбувається зліва направо.

### Другий алгоритм обчислення $a^d \pmod m$

*Вхід*: числа $a, d, m$.
*Вихід*: число $y = a^d \pmod m$.
*Метод*:
1. Записати число $d$ у двійковій системі числення $d = d_0 d_1 \cdots d_{r-1} d_r$;
2. $y := 1$;
3. for $i = r, r-1, \ldots, 0$ do
   3.1. $y := y \cdot y \pmod m$;
   3.2. if $d_i = 1$ then $y := y \cdot a \pmod m$;
4. return($y$).

Переконаємося в тому, що наведений алгоритм теж обчислює степінь $a^{100} \pmod m$.

**Приклад 2.5.23.** Результати проміжних обчислень показано в таблиці для $d = 100 = (1100100)_2$.

| $i$ | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| $d_i$ | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| $y$ | $a$ | $a^3$ | $a^6$ | $a^{12}$ | $a^{25}$ | $a^{50}$ | $a^{100}$ |

. ♠

**Обчислення оберненого $a^{-1}$ за модулем $n$.** Розв'язання конгруентності $a \cdot x \equiv 1\ (n)$ за умови НСД$(a, n) = 1$ еквівалентно задачі розв'язання в цілих числах рівняння $ax + ny = 1$.

### Алгоритм розв'язання рівняння $ax + ny = 1$

*Вхід*: числа $a, n$, де НСД$(a, n)$=1.
*Вихід*: число $a^{-1}$ таке, що $a \cdot a^{-1} \equiv 1\ (n)$.
*Метод*:
1. Застосувати розширений алгоритм Евкліда для обчислення чисел $x$ і $y$ таких, що $ax + ny = 1$.
2. return(x)

**Приклад 2.5.24.** Розв'язати рівняння $4x + 29y = 1$.

*Розв'язання.* Розширений алгоритм Евкліда знаходить значення $x = 22, y = -3$. Отже, $4^{-1} \equiv 22\ (29)$. ♠

Підводячи підсумок, наведемо таблицю складності обчислень арифметичних операцій у кільці цілих чисел $\mathcal{Z}$ і функцій у кільці лишків за модулем $n$, де числа подано у двійковій системі числення:

Операції в $\mathcal{Z}$, $a, b \le n$

| Операція | Складність |
|---|---|
| $a + b$ | $O(\log n)$ |
| $a - b$ | $O(\log n)$ |
| $a \cdot b$ | $O(\log^2 n)$ |
| $a = qb + r$ | $O(\log^2 n)$ |

,

Операції в $\mathcal{Z}_n$

| Операція | Складність |
|---|---|
| $(a + b)\ (\mathrm{mod}\ n)$ | $O(\log n)$ |
| $(a - b)\ (\mathrm{mod}\ n)$ | $O(\log n)$ |
| $(a \cdot b)\ (\mathrm{mod}\ n)$ | $O(\log^2 n)$ |
| $a^{-1}\ (\mathrm{mod}\ n)$ | $O(\log^2 n)$ |
| $a^d\ (\mathrm{mod}\ n)$ | $O(\log^3 n)$ |

.

**ρ-алгоритм Полларда** знаходить нетривіальний множник цілого числа $n$, яке не є степенем простого числа. Складність цього алгоритму в часі і пам'яті $O(\sqrt{n})$ [31].

**ρ-алгоритм**

*Вхід:* ціле число $n$, яке не є степенем простого числа.

*Вихід:* нетривіальний дільник числа $n$, якщо він існує.

*Метод:*

1. $a := 2;\ b := 2;$
2. for $i := 1, 2, \ldots$ do
   2.1. $a := a^2 + 1 \pmod n;\ b := b^2 + 1 \pmod n;\ b := b^2 + 1 \pmod n;$
   2.2. $d :=$НСД$(a - b, n)$;
   2.3. if $1 < d < n$ then (return($d$);STOP);
   2.4. if $d = n$ then (return("$d$ no exists");STOP).

**Приклад 2.5.25.** Застосування ρ-алгоритму до числа $n = 455459$. Результати обчислень показано в таблиці.

| $a$ | $b$ | $d$ |
|---|---|---|
| 5 | 26 | 1 |
| 26 | 2871 | 1 |
| 677 | 179685 | 1 |
| 2871 | 155260 | 1 |
| 44380 | 416250 | 1 |
| 179685 | 43670 | 1 |
| 121634 | 164403 | 1 |
| 155260 | 247944 | 1 |
| 44567 | 68343 | 743 |

.

Отже, двома нетривіальними дільниками числа 455459 є 743 і $455459/743 = 613$. ♠

**Алгоритм Поліга-Хеллмана** обчислює значення дискретного логарифма для відносно невеликих значень порядку мультиплікативної групи поля $\mathcal{F}_p$.

Нехай $n$ – порядок мультиплікативної групи і $n = p_1^{e_1} p_2^{e_2} \ldots p_r^{e_r}$ – канонічний розклад числа $n$ на прості множники. Якщо ж $x = \log_\alpha \beta$, то можна визначити $x_i \equiv x \ (p_i^{e_i})$ для $1 \le i \le r$, а потім скористатися алгоритмом Гаусса (див. формулу (2.13) в китайській теоремі про остачі) для знаходження $x \pmod n$. Кожне з $x_i$ визначається шляхом обчислення цілих чисел $l_0, l_1, \ldots, l_{e_i - 1}$ в системі числення за основою $p_i$, тобто обчислення зображення $x_i = l_0 + l_1 x + \ldots + l_{e_i - 1} p_i^{e_i - 1}$, де $0 \le l_j \le p_i - 1$.

**Алгоритм Поліга-Хеллмана**

*Вхід*: твірний елемент $\alpha$ групи $G$ порядку $n$ і довільний елемент $\beta \in G$.

*Вихід*: значення дискретного логарифма $x = \log_\alpha \beta$.
*Метод*:
1. Побудувати канонічний розклад числа $n = p_1^{e_1} p_2^{e_2} \dots p_r^{e_r}$;
2. for $i := 1$ to $r$ do (обчислення $x_i = l_0 + l_1 p_i + \dots + l_{e_i-1} p^{e_i-1}$, де $x_i \equiv x \ (p_i^{e_i})$)
   2.1. $q := p_i$; $e := e_i$;
   2.2. $\gamma := 1$; $l_{-1} := 0$;
   2.3. $\overline{\alpha} := \alpha^{n/q}$;
   2.4. for $j := 0$ to $e - 1$ do (обчислення $l_j$)
      2.4.1. $\gamma := \gamma \alpha^{l_{j-1} q^{j-1}}$; $\overline{\beta} := (\beta \gamma^{-1})^{n/q^{j+1}}$;
      2.4.2. $l_j := \log_{\overline{\alpha}} \overline{\beta}$ (для цього скористатися методом "крок гіганта, крок немовляти" (див пункт 2.10.4))
   2.5. $x_i := l_0 + l_1 q + \dots + l_{e-1} q^{e-1}$;
3. Користуючись алгоритмом Гаусса (із китайської теореми про остачі), обчислити ціле число $x$, де $0 \le x \le n - 1$, таке, що $x \equiv x_i \ (p_i^{e_i})$ для $1 \le i \le r$;
4. return($x$).

Аби переконатися в правильності алгоритму Поліга-Хеллмана зауважимо, що на кроці 2.3 порядок $\overline{\alpha}$ дорівнює $q$. Далі, на ітерації $j$ кроку 2.4 $\gamma = \alpha^{l_0 + l_1 q + \dots + l_{j-1} q^{j-1}}$. Отже,

$$\overline{\beta} = (\beta/\gamma)^{n/q^{j+1}} = (\alpha^{x - l_0 - l_1 q - \dots - l_{j-1} q^{j-1}})^{n/q^{j+1}} =$$
$$= (\alpha^{n/q^{j+1}})^{x_i - l_0 - l_1 q - \dots - l_{j-1} q^{j-1}} = (\alpha^{n/q^{j+1}})^{l_j q^j + \dots + l_{e-1} q^{j-1}} =$$
$$= (\alpha^{n/q})^{l_j + \dots + l_{e-1} q^{e-1-j}} = \overline{\alpha}^{l_j},$$

оскільки $\overline{\alpha}$ має порядок $q$. Таким чином, $\log_{\overline{\alpha}} \overline{\beta}$ дійсно дорівнює $l_j$.

Якщо відомий канонічний розклад числа $n$ на прості множники, то виконання алгоритму Поліга-Хеллмана потребує $O(\sum_{i-1}^{r} e_i(\log n + + \sqrt{p_i})$ групових множень. З наведеної оцінки випливає, що цей алгоритм ефективний тоді, коли всі прості множники відносно невеликі, а це означає що число $n$ є гладким.

**Означення 50.** *Нехай $p$ – ціле додатне число. Ціле число $n$ називається $p$-гладким, якщо всі його прості множники менші або дорівнюють $p$.*

Прикладом групи, в якій алгоритм Поліга-Хеллмана ефективний, є група $G_p$, де $p$ – число із 107 цифр:

$$p = 22708823198678103974314518195029102158525052496759285596453$$
$$269189798311427475159776411276642277139650833937.$$

Порядок групи $G_p$ дорівнює $n = p - 1 = 2^4 104729^8 224737^8 350377^4$. Оскільки найбільший простий множник $p - 1$ дорівнює 350377, то відносно легко обчислити дискретний логарифм у цій групі за допомогою алгоритму Поліга-Хеллмана.

Якщо число $n$ не є $p$-гладким, то алгоритм Поліга-Хеллмана стає неефективним.

**Приклад 2.5.26.** Нехай $p = 251$, $\alpha = 71$ – твірний елемент групи $G$, порядок якої $n = p - 1 = 250$. Візьмемо $\beta = 210$. Тоді $x = \log_{71} 210$ обчислюється алгоритмом Поліга–Хеллмана таким чином.

1. Канонічний розклад числа $250 = 2 \cdot 5^3$;
2. (Обчислюємо $x_1 \equiv x \ (2)$:)
   Обчислюємо $\overline{\alpha} = \alpha^{n/2} \pmod p = 250$ і $\overline{\beta} = \beta^{n/2} \pmod p = 250$.
   Тоді $x_1 = \log_{250} 250 = 1$.
   (Обчислюємо $x_2 \equiv x \ (5^3) = l_0 + l_1 5 + l_2 5^2$).
   Обчислюємо $\overline{\alpha} \equiv \alpha^{n/5} \pmod p = 20$.
   Обчислюємо $\gamma = 1$ і $\overline{\beta} = \beta^{n/5} \ (p) = 149$. Тоді $l_0 = \log_{20} 149 = 2$.
   Обчислюємо $\gamma = \gamma\alpha^2 \pmod p = 21$ і $\overline{\beta} = (\beta\gamma^{-1})^{n/25} \pmod p = 113$.
   Тоді $l_1 = \log_{20} 113 = 4$.
   Обчислюємо $\gamma = \gamma\alpha^{4,5} \pmod p = 115$ і $\overline{\beta} = (\beta\gamma^{-1})^{n/125} \pmod p = 149$.
   Тоді $l_2 = \log_{20} 149 = 2$.
   Отже, $x_2 = 2 + 4 \cdot 5 + 2 \cdot 562 = 72$.
3. Нарешті, розв'язуючи систему конгруентностей

$$\begin{aligned} x &\equiv 1 \ (2), \\ x &\equiv 72 \ (125), \end{aligned}$$

дістаємо $x = \log_{71} 210 = 197$. ♠

## 2.5.10. Алгоритми тестування чисел на простоту

Розглянемо декілька алгоритмів перевірки простоти цілого числа $n \in \mathcal{N}$.

**Метод пробних ділень** ґрунтується на твердженні 2 є), де говориться, що найменший простий дільник складеного числа $n = a \cdot b$, де $1 < a \leq b$, не більший $\sqrt{n}$. Звідси випливає спосіб тестування, який полягає в послідовному діленні числа $n$ на $d = 2, 3, \ldots, \lfloor\sqrt{n}\rfloor$. Якщо $n$ не ділиться на жодне із цих чисел, то воно просте число. У

протилежному випадку буде знайдений розклад $n$ на два множники.

Часова складність цього методу $O(\sqrt{n}\log^2 n)$ [31], оскільки складність виконання однієї операції ділення $O(\log^2 n)$, а всіх таких операцій не більше $\sqrt{n}$.

Цей метод можна модифікувати, якщо перевірити неподільність $n$ на 2 і 3. Якщо це так, то тоді перевірятимуться лише числа $d$ вигляду $6i+1$ і $6i+5$, де $i = 1, 2, \ldots$. Оцінка часової складності такої модифікації зменшується лише на константу.

**Решето Ератосфена** як метод полягає в тому, щоб у послідовності чисел $2, 3, 4, 5, \ldots, n$ виокремити всі прості числа. Ератосфен запропонував шукати ці числа шляхом викреслювання із цієї послідовності всіх чисел, які кратні 2, потім узяти перше не викреслене число, а це буде 3, і викреслити серед чисел, що залишилися, усі числа кратні 3, потім узяти перше невикреслене число, а це буде 5, і викреслити серед чисел, що залишилися, усі числа кратні 5 і т. д. У результаті виконання такої процедури залишаться лише прості числа. Для реалізації цього методу на комп'ютері потрібен великий об'єм пам'яті, але для побудови таблиць простих чисел цей метод на сьогодні є найкращим.

Програма реалізації решета Ератосфена на мові програмування ПАСКАЛЬ має вигляд [6]:

```
Procedure prime-num(n);
  type index = 1...n;
  var x, kvadrat : integer;   i, k, lim : index;   prim : boolean;
  p : array[index] of integer;   V : array[1..sqr(n)] of integer;
begin
  p[1] := 2; write(2); x := 1; lim := 1; kvadrat := 4;
  for i := 2 to n do
    repeat
      x := x + 2;
      if kvadrat ≤ x then
      begin
        V[lim] := kvadrat;   lim := lim + 1;
        kvadrat := sqr(p[lim]);
      end;
      k := 2; prim := true;
      while prim ∧ (k < lim) do
```

```
      if V[k] < x then V[k] := V[k] + p[k];
      prim := (x <> V[k]);   k := k + 1;
    od
  until prim;
  p[i] := x;   write (x);
 od
end
```

**Критерій Вільсона** перевірки простоти натурального числа ґрунтується на такому твердженні.

**Теорема 39.** *Число $n \in \mathcal{N}$ просте тоді і тільки тоді, коли* $(n-1)! \equiv -1 \ (mod\ n)$.

*Доведення.* У випадку $n = 2$ справедливість теореми очевидна. Якщо $n > 2$ просте число, то кожний елемент мультиплікативної групи поля $\mathcal{F}_n$, відмінний від 1 і $-1$ має обернений елемент $a^{-1}$, причому $a \neq a^{-1}$. Оскільки всіх елементів, серед яких немає 1, в цій групі буде $n-2$, то

$$(n-1)! \equiv (-1) \prod_{a \neq 1,-1} a a^{-1} \equiv n-1 \equiv -1 (\mathrm{mod}\ n).$$

Якщо число $n = ab$ складене, де $1 < a < n$, то $(n-1)!$ кратне $a$ і тому $(n-1)!$ не має оберненого елемента в кільці лишків $\mathcal{Z}_n$. Але тоді і $(n-1)! \not\equiv -1 \pmod{n}$. ■

Критерій Вільсона корисний у доведеннях тверджень, але застосовувати його для тестування простоти числа неефективно через велику часову складність обчислень.

**Тест на основі малої теореми Ферма**. Для тестування простоти чисел, порядок яких $10^{30}$–$10^{40}$, метод послідовних ділень стає неефективним. Тестування простоти таких чисел виконується на основі малої теореми Ферма. Згідно із цією теоремою, якщо число $n$ просте і НСД$(a, n) = 1$, де $a \in \mathcal{Z}$ – довільне ціле число, то виконується конгруентність:

$$a^n \equiv a(n), \text{ або } a^{n-1} \equiv 1\ (n). \tag{2.27}$$

Отже, для тестування простоти числа $n$ вибираємо довільне число $a \in \mathcal{Z}$ і перевіряємо виконання умов малої теореми Ферма за

$O(\log n)$ арифметичних операцій, які необхідні для виконання операції піднесення до степеня в кільці лишків $\mathcal{Z}_n$ за модулем $n$. Якщо умови теореми не виконуються, то число $n$ складене. У протилежному випадку вважати число $n$ простим ми не можемо, тому що мала теорема Ферма є лише необхідною умовою простоти. Отже, цей метод ефективний для знаходження складених чисел. Наприклад, для числа $n = 10^{99} + i$, де $i = 1, 3, 5, 7, \ldots$, виконання тесту $13^{n-1} \equiv 1 \pmod{n}$ дало такі результати: перші десять чисел, які успішно пройшли тест, виявилися простими. Ця перевірка виконувалася за допомогою такого алгоритму.

Якщо $n$ – просте число і $n - 1 = 2^s \cdot t$ , де $t$ – непарне, то згідно з малою теоремою Ферма для кожного $a$, такого, що НСД$(a, n)$=1, принаймні одна з дужок у добутку

$$(a^t - 1)(a^t + 1)(a^{2t} + 1) \cdots (a^{2^{s-1}t} + 1) = a^{n-1} - 1$$

ділиться на $n$. Контрапозицію цієї умови можна використати для того, щоб відрізнити складені числа від простих.

Нехай $n$ – непарне складене число і $n - 1 = 2^s \cdot t$, де $t$ – непарне. Ціле число $a$, де $1 < a < n$, називається "хорошим" для $n$, якщо порушується одна із таких двох умов:

$\alpha$) $n$ не ділиться на $a$;

$\beta$) $a^t \equiv 1 \pmod{n}$ або існує ціле число $k$, $0 \leq k < s$, таке, що $a^{2^k t} \equiv 1 \pmod{n}$.

Зі сказаного випливає, що для простого числа $n$ не існує хороших чисел $a$. Якщо ж число $n$ складене, то відомо, що таких чисел існує не більше $3/4(n - 1)$.

Звідси випливає простий імовірнісний алгоритм, який відрізняє складене число від простого.

**Алгоритм перевірки непростоти числа**
*Вхід*: непарне число $n$.
*Вихід*: "так" або "ні".
*Метод*:
1. Узяти випадкове число $a$, $1 < a < n$;
2. Перевірити виконання умов $\alpha$) і $\beta$);
3. Якщо хоча б одна з цих умов хибна, то return("так" – число складене);
4. Якщо умови $\alpha$) і $\beta$) виконуються, то перейти до кроку 1.

Імовірність виявлення непростоти числа $n$ із кожною ітерацією прямує до нуля, оскільки після $k$-ї ітерації вона має величину $4^{-k}$.

Згідно з наведеним алгоритмом достатньо перевірити умови $\alpha$) і $\beta$) для всіх цілих чисел $a$, де $2 < a \leq 70\ln^2 n$. Якщо для якого-небудь числа $a$ з указаного проміжку порушується одна з умов $\alpha$) чи $\beta$), то число $n$ складене. У протилежному випадку воно буде простим або буде степенем простого числа. Останнє перевіряється легко.

**Алгоритм побудови великого простого числа**. Найефективніший спосіб побудови простих чисел випливає з дещо модифікованої малої теореми Ферма.

**Теорема 40.** *Нехай $n$,$s$ – непарні натуральні числа і $n-1 = r\cdot s$, причому для кожного простого дільника $q$ числа $s$ існує ціле число $a$ таке, що*

$$a^{n-1} \equiv 1\ (n), \quad \text{НСД}(a^{\frac{n-1}{q}} - 1, n) = 1. \tag{2.28}$$

*Тоді кожний простий дільник числа $n$ задовольняє конгруентність $p \equiv 1\ (2s)$.*

*Доведення.* Нехай $p$ – простий дільник числа $n$, а $q$ – деякий дільник числа $s$. З умови (2.28) випливає, що в полі лишків $\mathcal{F}_p$ справедливі співвідношення

$$a^{n-1} = 1,\ a^{\frac{n-1}{q}} \neq 1,\ a^{p-1} = 1. \tag{2.29}$$

Нехай $r$ означає порядок елемента $a$ в мультиплікативній групі поля $\mathcal{F}_p$. Перші два співвідношення із (2.29) означають, що $q$ входить у розклад на прості множники числа $r$ у такому ж степені, як і в розклад числа $n-1$, а це означає що $p-1$ ділиться на $r$. Отже, кожний простий дільник числа $s$ входить у розклад числа $p-1$ у степені не меншому, ніж в $s$ так, що $p-1$ ділиться на $s$. Крім того, $p-1$ парне. ■

**Наслідок 5.** *Якщо умови теореми 40 виконані і $r \leq 4s+2$, то число $n$ – просте.*

Дійсно, нехай $n$ дорівнює добутку не менше двох простих чисел. Кожне із цих чисел на підставі теореми 40 не менше, ніж $2s+1$. Але тоді $(2s+1)^2 \le n = sr+1 \le 4s^2+2s+1$, а це суперечить нерівності $(2s+1)^2 \le 4s^2+2s+1$. ■

Опишемо тепер спосіб, як за допомогою теореми 40, маючи просте число $s$, можна побудувати суттєво більше просте число $n$.

Виберемо випадковим чином парне число $r$ із проміжку $s \le r \le 4s+2$ і покладемо $n = sr+1$. Перевіримо число $n$ на відсутність малих простих дільників, поділивши його на ці прості числа. Виконаємо цю перевірку декілька разів за допомогою алгоритму тестування непростоти числа. Якщо виявиться, що $n$ складене, то вибираємо нове число $r$ і знову повторюємо обчислення. Так діємо, доки не буде знайдене число $n$, яке витримало випробування алгоритмом достатню кількість разів. У цьому випадку появляється надія на те, що $n$ – просте число і цю простоту потрібно довести тестами теореми 40.

Із цією метою випадковим чином вибираємо число $a$, $1 < a < n$, і перевіряємо для нього виконання співвідношень

$$a^{n-1} \equiv 1 \ (n), \quad \text{НСД}(a^r - 1, n) = 1. \tag{2.30}$$

Якщо ці співвідношення виконуються для вибраного $a$, то на підставі наслідку 5 можна стверджувати, що число $n$ просте. Якщо ж ці співвідношення не виконуються, то вибираємо інше число $a$ і повторюємо вказані дії, доки таке число не буде знайдено.

Припустимо, що побудоване число $n$ дійсно просте. Скільки разів доведеться повторювати вибір числа $a$, доки не буде знайдено таке, для якого виконуються умови (2.30)? Зауважимо, що для простого числа $n$ перша умова (2.30) на підставі малої теореми Ферма виконуватиметься завжди. Числа $a$, для яких не виконується друга умова (2.30), задовольняють конгруентність $a^r \equiv 1 \pmod{n}$. А ця конгруентність у полі $\mathcal{F}_n$, як відомо, має не більше ніж $r$ розв'язків. Один із цих розв'язків $x = 1$. Тому на інтервалі $1 < a < n$ буде не більше $r-1$ чисел, які задовольняють умови (2.30). А це означає, що вибираючи випадковим чином число $a$ із проміжку $1 < a < n$ при простому $n$ можна знайти з імовірністю більшою ніж $1 - O(s^{-1})$ чи-

сло $a$, для якого виконуються умови (2.30) теореми 40 і тим самим довести, що $n$ дійсно просте число.

Отримане таким чином просте число $n$ буде задовольняти нерівність $n > s^2$, тобто буде записуватися з удвічі більшою кількістю цифр, ніж число $s$. Замінивши в цьому процесі число $s$ числом $n$ і повторивши всі описані дії із числом $n$, можна побудувати ще більше просте число. Якщо починати пошук з 10-розрядного числа (а його простоту можна легко перевірити шляхом ділення на табличні малі прості числа) і повторити описаний процес достатню кількість разів, то можна побудувати просте число потрібної величини.

Теоретичні підстави описаного процесу ґрунтуються на дослідженнях розподілу простих чисел в арифметичній прогресії $2sn+1$, де $n = 1, 2, \ldots$. У цій прогресії, як показав Діріхле в 1839 р., знаходиться нескінченно багато простих чисел. Нас цікавлять числа, які містяться недалеко від початку цієї прогресії. Відповідь на це питання було отримано Люстерніком в 1944 р., який показав, що найменше просте число в цій прогресії не перевищує величини $s^c$, де $c$ – досить велика константа.

Досвід обчислень на комп'ютерах показує, що прості числа в арифметичній прогресії зустрічаються досить близько від її початку і розміщені досить щільно. Існує гіпотеза Крамера, що відстань між сусідніми простими числами $p_n$ і $p_{n+1}$ в натуральному ряді чисел має оцінку $p_{n+1} - p_n = O(\ln^2 p_n)$, яка підтверджується й іншими результатами.

Отже, якщо припустити, що найменше просте число та відстань між сусідніми простими числами у прогесії $2sn + 1$, коли $s \leq n \leq 4s + 2$, оцінюється величиною $O(\ln^2 s)$, то описана схема побудови великого простого числа має поліноміальну оцінку часової складності.

**Тестування на простоту спеціальних чисел**. Серед цілих чисел виділяють деякі числа, які мають спеціальний вигляд.

***Числа Ферма***. Розглянемо числа, які мають вигляд $n = 2^m + 1$, де $m \in \mathcal{N}$. Якщо число $m$ ділиться на просте число $p > 2$, тобто $m = pm_1$, де $m_1 \geq 1$, то $n = (2^{m_1})^p + 1$ ділиться на $2^{m_1} + 1$ і тому є складеним. Отже, число $n = 2^m + 1$ може бути простим лише, коли

$m = 2^k$.

**Означення 51.** *Числа* $F_k = 2^{2^k} + 1$, *де* $k = 0, 1, 2, \ldots$, *називаються числами Ферма.*

Нині відомо, що числа $F_0, F_1, F_2, F_3, F_4$ – прості, а решта чисел Ферма, які вдалося перевірити, виявилися складеними. У загальному випадку справедлива така теорема.

**Теорема 41.** *Число* $n = F_k$, *де* $k > 0$, *просте тоді і тільки тоді, коли* $3^{\frac{n-1}{2}} \equiv -1 \ (mod\ n)$ [*3*].

Виконання тесту теореми потребує $O(\log n)$ арифметичних операцій, але числа Ферма дуже швидко зростають і всі методи тестування стають малоефективними.

***Числа Мерсенна***. Розглянемо числа вигляду $n = 2^m - 1$, де $m \in \mathcal{N}$. Якщо $m$ складене число, тобто $m = ab, 1 < a \leq b$, то $n = 2^{ab} - 1$ кратне $2^a - 1$. Тому число $n$ може бути простим лише, коли $m$ просте.

**Означення 52.** *Якщо число* $p$ – *просте і число* $M_p = 2^p - 1$ – *просте, то число* $M_p$ *називається числом Мерсенна.*

Числа Мерсенна досить рідко зустрічаються. У 2001 р. було знайдене тридцять дев'яте число Мерсенна – $M_{13466917}$.

Метод тестування чисел Мерсенна на простоту дає теорема 42.

**Теорема 42.** *Нехай* $q > 2$ – *просте число і* $n = 2^q - 1$. *Розглянемо послідовність* $L_0, L_1, L_2, \ldots$ *чисел, які визначаються співвідношеннями*

$$L_0 = 4,\ L_{i+1} = L_i^2 - 2 \ (mod\ n).$$

*Число* $n$ *просте тоді і тільки тоді, коли* $L_{q-2} \equiv 0 \ (mod\ n)$.

Наступна теорема висвітлює зв'язок між числами Мерсенна і числами Ферма.

**Теорема 43.** *Нехай* $p$ – *просте число, таке, що* $p \equiv 3 \ (mod\ 4)$ *і* $M_p = 2^p - 1$ – *число Мерсенна. Число Ферма* $F_p$ *просте тоді і тільки тоді, коли* $M_p^{\frac{F_p-1}{2}} \equiv -1 \ (mod\ F_p)$ [*3*].

## 2.6. Алгоритм RSA

Однією з основних реальних систем шифрування з відкритим ключем є система шифрування Рона Рівеста, Аді Шаміра і Лєна Адлемана (RSA-алгоритм) [36]. Ця система ґрунтується на теорії чисел і вважається досить надійною.

Система RSA є блоковим шифром, в якому відкритий і зашифрований тексти є цілими числами від 0 до $n-1$ для даного $n \in \mathcal{N}$.

### 2.6.1. Основи алгоритму

**Алгоритм.** Система RSA використовує показникову функцію. Явний текст шифрується блоками, кожний з яких є бінарним значенням, яке менше від заданого числа $n$. Шифрування і розшифрування для блока відкритого тексту $M$ і блока зашифрованого $Z$ має такий вигляд:

$C \equiv M^e \ (n)$

$M \equiv C^d \equiv (M^e)^d \equiv M^{ed} \ (n)$.

Надавач повідомлення, як і отримувач, мусять знати значення $n$. Надавач знає значення $e$, а отримувач знає єдине значення $d$. На цьому і ґрунтується алгоритм шифрування з відкритим ключем $KU = \{e, n\}$ і ключем приватним $KR = \{d, n\}$. Аби такий алгоритм був придатний до шифрування, він повинен задовольняти такі вимоги:

1. Можливість обчислення для кожного $M < n$ таких значень $e, d, n$, що $M^{ed} \equiv M \ (n)$;

2. Обчислення $M^e$ і $C^d$ для всіх значень $M < n$ повинні бути відносно простими;

3. Обчислення $d$ при відомих $e$ і $n$ повинно бути невиконуваним.

Спочатку сконцентруємося на першій вимозі, а потім розглянемо інші вимоги. Необхідно знайти залежність вигляду

$M^{ed} \equiv M \pmod{n}$.

За наслідком теореми Ойлера маємо: при заданих двох простих числах $p$ і $q$ і двох цілих числах $m$ і $n$ таких, що $n = pq$ і $0 < m < n$, і довільному цілому $k$, виконується залежність

$$m^{k\varphi(n)+1} = m^{k(p-1)(q-1)+1} \equiv m \pmod{n},$$

де $\varphi(n)$ – функція Ойлера.

З теореми Ойлера випливає, що для простих чисел $p, q$ значення функції Ойлера обчислюється просто $\varphi(pq) = (p-1)(q-1)$. Отримаємо потрібну залежність тоді, коли

$$ed = k\varphi(n) + 1,$$

що еквівалентно

$$ed \equiv 1 \pmod{\varphi(n)} \quad \text{і} \quad e \equiv d^{-1} \pmod{\varphi(n)}.$$

А це означає, що $e$ і $d$ взаємно обернені відносно mod $\varphi(n)$. На підставі правил модульної арифметики, це буде правильно тоді, коли $d$ (і, отже, $e$) і $\varphi(n)$ взаємно прості, тобто НСД$(\varphi(n), d) = 1$.

Тепер можна описати систему RSA. Її складовими є такі числа:

| | |
|---|---|
| $p, q$ два простих числа | (приватні, вибираються) |
| $n = pq$ | (явне, обчислюється) |
| $d$ при НСД$(\varphi(n), d) = 1$; $1 < d < \varphi(n)$ | (приватне, обчислюється) |
| $e \equiv d^{-1} \pmod{\varphi(n)}$ | (явне, вибирається) |

Ключ приватний складається із чисел $\{d, n\}$, а ключ відкритий із чисел $\{e, n\}$. Нехай користувач $A$ опублікував свій відкритий ключ, а користувач $B$ хоче вислати повідомлення $M$ до $A$. $B$ обчислює $C \equiv M^e \pmod{n}$ і висилає $C$ до $A$. Отримавши зашифрований текст, користувач $A$ дешифрує його, обчислюючи $M \equiv C^d \pmod{n}$.

Покажемо правильність цього алгоритму. Нехай вибрані числа $e$ і $d$ такі, що

$$e \equiv d^{-1}(\varphi(n)).$$

Тоді

$$ed \equiv 1(\varphi(n)),$$

де $ed$ має вигляд $k\varphi(n) + 1$. Згідно з наслідком з теореми Ойлера при даних двох простих числах $p$ і $q$ та цілих числах $n = pq$ і $M$, де $0 < M < n$:

$$M^{k\varphi(n)+1} \equiv M^{k(p-1)(q-1)+1} \equiv M(\text{mod } n).$$

На рис. 2.6.1 алгоритм RSA зображено у скороченому вигляді, а на рис. 2.6.2 наведено приклад його використання. У прикладі генерація ключів відбувається таким чином:

1. Вибрати два простих числа, наприклад $p = 7, q = 17$.

2. Обчислити $n = pq = 7 \cdot 17 = 119$.

3. Обчислити $\varphi(n) = (p-1)(q-1) = 96$.

4. Вибрати $e$ таке, що $e$ і $\varphi(n) = 96$ взаємно прості і такі, що $e$ менше ніж $\varphi(n)$; у нашому випадку $e = 5$.

5. Обчислити $d$ таке, що $de \equiv 1(\text{mod } 96)$ і $d < 96$. Знайдене значення $d = 77$, оскільки $77 \cdot 5 = 385 = 4 \cdot 96 + 1$.

Отримані ключі: ключ відкритий $KU = \{5, 119\}$ і ключ приватний $KR = \{77, 119\}$. Тепер застосуємо ці ключі до вхідних даних в явному вигляді: нехай $M = 19$. Під час шифрування 19 підноситься до п'ятого степеня, що дає значення 2476099. Після цього ділимо його на 119, отримуючи остачу 66. Отже, $19^5 \equiv 66(\text{mod } 119)$ і зашифрованим текстом є 66. Отримавши криптограму, отримувач обчислює, $66^{77} \equiv 19(\text{mod } 119)$.

| | |
|---|---|
| **Генерація ключа** | |
| Вибрати $p, q$ | $p$ і $q$ прості числа |
| Обчислити $n = pq$ | |
| Вибрати ціле число $d$ | НСД$(\varphi(n), d) = 1,\ 1 < d < \varphi(n)$ |
| Обчислити $e$ | $e \equiv d^{-1}(\text{mod } \varphi(n))$ |
| Явний ключ | $KU = \{e, n\}$ |
| Приватний ключ | $KR = \{d, n\}$ |
| **Шифрування** | |
| Явний текст | $M < n$ |
| Криптограма | $C \equiv M^e(\text{mod } n)$ |
| **Розшифрування** | |
| Зашифрований текст | $C$ |
| Явний текст | $M \equiv C^d(\text{mod } n)$ |

Рис. 2.6.1. Алгоритм RSA

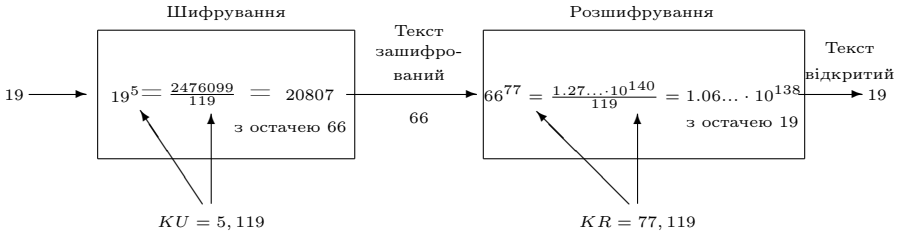



Рис. 2.6.2. Приклад роботи алгоритму RSA

### 2.6.2. Складність обчислень

Визначимо складність обчислень, які виконуються при застосуванні алгоритму RSA. Дві основні операції це – генерація ключів і шифрування. Розглянемо спочатку процес шифрування і розшифрування, потім повернемося до генерації ключів.

**Шифрування і розшифрування.** Як при шифруванні, так і при розшифруванні виконується операція піднесення цілого числа до цілого степеня. Властивості модульної арифметики дають можливість виконувати ці операції досить ефективно. Наприклад,

$$[(a \bmod n) \cdot (b \bmod n)] \pmod n = (a \cdot b) \pmod n.$$

У такий спосіб вдається спростити обчислення за модулем $n$, завдяки чому вони стають реальними.

Під час обчислень степеня і модуля теж виникає проблема ефективності обчислень. У системі RSA можна скористатися тими самими алгоритмами виконання цих операцій, які були описані вище.

**Генерація ключа.** Перед застосуванням системи з відкритим ключем, кожний користувач повинен згенерувати пару ключів. У зв'язку із цим виникає завдання:

- знайти два простих числа $p$ і $q$;
- вибрати одне із чисел $e$ або $d$ і обчислити друге.

Спочатку займемося вибором простих чисел $p$ і $q$. Оскільки значення $n = pq$ може стати відоме потенціальному зловмиснику, то щоб запобігти обчисленню чисел $p$ і $q$, вони повинні вибиратися з

великої множини таких чисел (отже, $p$ і $q$ мусять бути великими числами). Для цього можна застосувати алгоритм побудови вказаних чисел, який розглядався вище і який є практичним.

Розглянемо ще один алгоритм побудови таких чисел. Спочатку нагадаємо, що коли число просте, то жодне інше число не є його дільником. Простими числами є: 2, 3, 5, 7, 11, 13,..., 73, 2521, 2365347734339 а також $2^{756839}-1$. Метод перевірки простоти числа випливає з такої теореми.

**Теорема 44.** *Якщо непарне число $p$ просте, то конгруентність* $x^2 \equiv 1 \pmod{p}$ має тільки два розв'язки, а саме: $x = \pm 1$.

*Доведення.* Конгруентність $x^2 - 1 \equiv 0 \pmod{p}$ еквівалентна конгруетності $(x-1)(x+1) \equiv 0 \pmod{p}$. На підставі властивостей модульної арифметики остання конгруентність виконується, якщо $p$ є дільником $(x-1)$ або $(x+1)$ або обох цих чисел. Нехай $p$ – дільник $(x+1)$ і $(x-1)$. Тоді можна стверджувати, що для деяких цілих чисел $k$ і $j$, $x+1 = kp$ і $x-1 = jp$. Почленно віднімаючи ці рівності, дістаємо $2 = (k-j)p$. А це рівняння може виконуватися лише для $p = 2$. За умовами теореми нас цікавлять тільки непарні прості числа. Але тоді або $x$ або $p$ ділиться на $x+1$ або $p$ ділиться на $x-1$ (але не на кожне із цих чисел). Припустимо, що $p$ ділиться на $x-1$. Тоді $x-1 = kp$ для деякого $k$. А звідси випливає, що $x \equiv 1 (\mathrm{mod}\ p)$. Аналогічно доводиться і другий випадок, коли $x \equiv -1 (\mathrm{mod}\ p)$. ■

Обернене твердження до цієї теореми має вигляд: якщо конгруентність $x^2 \equiv 1 (\mathrm{mod}\ n)$ має розв'язки, відмінні від розв'язків $\pm 1$, то $n$ не є простим числом.

Як говорилося в пункті 2.5.9, багато методів тестування простоти числа носять імовірнісний характер. Це означає, що вони тестують ступінь правдоподібності простоти числа. Розглянемо ще один з таких методів, який є досить ефективним. Це алгоритм Міллера-Рабіна [33, 35].

**Алгоритм WITNESS($a, n$).**
**begin**
1. let $b_k b_{k-1} \cdots b_0$ – binary representation of $n-1$;

2. $d := 1$;
3. for $i = k$ downto 0 do
  3.1. $x := d$;  $d := (d \cdot d) \pmod n$;
  3.2. if $(d = 1) \wedge (x \neq 1) \wedge (x \neq n - 1)$ then return(TRUE);
  3.3. if $b(i) = 1$ then $d := (d \cdot a) \pmod n$
 od
4. if $(d \neq 1)$ then return(TRUE) else return(FALSE);

**end**

Ефективність алгоритму RSA ґрунтується на ефективності алгоритму тестування простоти заданого числа $n$. Використовується число $n$ і випадкове ціле число $a$. Якщо $n$ "не складає" тесту, то воно не є простим числом. Якщо ж "складає", то воно може бути як простим числом, так і складеним. Коли ж воно проходить багато тестувань із багатьма випадково вибраними числами $a$, то можна стверджувати, що $n$ дійсно просте число.

У скороченому вигляді ймовірнісна процедура тестування простоти випадкового числа має вигляд [26]:

1. Вибрати випадкове ціле число $n$, наприклад за допомогою генератора випадкових чисел.

2. Вибрати випадкове ціле число $a < n$.

3. Виконати ймовірнісний тест, який перевіряє, чи є число $n$ простим, наприклад, за допомогою алгоритму Міллера–Рабіна. Якщо $n$ не складає тест, то відкинути його і перейти на крок 1.

4. Якщо $n$ склало тест достатню кількість разів, узяти його і перейти до виконання кроку 2.

Процедура досить вичерпна. Але належить пам'ятати, що виконується вона відносно зрідка – тільки тоді, коли існує потреба вибору нової пари ключів $(KU, KR)$.

З теорії чисел відомо, яка кількість чисел буде перевірена, доки не буде знайдено просте число. Ця кількість не перевищує величини $\ln n$. А звідси отримуємо, що необхідно перевірити $\ln n$ цілих чисел. Реальна кількість проб не перевищує $\ln(n/2)$. Наприклад, якщо б ми шукали просте число порядок якого $2^{200}$, то для його знаходження необхідно було б виконати близько $\ln\lfloor 2^{200}/2\rfloor = 70$ проб.

Якщо прості числа $p$ і $q$ вибрані, то процес генерування ключів

вимагає вибору значень $d$ і обчислення $e$ або (друга можливість) вибору значення $e$ і обчислення $d$. Припускаючи, що маємо першу ситуацію, мусимо вибрати таке $d$, що НСД$(\varphi(n), d) = 1$, а потім обчислити $e \equiv d^{-1} \pmod{\varphi(n)}$. За допомогою розширеного алгоритму Евкліда ця задача розв'язується досить ефективно. Така процедура зводиться до генерації серії випадкових чисел і їхнього тестування відносно функції $\varphi(n)$, доки не буде знайдене просте число, яке взаємно просте з $\varphi(n)$. Знову можна задати питання: скільки випадкових чисел потрібно випробувати, щоб знайти число, взаємно просте з $\varphi(n)$? Відомо, що правдоподібність того, що два випадкових числа будуть взаємно простими, дорівнює 0,6, а тому для знаходження таких чисел потрібно виконати не так багато перевірок.

### 2.6.3. Криптоаналіз алгоритму RSA

У криптоаналізі алгоритму RSA можна виокремити чотири можливі підходи:

1. Брутальний або метод повного перебору: випробовувати всі можливі приватні ключі.

2. Розкласти $n$ у добуток двох простих чисел $p$ і $q$. Це дає можливість обчислення $\varphi(n) = (p-1)(q-1)$, а тому стає можливим обчислення $d \equiv e^{-1} (\mathrm{mod}\ \varphi(n))$.

3. Знайти $\varphi(n)$ безпосередньо, без пошуку чисел $p$ і $q$. Завдяки цьому стає можливим обчислення $d \equiv e^{-1} \pmod{\varphi(n)}$.

4. Знайти $d$ безпосередньо, без обчислення $\varphi(n)$.

Оборона перед атакою брутальним методом у випадку RSA є такою: використання великого простору вибору ключів. Чим більшим є число бітів у зображенні $e$ і $d$, тим краще.

Більшість досліджень на тему криптоаналізу алгоритму RSA концентрується на проблемі розкладу числа $n$ на два прості множники. На сьогоднішній день не відомі прийнятні алгоритми, які розв'язують цю проблему для великих значень, наприклад, з кількома сотнями знаків. Найефективніші алгоритми знаходять розклад цілого числа $n$ на прості множники в часі:

$$t(n) = e^{\ln^{\frac{1}{3}} n} (\ln \ln n)^{\frac{2}{3}} (c + o(1)).$$

Сучасний рівень комп'ютерної техніки показує, що 100-цифрове число можна розкласти на прості множники за два тижні. У випадку більшого числа зі 150 цифрами розклад відбувається протягом року. А число з 200 цифрами взагалі неможливо розкласти в розумному проміжку часу, хіба що наступить корінна зміна в теорії чисел. Наприклад, якщо буде досягнута ефективність $10^{12}$ операцій за секунду, що не є реальним для сучасної комп'ютерної техніки, то це вимагатиме часу, порядок якого 1000 років. Для ефективного використання RSA прийнятним є розмір 154 цифр, який вміщується в 512 бітах, у той час як 200-цифрове число потребує 600 бітів, а це не є прийнятним.

Крім вимоги, що $n$ повинно мати порядок від $10^{150}$ до $10^{200}$, дослідники запропонували багато інших обмежень. Щоб зробити пошук таких значень числа $n$ ефективнішим, винахідники алгоритму запропонували такі обмеження для чисел $p$ і $q$:

1. $p$ і $q$ повинні відрізнятися довжиною не більшою, ніж кілька цифр. Як $p$, так і $q$ повинні мати значення, порядок яких від $10^{75}$ до $10^{100}$.

2. Як $(p-1)$, так і $(q-1)$ повинні мати великий простий множник.

3. НСД$(p-1, q-1)$ повинен бути невеликим.

Крім того, якщо $e < n$ і $d < n^{1/4}$, то можна легко обчислити $d$.

На закінчення ще розглянемо приклад застосування алгоритму RSA.

**Приклад 2.6.1**. Зашифруємо повідомлення "CAB". Скористаємося малими простими числами, але в застосуваннях ці числа мають бути великими.

1. Візьмемо $p = 3,\ q = 11$.

2. Обчислимо $n = 3 \cdot 11 = 33$.

3. Обчислимо $(p-1)(q-1) = 20$. Отже, $d$ може бути довільним числом взаємно простим із 20, наприклад, $d = 3$.

4. Візьмемо число $e$, яке задовольняє умові $e \cdot 3 \equiv 1 \pmod{20}$, наприклад $e = 7$.

5. Запишемо повідомлення у вигляді послідовності цілих чисел за допомогою відображення $A \to 1,\ B \to 2,\ C \to 3$. Тоді повідомлення матиме вигляд (3,1,2). Зашифруємо повідомлення за допомогою ключа $\{7, 33\}$:

SZ1 $= 3^7 (\text{mod } 33) = 2187 (\text{mod } 33) = 9$,

SZ2 $= 1^7 (\text{mod } 33) = 1 (\text{mod } 33) = 1$,

SZ3 $= 2^7 (\text{mod } 33) = 128 (\text{mod } 33) = 29$.

Отже, зашифроване повідомлення має вигляд (9,1,29).

6. Розшифрування повідомлення виконуємо за допомогою ключа $\{3, 33\}$. Дістаємо

такий текст:

$\mathrm{DSZ1} = 9^3 (\mathrm{mod}\ 33) = 729 (\mathrm{mod}\ 33) = 3,$
$\mathrm{DSZ2} = 1^3 (\mathrm{mod}\ 33) = 1 (\mathrm{mod}\ 33) = 1,$
$\mathrm{DSZ3} = 29^3 (\mathrm{mod}\ 33) = 24389 (\mathrm{mod}\ 33) = 2.$

А це означає, що текст відкритий є "CAB". У цьому прикладі пара $\{e = 7, n = 33\}$ утворює ключ відкритий, а пара $\{d = 3, n = 33\}$ – ключ таємний. ♠

Зауважимо, що алгоритм RSA використовується у багатьох стандартах: SSl, S-HHTP, S-MIME, S/WAN, STT і PCT.

## 2.7. Криптосистеми поділу секрету

У розглянутих вище прикладах було два або три учасники процесу обміну повідомленнями. А як бути в ситуації, коли число користувачів відносно велике і необхідно зберігати секретність?

Відповідь на це питання пов'язана із **задачею поділу секрету**. Формулювання цієї задачі є таким.

Нехай у криптосистемі є $t > 2$ легальних користувачів $A_i$, $i = 1, 2, \ldots, t$. Говорять, що ці користувачі виконують $k$-зберігання секрету $C$, $1 \le k \le t$, якщо виконуються такі три умови.

1. Кожний $A_i$ знає лише свою інформацію $a_i$, яка невідома іншим користувачам.

2. Секрет $C$ можна легко обчислити, якщо відомі довільні $k$ секретів $a_i$.

3. Знання довільних $k - 1$ секретів $a_i$ не дає можливості обчислити секрет $C$.

Множина $\{a_1, a_2, \ldots, a_t\}$, яка задовольняє ці умови, називається $(k, t)$-*пороговою схемою.*

Порогові схеми будують на основі модульної арифметики з обчисленням (відновленням) інформації за допомогою китайської теореми про остачі.

Розглянемо найпростіший спосіб поділу секрету. Для цього виберемо сукупність попарно взаємно простих модулів $\{m_1, m_2, \ldots, m_t\}$ і цілі числа $a_i$, $i = 1, 2, \ldots, t$, які задовольняють умові

$$1 \le a_i < m_i. \tag{2.31}$$

Нехай $M = \prod_{i=1}^{t} m_i$ і

$$M_i = \frac{M}{m_i}. \tag{2.32}$$

Нехай $N_i$ число, обернене до $M_i$ за модулем $m_i$, $i = 1, 2, \ldots, t$, тобто $M_i N_i \equiv 1 \pmod{m_i}$. Система конгруентностей для (2.31)

$$x \equiv a_i \ (m_i)$$

матиме єдиний розв'язок за модулем $M$, який знаходиться як

$$x \equiv \sum_{i=1}^{t} a_i M_i N_i(M). \tag{2.33}$$

Нехай тепер $k$ – фіксоване, де $1 \leq k \leq t$. Позначимо $M_1$ найменший добуток $k$ різних модулів і розмістимо ці модулі у зростаючому порядку $M_1 = m_1 \cdot m_2 \cdots m_k$. Нехай $M_2$ означає найбільший добуток $k-1$ модулів. Модулі слід вибирати так, щоб різниця $M_1 - M_2$ була велика. Секрет $C$ мусить задовольняти нерівності $M_2 < C < M_1$. Тепер як секрети $a_i$ користувачів $A_i$ беруть найменші невід'ємні лишки секрету $C$ за модулем $m_i$ так, що

$$a_i \equiv C \pmod{m_i}, \ i = 1, 2, \ldots, t.$$

**Приклад 2.7.1**. Нехай маємо модулі $m_i$, $i = 1, 2, 3, 4, 5$, значення яких наведені нижче в таблиці. Тоді частина секрету $a_i$ $i$-го користувача $A_i$ визначається найменшим невід'ємним лишком секрету $C$ за модулем $m_i$.

| $m_1 = 97$ | $m_2 = 98$ | $m_3 = 99$ | $m_4 = 101$ | $m_5 = 103$ |
|---|---|---|---|---|
| $a_1 = 73$ | $a_2 = 15$ | $a_3 = 61$ | $a_4 = 61$ | $a_5 = 49$ |

.

Перевіримо коректність (3,5)-порогової схеми. Обчислимо $M_1$ – найменший добуток $k = 3$ різних модулів: очевидно,

$$M_1 = 97 \cdot 98 \cdot 99 = 941094.$$

Обчислимо $M_2$ – найбільший добуток $k - 1 = 3 - 1 = 2$ модулів:

$$M_2 = 101 \cdot 103 = 10403.$$

Модулі вибирають так, щоб різниця була велика. У нашому випадку $M_1 - M_2 \geq \geq 3M_2$, оскільки

$$M_1 - M_2 = 941094 - 10403 = 930691 \geq 3 \cdot 10403 = 31209.$$

Секрет $C$ вибираємо з інтервалу $(M_2, M_1)$ і обчислюємо

$$M_1 = \frac{M}{m_1} = \frac{941094}{97} = 98 \cdot 99 = 9702,$$
$$M_2 = \frac{M}{m_2} = \frac{941094}{98} = 97 \cdot 99 = 9603,$$
$$M_3 = \frac{M}{m_3} = \frac{941094}{99} = 97 \cdot 98 = 9506.$$
$$N_1 \equiv 49 \pmod{97},\ N_2 \equiv 97 \pmod{98},\ N_3 \equiv 50 \pmod{99}.$$

Знаходимо секрет $C$:

$$C = \sum_{i=1}^{3} a_i M_i N_i \pmod{M} = 73 \cdot 9702 \cdot 49 + 15 \cdot 9603 \cdot 97 + 61 \cdot 9506 \cdot 50 \equiv 50001 \pmod{941094}.$$

Це і є початковий секрет $C = 50001$. ♠

### Контрольні запитання

1. Навести означення дільника, простого та складеного числа.
2. Навести означення відношення подільності та сформулювати основні властості цього відношення.
3. Сформулювати основні властивості відношення подільності на множинах $\mathcal{N}$ і $\mathcal{Z}$. Чи буде це відношення частковим порядком на цих множинах?
4. Навести означення спільного дільника та НСД двох чисел.
5. Сформулювати основні властивості НСД.
6. Сформулювати основну теорему арифметики. Який розклад на прості множники називається канонічним?
7. Дати означення функції Ойлера $\varphi(n)$ і написати формулу для обчислення її значень.
8. Дати означення чисел Ферма і сформулювати їхні властивості.
9. Дати означення чисел Мерсенна і сформулювати їхні властивості.
10. Які проблеми пов'язані з тестуванням чисел Ферма і Мерсенна на простоту?

### Задачі і вправи

1. Довести, що коли
   а) $mn + pq$ кратне $m - p$, то $mq + np$ кратне $m - p$;
   б) цілі числа $a, b, c, d, n$ задовольняють умови НСД$(b, n)$=1, $ad - bc$ кратне $n$, $a - b$ кратне $n$, то $c - d$ кратне $n$.
2. За допомогою алгоритму Евкліда знайти НСД чисел
   а) 42628 і 33124; б) 299, 391, 667;
   в) 1955, 2431; г) 3111, 4862. (*Відп.* а) 4; б) 23; в) 17).

   Довести, що
   а) НСД$(m + nk, n)$ = НСД$(m, n)$,
   б) НСД$(f_{k+2} + f_{k+1})$ = НСД$(f_1, f_2)$, де $f_i$ – $i$-те число ряду Фібоначчі,
   в) НСД$(m, n, k)$ = НСД(НСД$(m, n), k)$ = НСД($m$,НСД$(n, k)$),

г) коли натуральне число $n$ не кратне 7, то $n^3-1$ або $n^3+1$ кратне 7,

д) коли натуральне число $n>1$ не кратне 2 і 3, то $n^2-1$ кратне 24.

3. Знайти канонічний розклад чисел

а) 82 798 848; б) 4 497 552 259 200;

в) 67 463 283 888 000; г) 2013 2014 2015. (*Відп.* а) $2^8 3^5 11^3$; б) $2^7 3^3 5^2 7^2 11\cdot 13\cdot 17\cdot 19\cdot 23$; в) $2^6 3^4 5^2 7^2 11\cdot 13\cdot 17\cdot 19\cdot 23$).

4. Розв'язати рівняння $\lfloor 2ax\rfloor = m$, де $a\neq 0$ і $x$ – дійсні числа. (*Відп.* $x=\frac{m+\alpha}{2a}$, де $\alpha$ – дійсне число, що задовольняє умові $0\leq\alpha<1$).

5. Знайти значення числа $m$, якщо

а) $\lfloor 32,6\cdot m\rfloor = 97$;

б) $\lfloor 27,4\cdot m\rfloor = 140$. (*Відп.* а) $m=3$; б) такого $m$ не існує).

6. Довести, що для довільного дійсного числа $x$ справедлива рівність $\lfloor x\rfloor+\lfloor x+\frac{1}{2}\rfloor = \lfloor 2x\rfloor$.

7. Знайти канонічний розклад чисел

а) 40!; б) 50!; в) 60!; г) 75!.

8. Знайти кількість дільників чисел

а) 4520; б) 27 504; в) 116 424; г) 1 002 001; д) 1 294700.

9. Знайти всі натуральні дільники числа 4520.

10. Знайти найменше натуральне число, яке має 15 дільників.

11. Знайти значення $\varphi(2429)$.

12. Знайти кількість натуральних чисел, які не більші за 1327 і не діляться на жодне з простих чисел 5,7,13.

13. Знайти кількість натуральних чисел, менших $n$ і взаємно простих з $n$, якщо

а) $n=3560$; б) $n=4520$; в) $n=116424$; г) $n=1002001$; д) $n=1294700$;

е) $n=1294699$. (*Відп.* а) 1408; б) 1792; в) 30 240; г) 720; д) 720; е) 466 400).

14. Довести, що коли

а) $n=m\cdot p$, де НСД$(m,p)=p$ і $p$ – просте число, то $\varphi(n)=\varphi(m\cdot p)=\varphi(m)p$;

б) $n=m\cdot p$, НСД$(m,p)=1$ і $p$ – просте число, то $\varphi(n)=\varphi(m\cdot p)=\varphi(m)(p-1)$.

15. Користуючись результатом попередньої задачі, вивести рекурентну формулу за умови НСД$(m,p)$=1: $\varphi(mp^k)=\varphi(m)p^{k-1}(p-1)$.

16. Довести, що $\varphi(4n)=2\varphi(2n)$ і $\varphi(4n+2)=\varphi(2n+1)$.

17. Дано $\varphi(11^n)=13310$. Знайти $n$. (*Відп.* $n=4$).

18. Дано $\varphi(n)=1792$ і $n=2^x5^y113^z$. Знайти $n$. (*Відп.* $n=2^3 5\cdot 113$).

19. При якому натуральному $n$ справедлива рівність $\varphi(n)=\frac{1}{3}n$? (*Відп.* $n=2^k3^l$, де $k,l>0$).

20. Довести, що рівність $\varphi(n)=\frac{1}{4}n$ неможлива.

21. Довести *тотожність Гаусса*: $\varphi(n)=\sum\limits_{d|n}\varphi(d)=n$.

22. Знайти найменші невід'ємні лишки за модулем 13 чисел 3,8,16, $-43$, 132,278,-423, 1327. Які із цих чисел належать до одного і того самого класу за модулем 13? (*Відп.* 3,8,3,9,2,5,6,1. Отже, 3 і 16 належать до одного класу).

23. Знайти зведену систему найменших невід'ємних лишків за модулем 40. (*Відп.* 1,3,7,9,11,13,17,19,21,23,27,29,31,33,37,39).

24. Чи утворюють степені $2^0,2^1,2^2,2^3,2^4,2^5,2^6,2^7,2^8,2^9$ разом із числом 0, повну систему лишків за модулем 11? (*Відп.* Так).

25. Перевірити справедливість конгруентностей: $5^{\varphi(26)}\equiv 1(26), 2^{\varphi(45)}\equiv 1(45)$, $3^{\varphi(40)}\equiv 1\ (40)$.

26. Припустимо, що $x$ пробігає повну систему найменших невід'ємних лишків за

модулем 8. Знайти відповідні найменші невід'ємні лишки для виразу $7x+4$. (*Відп.* Якщо $x = 0, 1, 2, 3, 4, 5, 6, 7$, то $7x + 4 = 4,3,2,1,0,7,6,5 \pmod 8$).

27. Знайти найменший невід'ємний лишок $8^{80} + 13^{90}$ за модулем 17. (*Відп.* 0).

28. Довести, що коли $n$ непарне число, то $n^2 \equiv 1 \pmod 8$.

29. Довести, що коли НСД$(n, 8) = 1$, то $n^2 \equiv 1 \pmod 8$.

30. Довести формулу множення біномів: $(1 - p_1)(1 - p_2)\dots(1 - p_k) = 1 - p_1 - p_2 - \dots - p_k + p_1p_2 + \dots + p_{k-1}p_k - p_1p_2p_3 - \dots - p_{k-2}p_{k-1}p_k + \dots + (-1)^k p_1p_2\dots p_k$. (*Вказ.* Скористатися методом математичної індукції).

31. Виконати перевірку чисел 349, 641, 647, 1001, 2371 на простоту за допомогою послідовних ділень, критерію Вільсона та теореми Ферма.

32. У системі RSA з даними параметрами $p_A, q_A, d_A$ знайти параметри, яких бракує, і описати процес передачі повідомлення $m$ користувачу $A$:

а) $p_A = 5,\ q_A = 11,\ d_A = 3,\ m = 12$; б) $p_A = 5,\ q_A = 13,\ d_A = 5,\ m = 20$;
в) $p_A = 7,\ q_A = 11,\ d_A = 7,\ m = 17$; г) $p_A = 7,\ q_A = 13,\ d_A = 5,\ m = 30$;
д) $p_A = 3,\ q_A = 11,\ d_A = 3,\ m = 15$.

33. Користувачу системи RSA з параметрами $n = 187, d = 3$ передана криптограма $e = 100$. Розшифрувати цю криптограму, зламавши систему RSA користувача.

34. Написати програму реалізації системи RSA, передачі повідомлень між абонентами $A$ і $B$ з такими параметрами: $p_A131, q_A = 227, p_B = 113, q_B = 281, d_A = = d_B = 3$.

## 2.8. Еліптичні криві у криптографії

Використання еліптичних кривих у криптосистемах – це один із нових напрямків у криптографії. Еліптичні криві давно вивчаються в математиці, але їх криптографічне застосування було запропоновано Кобліцем і Міллнером лише в 1985 р. В 1998 р. криптографічне використання еліптичних кривих, таких як цифровий підпис, внесено до стандартів у США (ANSI X9.6.2 і FIPS 186-21), а з 2001 р. аналогічні стандарти були прийняті і в інших країнах (зокрема в Росії).

Основною перевагою криптосистем на еліптичних кривих є те, що порівняно з іншими криптосистемами, вони суттєво стійкіші до атак і їхня складність не вища за інші криптосистеми. Це пояснюється тим, що для обчислення обернених функцій на еліптичних кривих відомі лише експоненціальні алгоритми, у той час як для описаних вище систем запропоновані алгоритми лише субекспоненціальні. У результаті рівень стійкості, який досягається у інших системах (наприклад у системі RSA) з використанням 1024-бітового ключа,

у криптосистемі на еліптичних кривих використовується ключ 160 біт. А такий ключ має простішу програмну і апаратну реалізацію.

### 2.8.1. Математичні підстави

Крива третього порядку $E$, яка задається рівнянням

$$E : Y^2 = X^3 + aX + b, \tag{2.34}$$

називається **еліптичною кривою**.

Оскільки $Y = \pm\sqrt{X^3 + aX + b}$, то графік кривої симетричний відносно осі абсцис. Для знаходження точок перетину з віссю абсцис необхідно розв'язати кубічне рівняння

$$X^3 + aX + b = 0. \tag{2.35}$$

Корені цього рівняння можна знайти за допомогою відомих формул Кардано. Дискримінантом $D$ даного рівняння є вираз

$$D = (\frac{a}{3})^3 + (\frac{b}{2})^2. \tag{2.36}$$

Залежно від знака $D$ матимемо такі випадки:

– якщо $D < 0$, то (2.35) матиме три різні дійсні корені,

– якщо $D = 0$, то (2.35) матиме три дійсні корені, принаймні два з яких рівні між собою,

– якщо $D > 0$, то (2.35) матиме один дійсний корінь і два комплексні спряжені між собою корені.

Нижче (рис. 2.8.1 – 2.8.3) показано вигляди кривої у всіх трьох випадках.

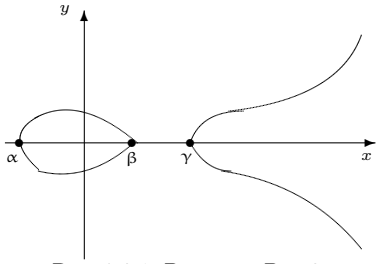



Рис. 2.8.1. Випадок $D < 0$

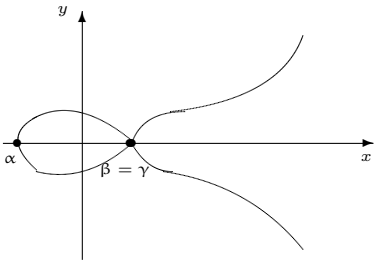



Рис. 2.8.2. Випадок $D = 0$

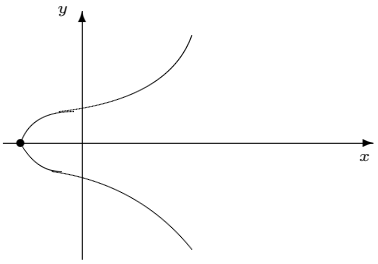



Рис. 2.8.3. Випадок $D > 0$

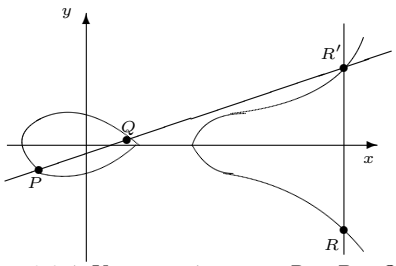



Рис. 2.8.4. Композиція точок $R = P + Q$

Еліптична крива з рис. 2.8.2 називається *сингулярною*, а точка $(\beta, 0)$ – *точкою сингулярності*. У цій точці крива має дві дотичні. Але сингулярні криві не цікаві з точки зору криптографії. Тому при визначенні кривої за допомогою параметрів $a$ і $b$ будемо вимагати виконання умови $D \neq 0$, що еквівалентно умові

$$4a^3 + 27b^2 \neq 0. \tag{2.37}$$

Нехай еліптична крива $E$ задана рівнянням (2.34) з обмеженнями на коефіцієнти (2.37). Визначимо композицію точок на кривій. Візьмемо які-небудь дві точки $P = (x_1, y_1), Q = (x_2, y_2) \in E$ і проведемо через них пряму (див. рис. 2.8.4). Ця пряма обов'язково перетне криву у третій точці, яку позначимо $R'$. Така точка обов'язково

існує, тому що в даному випадку кубічне рівняння має два дійсних корені, а тому і третій корінь цього рівняння теж має бути дійсним. Точку $R = (x_3, y_3)$ отримуємо шляхом зміни знака ординати точки $R'$. Точка $R$ називається *композицією точок* $P$ і $Q$ і позначається $R = P + Q$.

Нехай точка $P \in E$ має координати $(x, y)$, тоді точку $(x, -y)$ позначатимемо $-P$. Будемо вважати, що вертикальна пряма, яка проходить через точки $P$ і $-P$, перетинає еліптичну криву в нескінченно віддаленій точці $\mathcal{O}$, тобто $P + (-P) = \mathcal{O}$. Припускаємо, що $P + \mathcal{O} = \mathcal{O} + P = P$. Далі буде видно, що точка $\mathcal{O}$ дійсно відіграватиме роль нуля в операціях на еліптичній кривій.

Припустимо, що точки $P$ і $Q$ зближаються і зрештою зливаються в одну точку $P = Q = (x_1, y_1)$. Тоді композиція $R = (x_3, y_3) = P + Q = P + P = [2]P$ буде одержана шляхом проведення дотичної в точці $P$ і відбиття її другого перетину з кривою $R'$ відносно осі абсцис (рис. 2.8.5).

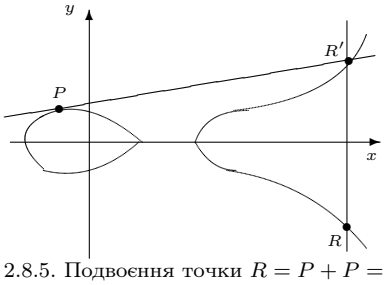



Рис. 2.8.5. Подвоєння точки $R = P + P = [2]P$

Знайдемо формули для обчислення координат точки $R(x_3, y_3)$ за координатами точок $P = (x_1, y_1)$ і $Q = (x_2, y_2)$. Розглянемо спочатку випадок, коли $P \neq Q$ і $R = P+Q$ (рис. 2.8.4). Нехай $k$ означає кутовий коефіцієнт прямої, яка проходить через точки $P$ і $Q$. Тоді

$$k = \frac{y_2 - y_1}{x_2 - x_1} \tag{2.38}$$

і рівняння прямої набуває вигляду $Y - y_1 = k(X - x_1)$. Звідси знаходимо

$$Y = y_1 + k(X - x_1). \tag{2.39}$$

Підставивши знайдений вираз для $Y$ у рівняння кривої, отримуємо

$$(y_1 + k(X - x_1))^2 = X^3 + aX + b.$$

Обчислюючи квадрат і зводячи подібні члени, дістаємо кубічне рівняння $X^3 - k^2X^2 + \cdots = 0$. За теоремою Вієта сума коренів кубічного рівняння дорівнює коефіцієнту при $X^2$, взятому з протилежним знаком, тобто $x_1 + x_2 + x_3 = k^2$, а звідси знаходимо

$$x_3 = k^2 - x_1 - x_2. \tag{2.40}$$

Підставивши $x_3$ в рівняння прямої (2.39), знаходимо ординату точки $R'$: $y'_3 = y_1 + k(x_3 - x_1)$ і, змінивши її знак на протилежний, дістаємо

$$y_3 = k(x_1 - x_3) - y_1. \tag{2.41}$$

Розглянемо тепер випадок, коли $P = Q$ і точка $R = [2]P$ (рис. 2.8.5). Диференціюючи обидві частини рівності (2.34) по $X$, дістаємо $2YY' = 3X^2 + a$. Кутовий коефіцієнт дотичної в точці $P$ дорівнює значенню похідної в цій точці, тобто

$$k = \frac{3x^2 + a}{2y_1}. \tag{2.42}$$

Далі йде повна аналогія з першим випадком і координати точки $R$ знаходять за формулами (2.40) і (2.41). Зауважимо, що коли ордината точки $P$ дорівнює нулю, то дотична проходить паралельно осі ординат і $[2]P = \mathcal{O}$.

Користуючись знайденими формулами для обчислення композиції точок і домовленості відносно точки $\mathcal{O}$ (нескінченно віддаленої), можна довести такі властивості точок і операції композиції на еліптичній кривій: для довільних точок $P, Q, S \in E$

1) $P + Q = Q + P$ (комутативність),

2) $(P + Q) + S = P + (Q + S)$ (асоціативність),

3) $P + \mathcal{O} = \mathcal{O} + P = P$ для нескінченно віддаленої точки $\mathcal{O}$,

4) існує точка $(-P) \in E$ така, що $P + (-P) = \mathcal{O}$.

Ці властивості збігаються з властивостями операції додавання для цілих чисел і власне через це композиція точок називається додаванням, а операція $[2]P$ – подвоєнням точки.

Завдяки цій аналогії вводяться такі позначення. Для довільного цілого $m \in Z$

$$[m]P = \underbrace{P + P + \cdots + P}_{m},$$
$$[-m]P = \underbrace{-(P + P + \cdots + P)}_{m},$$
$$[0]P = \mathcal{O}.$$

Тепер можна розглянути криптографічні застосування еліптичних кривих. Далі всі обчислення будуть виконуватися в полі $\mathcal{F}_p$ – полі лишків за модулем простого числа $p$. У цьому полі, як нам відомо, конгруентність $ax \equiv b \pmod p$ має єдиний розв'язок, а це гарантує існування елемента $a^{-1}$ такого, що $\forall a \neq 0\ a \cdot a^{-1} \equiv a^{-1} \cdot a \equiv \equiv 1 \pmod p$.

Отже, маємо еліптичну криву

$$E : Y^2 = X^3 + aX + b \pmod p, \tag{2.43}$$

яка називається *еліптичною кривою у формі Веєрштраса*. У цьому рівнянні змінні $X, Y$ і коефіцієнти $a, b$ набувають цілих значень, а решта обчислень виконується за модулем $p$. На підставі домовленості відносно коефіцієнтів $a$ і $b$, вони мають задовольняти умову

$$(4a^3 + 27b^2) \not\equiv 0 \pmod p. \tag{2.44}$$

Множина точок кривої $E$, яку позначатимемо $E_p(a, b)$, складається зі всіх пар чисел $(x, y), 0 \leq x, y < p$, які задовольняють рівняння (2.43), і нескінченно віддаленої точки $\mathcal{O}$. Кількість цих точок, тобто $|E_p(a, b)|$, має важливе значення для криптографічних систем.

**Приклад 2.8.1**. Розглянемо криву $E_7(2, 6) : Y^2 = X^3 + 2X + 6 \pmod 7$.

Перевіримо умову (2.44): $4 \cdot 2^3 + 27 \cdot 6^2 = 4 \cdot 1 + 6 \cdot 1 \equiv 3 (\neq 0) \pmod 7$. Отже, еліптична крива не сингулярна. Візьмемо деяку випадкову точку у множині $E_7(2, 6)$, наприклад, $X = 5$. Тоді $Y^2 = 5^3 + 2 \cdot 5 + 6 = 6 + 3 + 6 \equiv 1 \pmod 7$ і $y = 1 \pmod 7$ або $y = -1 = 6 \pmod 7$. Ми знайшли відразу дві точки (5,1) і (5,6). Знайдемо за допомогою композиції ще дві точки. Спочатку обчислимо [2](5,1). За формулами (2.42), (2.40) і (2.41) знаходимо

$$k = \tfrac{3 \cdot 5^2 + 2}{2 \cdot 1} = \tfrac{0}{2} \equiv 0 \pmod 7,$$
$$x_3 = 0 - 2 \cdot 5 \equiv 4 \pmod 7,$$
$$y_3 = 0 \cdot (5 - 4) - 1 \equiv 6 \pmod 7.$$

Отже, ми знайшли точку [2](5,1) = (4,6), яка належить кривій (у чому легко переконатися, підставивши у рівняння (2.43)). Обчислимо ще одну точку [3](5,1)=(5,1)+(4,6). За формулами (2.38), (2.40) і (2.41) знаходимо

$$k = \frac{6-1}{4-5} = \frac{5}{6} = 5 \cdot 6 \equiv 2 \pmod 7,$$
$$x_3 = 2^2 - 5 - 4 \equiv 2 \pmod 7,$$
$$y_3 = 2(5-2) - 1 = 2 \cdot 3 - 1 \equiv 5 \pmod 7.$$

Отже, знайшли точку [3](5,1) = (2,5), а всіх знайдених точок чотири. Для криптографічних застосувань кривої важливо знати, скільки всього точок у множині $E_7(2,6)$. Цю відповідь ми одержимо пізніше. ♠

Розглянемо множину $E_p(a,b)$. Очевидно, що ця множина скінченна, оскільки вона включає тільки точки $(x,y)$ із цілими координатами, $0 \le x, y < p$. Існує пряма аналогія між $E_p(a,b)$ і множиною степенів цілих чисел, які обчислюють за модулем $p$. Тоді на цій підставі у множині $E_p(a,b)$ існує породжуючий елемент, тобто такий елемент $g$, що ряд $g, [2]g, \ldots, [n]g$, де $n = |E_p(a,b)|$, включає всі точки із $E_p(a,b)$, причому $[n]g = \mathcal{O}$. Число точок на кривій, при відповідному виборі параметрів $p, a, b$, може бути простим числом, тобто $|E_p(a,b)| = q$. У цьому випадку довільна точка, крім $\mathcal{O}$, буде породжуючим елементом усієї множини точок. Така крива має певні переваги перед іншими і завжди може бути побудована в розумних часових межах. Якщо ж з яких-небудь причин таку криву не вдалося знайти і $|E_p(a,b)| = hq$, де $q$ – просте число, то в $E_p(a,b)$ існує підмножина із $q$ точок, породжуючим елементом якої може служити довільна точка $G \ne \mathcal{O}$, причому $[q]G = \mathcal{O}$. Далі, без утрати загальності, будемо вважати, що маємо саме таку підмножину точок потужності $q$, а при виборі кривої будемо намагатися отримати $|E_p(a,b)| = q$.

Основною операцією на еліптичній кривій є операція $m$-кратної композиції, тобто обчислення

$$Q = [m]p = P + P + \ldots + P(m \text{ доданків}). \tag{2.45}$$

Ця операція виконується досить ефективно і вимагає не більше $2\log m$ композицій точок. Метод її реалізації такий самий, як і операції піднесення до степеня. Наприклад, щоб отримати точку

$[21]P$, обчислюємо $[2]P, [4]P, [8]P, [16]P$, кожний раз подвоюючи попередню точку, і додаємо $P + [4]P + [16]P = Q$ (всього 4 подвоєння і 2 додавання).

Обернена задача, яка за традицією теж називається дискретним логарифмом на еліптичній кривій, формулюється таким чином. Знаючи точки $P$ і $Q$, знайти таке число $m$, що $[m]P = Q$. Ця задача виявляється надзвичайно складною. Якщо вдало вибрати параметри кривої, то найкращі відомі нині алгоритми пошуку такого числа $m$ потребують $O(\sqrt{q})$ операцій на кривій, де $q$ – потужність підмножини точок, якій належать точки $P$ і $Q$. Усі обчислення на кривій виконують за модулем $p$, тобто із числами довжини $t \approx \log p$ бітів. Для криптографічних застосувань $\log p \approx \log q$ і тому $O(\sqrt{q}) = O(2^{t/2})$ означає експоненціальне зростання складності при збільшенні довжини чисел.

### 2.8.2. Вибір параметрів еліптичної кривої

Вибір параметрів еліптичної кривої для розв'язання криптографічних задач зводиться до вибору параметрів $a, b$ і модуля $p$. Очевидно, що критерієм вибору є неможливість застосування методів криптоаналізу, які розроблені для певного типу таких кривих. Однією зі стратегій вибору параметрів є стратегія вибору **випадкової кривої**, яка вважається найбільш надійною і стійкою. Альтернативним підходом, який не буде розглядатися, є систематична побудова еліптичної кривої із заданими властивостями.

**Процес формування хорошої випадкової еліптичної кривої** подамо у вигляді кроків.

1. Вибираємо випадкове просте число $p$ (число точок на кривій – величина того самого порядку, що і $p$). У зв'язку із цим довжина в бітах числа $p$, $t = \lceil \log p \rceil + 1$, повинна бути такою, щоб зробити неможливим застосування загальних методів знаходження дискретних логарифмів на кривій, які мають складність $O(2^{t/2})$. Довжина $t = 128$ бітів (довжина чотирьох машинних слів на 32-розрядних комп'ютерах) на сьогодні недостатня, оскільки відомі випадки ламання відповідних

кривих за декілька місяців інтенсивних обчислень. А довжина $t = 160$ бітів (п'ять машинних слів) нині недосяжна для криптоаналітиків і може служити відправним пунктом. Другим аргументом є те, що шифр на еліптичній кривій повинен бути не менш стійким, ніж блоковий шифр AES (новітня модифікація стандартного шифру DES). Вважається, що стійкість AES забезпечується довжиною його ключа 128, 196 або 256 бітів. Оскільки стійкість шифру на еліптичній кривій визначається величиною $t/2$, то довжина модулів на цих кривих повинна складати відповідно 256, 392 і 512 бітів.

2. Вибираємо такі випадкові числа $a$ і $b$, що $a, b \not\equiv 0 \pmod{p}$ і $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$. Оскільки в обчисленнях композиції точок параметр $b$ не бере участі, то для підвищення ефективності обчислень рекомендується випадково вибирати тільки $b$, а $a$ вибирати рівним невеликому цілому числу. Стандарт США FIPS 186-2 передбачає використання еліптичних кривих із параметром $a = -3$, що спрощує обчислення.

3. Визначаємо кількість точок на кривій $n = |E_p(a, b)|$ (це найтрудозатратніший етап процесу). Важливо, щоб число $n$ мало великий простий дільник $q$, а краще аби воно само було простим числом, тобто $n = p$. Якщо $n$ розкладається на малі множники, то в $E_p(a, b)$ існує багато маленьких підмножин зі своїми генераторами, і алгоритм Поліга–Хеллмана (див. пункт 2.5.8) швидко обчислює логарифм на кривій через логарифми в цих маленьких підмножинах. Якщо пошук кривої з $n = q$ займає багато часу, то можна взяти $n = h \cdot q$, де $h$ – невелике число. Ще раз наголосимо, що стійкість криптосистеми на еліптичній кривій визначається не модулем $p$, а числом елементів $q$ у множині $E_p(a, b)$. Але коли множник $h$ – невелике число, то $q$ є величиною того ж порядку, що і $p$. Якщо ж $n$ не відповідає вимогам, то перейти до виконання кроку 2.

4. Перевіряємо виконання умови $(p^k - 1) \not\equiv 0 \pmod{q}$ для всіх $k$, де $0 < k < 32$. Якщо умова не виконується, то переходимо до виконання кроку 2. Ця перевірка виключає можливість

атаки MOV-методом (від прізвищ авторів Menezes, Okamoto, Vanstone [31]), а також виключає з розгляду так звані суперсингулярні криві і криві, у яких $|E_p(a,b)| = p - 1$.

Метод MOV і вказані особливі типи кривих дають можливість звести задачу обчислення логарифма на кривій до простіших задач.

5. Перевіряємо виконання нерівності $p \neq q$. Якщо вона не виконується, то перейти до виконання кроку 2. Зазначимо, що для кривих з умовою $p = q$, які називаються аномальними, існують ефективні методи обчислення логарифмів.

6. На цьому кроці необхідна крива для криптографічних застосувань знайдена. Знайдено параметри $a, b, p$, число точок $n$ і розмір підмножини точок $q$. Як правило, ще потрібно знайти точку $G$ – генератор цієї підмножини. Якщо $q = n$, то довільна точка (крім $\mathcal{O}$) є генератором. Якщо $q < n$, то вибираються випадкові точки $G'$ доки не отримаємо $G = \lfloor n/q \rfloor G' \neq \mathcal{O}$. Щоб отримати випадкову точку на кривій, беремо випадкове число $x < p$, обчислюємо $e = (x^3 + ax + b) \pmod p$ і намагаємося знайти корінь $y = \sqrt{e} \pmod p$. Якщо корінь існує, то отримуємо точку $(x, y)$, інакше пробуємо інше число $x$. Алгоритми обчислення квадратних коренів за модулем простого числа описано в [31].

### 2.8.3. Криптосистеми на еліптичних кривих

Довільна криптосистема, побудована на дискретному логарифмі, легко переноситься на еліптичні криві. Основний принцип побудови системи полягає в заміні операції $y = g^x \pmod p$ на $Y = [x]G \pmod p$. У другій формулі вказувати модуль не прийнято, хоча всі обчислення на кривій виконуються за модулем $p$. Перехід найбільш ефективний, коли показник степеня $x$ обчислюється за модулем $q$. На еліптичній кривій із потужністю робочої множини точок $q$ те саме відбувається з множником $x$. Різниця лише полягає в тому, що $y$ – це число, а $Y$ – це точка, і потрібно переходити від точки

до числа. Найбільш простий спосіб такого переходу – використати абсцису точки.

Розглянемо метод використання еліптичних кривих на прикладі шифра Ель-Гамаля.

### Шифр Ель-Гамаля на еліптичній кривій

Для користувачів певної мережі вибирається спільна еліптична крива $E_p(a,b)$ і точка $G$ на ній, така, що $G, [2]G, [3]G, \ldots, [q]G$ – всі різні точки, і $[q]G = \mathcal{O}$ для деякого простого числа $q$.

Кожний користувач $U$ вибирає число $c_U$, $0 < c_U < q$, яке зберігає як свій секретний ключ, і обчислює на кривій точку $D_U = [c_U]G$, яка буде його відкритим ключем. Параметри кривої і список відкритих ключів передаються всім користувачам мережі.

Припустимо, що користувач $A$ хоче передати повідомлення користувачу $B$. Будемо вважати, що повідомлення подано у вигляді числа $m < p$. $A$ виконує такі дії:

1) вибирає випадкове число $k$, $0 < k < q$;
2) обчислює $R = [k]G$, $P = [k]D_B = (x, y)$;
3) шифрує $e = mx \pmod p$;
4) висилає до $B$ шифрограму $(R, e)$.

Користувач $B$, отримавши шифрограму $(R, e)$,

1) обчислює $Q = [c_B]R = (x, y)$;
2) розшифровує $m' = ex^{-1} \pmod p$.

Для обґрунтування цього протоколу достатньо показати, що

$$[c_B]R = [c_B]([k]G) = [k]([c_B]G) = [k]D_B,$$

тобто $Q = P$ і тому $m' = m$.

Координата $x$ точки $Q$ залишається секретною для зловмисника, оскільки він не знає числа $k$. Зловмисник може спробувати обчислити $k$ з точки $R$, але для цього йому потрібно розв'язати проблему дискретного логарифма на кривій, а це вважається неможливим.

Найімовірнішим варіантом використання описаного протоколу буде передача числа $m$ як секретного ключа для блокового або потокового шифру. У такому випадку варто вибирати параметри

кривої так, щоб $\log q$ приблизно вдвічі перевищував довжину ключа шифру.

### 2.8.4. Визначення кількості точок на кривій

Розглянемо алгоритм Схоуфа для визначення кількості точок $|E_p(a,b)|$ на кривій, тобто точок, координати яких задовольняють рівнянню кривої (2.43) і є невід'ємними цілими числами, меншими $p$. Алгоритм Схоуфа був першим поліноміальним алгоритмом підрахунку кількості точок на еліптичній кривій і його складність $O(\log^6 p)$ операцій за модулем $p$. Цей алгоритм лежить в основі всіх сучасних методів, які застосовують до випадкових кривих.

В 1933 р. Хельмут Ґассе довів таку теорему.

**Теорема 45 (Ґассе).** *$|E_p(a,b)|$ задовольняє нерівність*

$$p+1-2\sqrt{p} \le |E_p(a,b)| \le p+1+2\sqrt{p}.$$

Зручним є зображення $|E_p(a,b)|$ у вигляді

$$|E_p(a,b)| = p+1-t. \tag{2.46}$$

Параметр $t$ в (2.46) називається *слідом Фробеніуса* для $E_p(a,b)$ і може набувати як додатних, так і від'ємних значень або може дорівнювати нулю. Згідно з теоремою Ґассе

$$|t| \le 2\sqrt{p}. \tag{2.47}$$

До цих пір нас цікавили тільки точки на кривій із невід'ємними цілими координатами, меншими $p$. Але тепер будемо розглядати всю множину розв'язків рівняння кривої (2.43) у множині комплексних чисел. Слід Фробеніуса пов'язаний із модулем $p$ таким чином.

**Теорема 46.** *Для всіх (комплексних чисел) $x$ і $y$, які задовольняють рівнянню кривої (2.43) з параметрами $0 \le a, b < p$, справедливе співвідношення*

$$(x^{p^2}, y^{p^2}) + |p|(x,y) = |t|(x^p, y^p), \tag{2.48}$$

*де додавання у формулі означає композицію точок на кривій.*

Щоб навчитися знаходити загальний розв'язок рівняння (2.48), зазначимо, що композиція точок вигляду $Q = [m]P$ може бути виражена через координати точки $P$. Наприклад,

$$[2](x,y) = (x' = (\tfrac{3x^2+a}{2y})^2 - 2x, y' = \tfrac{3x^2+a}{2y}(x - x') - y) = \\ = (\tfrac{x^4-2ax^2-8bx+a^2}{4y^2}, \tfrac{x^6+5ax^4+20bx^3-5a^2x^2-4abx-8b^2-a^3}{8y^3}).$$

Точку $[3](x,y)$ можна отримати як $[2](x,y) + (x,y)$ шляхом підстановки знайдених виразів для координат точки $[2](x,y)$ у формули (2.38), (2.40), (2.41). Процес отримання виразів для наступних точок виглядає досить складним, але він описується простою рекурсивною залежністю:

$\psi_0 = 0,$
$\psi_1 = 1,$
$\psi_2 = 2y,$
$\psi_3 = 3x^4 + 6ax^2 + 12bx - a^2,$
$\psi_4 = 4y(x^6 + 5ax^4 + 20bx^3 - 5a^2x^2 - 4abx - 8b^2 - a^3),$
$\psi_{2m+1} = \psi_{m+2}\psi_m^3 - \psi_{m-1}\psi_{m+1}^3,\ m \geq 2,$
$\psi_{2m} = (\psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2)\psi_m/2y,\ m > 2.$

Для $m \geq 2$ і $P = (x,y)$

$$[m]P = (\frac{\psi_m^2 x - \psi_{m-1}\psi_{m+1}}{\psi_m^2}, \frac{\psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2}{4y\psi_m^3}). \qquad (2.49)$$

Поліном $\psi_m(x,y)$ називається поліномом ділення $m$-го порядку. Оскільки точка $[m]P$ лежить на кривій, яка обчислюється за модулем $p$, то обчислення коефіцієнтів поліномів $\psi$ теж виконується за модулем $p$. Використовуючи наведену рекурсивну залежність, поліном ділення $m$-го порядку можна обчислити за $O(\log m)$ кроків (необхідні алгоритми і їхні оцінки можна знайти в [11]). Зауважимо, що коли $m$ непарне, то поліноми $\psi_m$ залежать тільки від однієї змінної $x$, оскільки друга змінна $y$ входить у них тільки в парних степенях, а $y^2$ замінюється на праву частину рівняння (2.43). Відмітимо також, що поліноми $(\psi_{2m}, \psi_{2m+1})$ включають у себе добутки чотирьох поліномів від $\psi_{m-2}$ до $\psi_{m+2}$ і тому степінь полінома $\psi_m$ зростає як $O(m^2)$.

Комплексна точка $P$ на кривій $E$ називається **торсійною** точкою $m$-го порядку, якщо $[m]P = \mathcal{O}$. Множину торсійних точок $m$-го порядку позначатимемо $E[m]$. На підставі (2.49) отримуємо, що точка $P = (x, y)$ є торсійною точкою $m$-го порядку тоді і тільки тоді, коли $\psi_m(x, y) = 0$.

Ідея алгоритму Схоуфа полягає у знаходженні розв'язків рівняння (2.48) відносно $t$ на множині торсійних точок малих порядків із наступним обчисленням загального розв'язку. Для торсійних точок $m$-го порядку множники обчислюють за модулем $m$, а поліноми – за модулем $\psi_m$, так що рівняння (2.48) набуває вигляду

$$(x^{p^2}, y^{p^2})_m + |p_m|(x, y) = |t_m|(x^p, y^p)_m, \tag{2.50}$$

де $p_m = p \pmod m, t_m = t \pmod m$, $(x^k, y^k)_m = (x^k \pmod{\psi_m}$, $y^k \pmod{\psi_m})$, крім того, $x$ і $y$ зв'язані рівнянням кривої $y^2 = x^3 + ax + b$. Візьмемо за модулі $m$ прості числа аж до деякого $m_{max}$, такого, що

$$\prod_{(m \text{ просте}) \wedge (2 \le m \le m_{max})} m > 4\sqrt{p}. \tag{2.51}$$

Тоді за знайденими числами $t_m$ однозначно відновлюється слід Фробеніуса $t$, який задовольняє (2.47), на підставі китайської теореми про остачі [12, 31].

Розглянемо коротко метод розв'язання рівняння (2.50) для випадку $m > 2$. Спочатку обчислюємо поліном $\psi_m$. Оскільки $m$ число непарне, то $\psi_m = \psi_m(x)$ (далі всі дії над поліномами виконуються за модулем $\psi_m$, степінь $y$ знижується до одиниці за допомогою рівняння кривої, коефіцієнти обчислюються за модулем $p$). Обчислимо поліноми $x^p, y^p, x^{p^2}, y^{p^2}$, користуючись тим самим алгоритмом піднесення до степеня, що і для цілих чисел. За формулою (2.49) обчислюємо $Q = [p_m](x, y)$ і, використовуючи формули додавання на кривій, знайдемо у символьному вигляді композицію точок $R = (x^{p^2}, y^{p^2})_m + Q$. Якщо $R = \mathcal{O}$, то $t_m = 0$. У протилежному випадку, щоб знайти $t_m$, обчислюємо абсцису точки $P = [\tau](x^p, y^p)_m$ для всіх $\tau$, $0 < \tau < m$. Для кожного значення $\tau$ потрібно перевірити виконання рівності $x_R$ і $x_P$. У загальному випадку різниця абсцис зображується у вигляді $x_R - x_P = u(x) - yv(x) = 0$. Беремо звідси

$y = \frac{u(x)}{v(x}$, підставляємо в рівняння кривої і дістаємо деякий поліном $h_x(x) = 0$. Якщо $h_x \neq 0 \pmod{\psi_m}$, то $x_R$ і $x_P$ не рівні між собою і потрібно випробовувати друге значення $\tau$. Якщо ж $x_R = x_P$, то обчислюємо ординату точки $P$ і, користуючись аналогічними прийомами, переводимо різницю $y_R - y_P$ в поліном $h_y(x) = 0$. Якщо $h_y \equiv 0 \pmod{\psi_m}$, то $t_m = \tau$, інакше $t_m \equiv -\tau \pmod{m}$.

Якщо $m = 2$, $\psi_2 = y$, то виникає складність з обчисленням $x^p \pmod{y}$. Але в цьому випадку допомагає один простий факт, який наводиться далі. Оскільки сингулярні криві не розглядаються, то приведена еліптична крива за модулем $p$ може мати один або три перетини з віссю абсцис. Усі інші точки йдуть парами $(x, y), (x, -y)$, і є одна точка на нескінченності $\mathcal{O}$. Таким чином, число точок на кривій парне або непарне залежно від того, розкладається чи ні поліном $X^3 + aX + b$ на множники за модулем $p$. Відомий простий критерій нерозкладності полінома за модулем $p$ [11, 31].

**Теорема 47.** *Поліном третього степеня F(X) не розкладається на множники за модулем p тоді і тільки тоді, коли НСД(F(X), $X^p - X$) = 1.*

У результаті маємо $t_2 = 1$, якщо НСД$(X^3 + aX + b, X^p - X) =$ $= 1$, і $t_2 = 0$ у протилежному випадку.

Розглянемо складність алгоритму Схоуфа. Спочатку нагадаємо одну відому властивість простих чисел (точне формулювання і доведення властивості можна знайти в [5]).

**Теорема 48.** *Кількість простих чисел, менших n, приблизно дорівнює* $\frac{n}{\ln n}$.

Із цієї теореми випливає, що $m_{max} = O(\log p)$ і кількість модулів, для яких ведуться обчислення, дорівнює $O(\log p)$. Найбільш складною операцією в алгоритмі є операція обчислення $x^{p^2}$ та інших подібних поліномів. При використанні швидких алгоритмів піднесення до степеня ці обчислення потребують у найгіршому випадку $O(\log p)$ операцій типу множення на поліноми степені $O(m^2) = O(\log^2 p)$. Кожна така операція вимагає $O(\log^4 p)$ операцій за модулем $p$, тобто всього налічується $O(\log^5 p)$ операцій за

модулем $p$. Одна операція за модулем $p$ виконується за допомогою $O(\log^2 p)$ бітових операцій. Отже, обчислення $x^{p^2}$ потребує $O(\log^7 p)$ бітових операцій. Беручи до уваги число різних модулів, для яких необхідно обчислювати $x^{p^2}$, дістаємо загальну складність $O(\log^8 p)$ бітових операцій для алгоритму Схоуфа.

**Приклад 2.8.2**. Знайти кількість точок на кривій, яка розглядалася у прикладі 2.8.1 (цей приклад запозичено з роботи [20]):

$$Y^2 = X^3 + 2X + 6 \pmod 7,\ (a = 2, b = 6).$$

Скористаємося спочатку алгоритмом, складність якого росте експоненціально. Будемо задавати значення $X$ від 0 до 6 і обчислювати відповідні їм значення $Y$. Спочатку запишемо таблицю квадратів за модулем 7:

$$0^2 = 0,\ 1^2 = 1,\ 2^2 = 4,\ 3^2 = 9 = 2,$$
$$4^2 = 2,\ 5^2 = 4,\ 6^2 = 1 \pmod 7.$$

Користуючись цією таблицею, знайдемо множину точок $E_7(2, 6)$:

| | | |
|---|---|---|
| $x = 0$ | $y^2 = 6$ | $y$ не існує |
| $x = 1$ | $y^2 = 1 + 2 + 6 = 2$ | $y = 3$ і $y = -3 = 4$ |
| $x = 2$ | $y^2 = 8 + 4 + 6 = 4$ | $y = 2$ і $y = -2 = 5$ |
| $x = 3$ | $y^2 = 27 + 6 + 6 = 4$ | $y = 2$ і $y = -2 = 5$ |
| $x = 4$ | $y^2 = 64 + 8 + 6 = 1$ | $y = 1$ і $y = -1 = 6$ |
| $x = 5$ | $y^2 = 125 + 10 + 6 = 1$ | $y = 1$ і $y = -1 = 6$ |
| $x = 6$ | $y^2 = 216 + 12 + 6 = 3$ | $y$ не існує |

.

Підраховуючи кількість знайдених точок і додаючи до них точку в нескінченності, дістаємо

$$|E_7(2, 6)| = 11.$$

Зрозуміло, що цей метод не застосовний при великому $p$, але будемо застосовувати отримане значення для перевірки.

Приступимо до виконання алгоритму Схоуфа. Нам достатньо буде трьох модулів $m = 2, 3, 5$, оскільки $2 \cdot 3 \cdot 5 = 30 > 4\sqrt{7} \approx 10,58$ (насправді вистачило б двох модулів $m = 3, 5$, але ми розглянемо ще і $m = 2$ для демонстрації алгоритму).

З метою спрощення позначень домовимося записувати поліноми у вигляді десяткових чисел (величина модуля дозволяє нам це зробити). Так, наприклад,

$$x^4 + 5x^2 + 2 = 1x^4 + 0x^3 + 5x^2 + 0x + 2 = 10502,$$
$$x^4 + x = 1x^4 + 0x^3 + 0x^2 + 1x + 0 = 10010.$$

Усі дії з коефіцієнтами поліномів виконуються за модулем 7. Спочатку обчислимо необхідні поліноми ділення для $m = 2$:

$\psi_0 = 0$,
$\psi_1 = 1$,
$\psi_2 = 2y$,
$\psi_3 = 30523$,
$\psi_4 = 4y \cdot 1031115 = 4054446y$,
$\psi_5 = \psi_4\psi_2^3 - \psi_1\psi_3^3 = 4054446 \cdot 1026^2 - 6025554626356 = 5055036550230$.

Розв'яжемо рівняння (2.50) для $m = 3$. Обчислюємо за модулем $\psi_3$:

$x^7 = 1363,$
$y^7 = (y^2)^3 y = 1026^3 y = 1360y,$
$x^{49} = 10 = x,$
$y^{49} = 1026^{24} = 6y,$
$p_3 = 7 \pmod 3 = 1.$

Ліва частина (2.50) перетворюється до

$$R = (x, 6y) + (x, y) = \mathcal{O}.$$

Отже, $t_3 = 0$.

Тепер розв'яжемо рівняння (2.50) для $m = 5$. Обчислюємо за модулем $\psi_5$:

$x^7 = 10000000,$
$y^7 = 1064524266y,$
$x^{49} = 531353334500,$
$y^{49} = 650465522521y,$
$p_5 = 2.$

Знаходимо точку $Q = [2](x, y)$. У загальному випадку її обчислюють за формулою (2.51):

$$Q = (\tfrac{4y^2x - \psi_3}{4y^2}, \tfrac{\psi_4}{4y^4}) = (\tfrac{10314}{4013}, \tfrac{1031115y}{1045431}).$$

Зважаючи, що $x_Q \neq x^{49}$, то потрібно шукати $\tau > 0$. Знайдемо точку $R = (x^{49}, y^{49}) + Q$ за формулами (2.38), (2.40), (2.41):

$$k = \tfrac{\frac{1031115y}{1045431} - 650465522521y}{\frac{10314}{4013} - 531353334500} = \tfrac{(1031115 - 650465522521 \cdot 1045431)4013y}{1045431(10314 - 531363334500 \cdot 4013)} = \tfrac{541024434205y}{115461562234},$$
$$x_R = \tfrac{541024434205^2 y^2}{115461562234^2} - 531363334500 - \tfrac{10314}{4013} = \tfrac{552631612401}{533030166456},$$
$$y_R = (531363334500 - \tfrac{552631612401}{533030166456}) \cdot \tfrac{541024434205y}{115461562234} - 650465522521y = \tfrac{515441613166y}{115165441243}.$$

Пробуємо $\tau = 1$, $P = (x^7, y^7)$ і перевіряємо виконуваність умови $x_R = x_P = 0$:

$$h_x = 552631612401 - 533030166456 \cdot 10000000 = 61115566241 \neq 0 \pmod{\psi_5}.$$

Пробуємо $\tau = 2$. Обчислюємо точку $P = [2](x^7, y^7)$. Скористаємося формулами додавання точок, оскільки додається $(x^7, y^7)$ до попередньої точки $P$. Дістаємо

$$k = \tfrac{3 \cdot 10000000^2 + 2}{2 \cdot 1064524266y} = \tfrac{434232361462}{2051341455y},$$
$$x_P = \tfrac{434232361462^2}{2051341455^2 y^2} - 2 \cdot 10000000 = \tfrac{213203662514}{220445441503}.$$

Перевіряємо виконання умови $x_R - x_P = 0$:

$$h_x = 552631612401 \cdot 220445441503 - 213203662514 \cdot 533030166456 = 0 \pmod{\psi_5}.$$

Отже, $x_R = x_P$ і тепер потрібно порівняти $y_R$ і $y_P$. Маємо

$$y_P = (10000000 - \tfrac{213203662514}{220445441503}) \cdot \tfrac{434232361462}{2051341455y} - 1064524266y = \tfrac{510334350655}{221015611231y}.$$

Перевіряємо виконання умови $y_R - y_P = 0$:

$$h_y = 515441613166y \cdot 221015611231y - 510334350655 \cdot 115165441243 = 0 \pmod{\psi_5}.$$

Отже, умови виконані і $t_5 = 2$.

Знайдемо тепер $t_2$. Нам потрібно знайти НСД для поліномів $x^3 + ax + b$ і $x^p - x$. Знайдемо це за допомогою алгоритму Евкліда для поліномів. При великих значеннях $p$, які використовуються у криптосистемах, ми не зможемо записати поліном $x^p - x$. Але на першому кроці алгоритму Евкліда обчислюється остача $(x^p - x) \pmod{(x^3 + ax + b)}$ і тому досить подати на вхід алгоритму не сам поліном $x^p - x$, а його остачу. Обчислюємо $x^p \pmod{(x^3 + ax + b)}$ за допомогою швидкого алгоритму піднесення до степеня і віднімаємо $x$. Після цього використовуємо алгоритм Евкліда. У нашому прикладі

$x^7 \equiv 304 \pmod{1026}$,
$x^7 - x = 364 \pmod{1026}$,
НСД(1026,364) = 1 (нагадаємо, що 1026 – це поліном $x^3 + 2x + 6$).
Отже, $t_2 = 1$. Тепер, користуючись китайською теоремою про остачі, маємо
$t = 1 \pmod 2$,
$t = 0 \pmod 3$,
$t = 2 \pmod 5$,
$N = 2 \cdot 3 \cdot 5 = 30$.
Розв'язок $t' = t \pmod N$ знаходимо за формулою

$$t' = \sum_{i=1}^{3} a_i N_i M_i \pmod N,$$

де

$$\begin{array}{lll} a_1 = 1, & N_1 = 30/2 = 15, & M_1 = 15^{-1} \equiv 1 \pmod 2, \\ a_2 = 0, & N_2 = 30/3 = 10, & M_1 = 10^{-1} \equiv 1 \pmod 3, \\ a_3 = 2, & N_3 = 30/5 = 6, & M_1 = 6^{-1} \equiv 1 \pmod 5. \end{array}$$

Підставляючи числа, дістаємо

$$t' = 1 \cdot 15 \cdot 1 + 0 \cdot 10 \cdot 1 + 2 \cdot 6 \cdot 1 = 27.$$

Щоб отримати розв'язок, який задовольняє нерівність (2.47), віднімаємо модуль:

$$t = t' - N = -3.$$

За формулою (2.46) знаходимо $|E_7(2,6)| = 7 + 1 - (-3) = 11$ ♠

Із цього прикладу випливає, що визначення кількості точок на кривій не зовсім проста задача і її розв'язання потребує потужної обчислювальної техніки. У практичних застосуваннях використовують покращені варіанти алгоритму Схоуфа, які ґрунтуються на

тонких конструкціях загальної алгебри, основною перевагою яких є зниження степеня поліномів ділення з $O(m^2)$ до $O(m)$. У результаті складність знижується до $O(\log^6 p)$ і може бути знижена і до $O(\log^{4+\varepsilon} p)$.

У зв'язку з такою ситуацією пропонується використовувати криві, які описані в різних стандартах або інших джерелах. Наприклад, в американському стандарті FIPS 186-2 наведено параметри еліптичних кривих для модулів різної довжини. А загалом немає жодних обмежень на використання всіма однієї добре підібраної еліптичної кривої.

## 2.9. Елементи загальної алгебри

Розглянемо коротко алгебраїчні структури, які використовують у критографії. Для цього нагадаємо означення алгебри, підалгебри та ізоморфізму.

**Означення 53. Універсальною Ω-алгеброю** (*або просто* **алгеброю**) *називається система* $G = (A, \Omega)$*, яка складається з деякої непустої множини* $A$ **(основна множина алгебри** *або* **носій алгебри)** *і множини визначених на* $A$ *операцій* $\Omega = \{\omega_1^{k_1}, \omega_2^{k_2}, \ldots, \omega_n^{k_n}, \ldots\}$ *фіксованої арності* **(сигнатура алгебри)**, *де* $k_i \in \in N$, $i = 1, \ldots, n, \ldots$. *Операції із множини* $\Omega$ *називаються* **основними операціями алгебри.**

*Якщо носій* $A$ *алгебри* $G = (A, \Omega)$ *має скінченну кількість елементів і це число дорівнює* $k$*, то така алгебра називається скінченною алгеброю* $k$*-го порядку.*

Скінченні алгебри при невеликому числі елементів, як правило, задають таблицями своїх операцій. Такі таблиці називають *таблицями Келі.*

Нехай $G = (A, \Omega)$ – довільна алгебра, $\omega \in \Omega$ – $n$-арна операція і $A' \subseteq A$. Підмножина $A'$ називається *замкнутою* відносно операції $\omega$, якщо для довільних $a_1, \ldots, a_n$ із $A'$ істинно $\omega(a_1, \ldots, a_n) \in A'$. Система $(A', \Omega)$ називається *підалгеброю алгебри* $(A, \Omega)$, якщо $A' \subseteq \subseteq A$ і $A'$ замкнута відносно довільної основної операції алгебри $G = = (A, \Omega)$.

Безпосередньо з означення підалгебри випливає, що непустий перетин довільної сукупності підалгебр універсальної алгебри $G = = (A, \Omega)$ буде підалгеброю цієї алгебри. Отже, якщо в алгебрі $G$ взята довільна підмножина $D \subset A$, то існує однозначно визначена підалгебра $\{D\}$, мінімальна серед підалгебр, які включають множину $D$. Це буде перетин усіх підалгебр із $G$, які цілком включають у себе $D$. Якщо $\{D\} = G$, а це означає, що довільний елемент із $G$ хоча б одним способом може бути записаний у вигляді скінченного виразу через елементи із $D$ і операції із $\Omega$, то $D$ називається **системою твірних або породжуючих** для $G$. Алгебра $G = \{D\}$

називається **скінченно породжуваною**, якщо множина $D$ скінченна.

**Означення 54.** *Універсальні алгебри $G = (A, \Omega)$ і $Q = (B, \Omega')$ називаються алгебрами одного* **типу**, *якщо між елементами сигнатур $\Omega$ і $\Omega'$ існує бієкція, за якої довільна операція $\omega$ із $\Omega$ і відповідна їй операція $\omega'$ із $\Omega'$ мають одну і ту ж арність.*

Оскільки операції алгебр одного типу ведуть себе однаково, то можна вважати, що ці алгебри мають однакові сигнатури операцій.

**Означення 55.** *Нехай алгебри $G = (A, \Omega)$ і $Q = (B, \Omega)$ одного типу. Якщо існує відображення $\varphi : A \to B$ таке, що для всіх елементів $a_1, \ldots, a_n$ із $A$ і довільної $n$-арної операції $\omega$ із $\Omega$ справедлива рівність $\varphi(\omega(a_1, \ldots, a_n)) = \omega(\varphi(a_1), \ldots, \varphi(a_n))$, то відображення $\varphi$ називається* **гомоморфізмом**, *а алгебра $G$ такою, що гомоморфно відображається в алгебру $Q$.*

*Якщо гомоморфізм алгебри $G$ в алгебру $Q$ є ін'єкцією, то він називається* **мономорфізмом**, *а якщо гомоморфізм алгебри $G$ в алгебру $Q$ є сюр'єкцією, то він називається* **епіморфізмом**.

*Якщо гомомрфізм $\varphi$ – бієкція алгебри $G$ на алгебру $Q$, то він називається* **ізоморфізмом**, *а алгебри $G$ і $Q$ –* **ізоморфними** *($G \sim Q$).*

*Гомоморфізм алгебри $G$ в алгебру $G$ (тобто $\varphi : G \to G$) називається* **ендоморфізмом**, *а коли $\varphi$ ізоморфізм $G$ на $G$, то він називається* **автоморфізмом**.

*Відношення еквівалентності $R$, задане на носієві алгебри $G$, називається* **конгруентністю**, *якщо для довільної $n$-арної операції $w \in \Omega$ і довільних елементів $a_i, a'_i \in G$, $i = 1, \ldots, n$, із того, що $a_i R a'_i$ випливає $w(a_1, \ldots, a_n) R w(a'_1, \ldots, a'_n)$.*

### 2.9.1. Групи

Групи – одна з найуживаніших універсальних алгебр не тільки у криптографії, а і в інших сферах людської діяльності.

**Означення 56.** *Універсальна алгебра $G(A,\Omega)$ називається групою, якщо множина $\Omega$ включає бінарну операцію множення $(\cdot)$, унарну операцію взяття оберненого елемента $(^{-1})$ і нульарну операцію $(e)$, яка фіксує одиницю групи, а операція множення задовольняє законам асоціативності $((a\cdot b)\cdot c = a\cdot(b\cdot c))$, скорочення $(a\cdot a^{-1} = a^{-1}\cdot a = e)$ та законам для одиниці $(a\cdot e = e\cdot a = a)$, де $a,b,c\in A$.*

Елементи $a$ і $a^{-1}$ називають *взаємно оберненими*. Послідовність елементів групи утворює вирази, які часто називають словами. Таке слово називається *нескорочуваним*, якщо воно не має жодної пари взаємно обернених символів, які стоять поруч. Тотожні співвідношення групи дозволяють записати довільне слово групи у вигляді нескорочуваного слова

$$a_{i_1}^{n_1} a_{i_2}^{n_2}\cdots a_{i_k}^{n_k}, \tag{2.52}$$

де $n_j$ – ціле число, $a_{ij}\in A$ не обов'язково всі різні, $j = 1,2,\ldots,k$, а $a^n$ – слово довжини $n$, яке визначається таким чином:

$$a^n = \begin{cases} \underbrace{aa\cdots a}_{n \text{ разів}}, & \text{якщо} \quad n>0; \\ e, & \text{якщо} \quad n=0; \\ \underbrace{a^{-1}a^{-1}\cdots a^{-1}}_{n \text{ разів}}, & \text{якщо} \quad n<0. \end{cases}$$

Безпосередньо з означення групи випливають такі наслідки:

а) одиничний елемент у групі єдиний;

б) обернений елемент до даного елемента групи єдиний;

в) рівняння $ax = b$ у групі має єдиний розв'язок.

У випадку а) припустимо, що у групі існує принаймні два одиничні елементи $e$ і $e'$. Тоді, на підставі їхньої властивості дістаємо $e = e\cdot e' = e'$. Отже, $e = e'$.

У випадку б) з означення групи випливає існування таких елементів $a'$ і $a''$, що $a\cdot a' = e$ і $a''\cdot a = e$. Але тоді на підставі закону асоціативності дістаємо $a''aa' = a''(aa') = a''\cdot e = a''$ і $a''aa' = (a''a)a' = e\cdot a' = a'$. Звідки отримуємо, що $a' = a''$.

У випадку в) нехай $c$ – розв'язок рівняння $ax = b$, тоді дістаємо тотожність $ac = b$. Домножаючи обидві частини цієї тотожності зліва на $a^{-1}$ дістаємо $c = a^{-1}(ac) = a^{-1}b$. Якщо $d$ – деякий інший розв'язок рівняння $ax = b$, то виконуючи ті самі дії, що і вище, дістаємо $d = a^{-1}b$. А це означає, що $c = d$.

**Приклад 2.9.1.** Розглянемо множину $A =$ {а,б,в,г,д,е} з операцією множення, яка задана такою таблицею, і покажемо, що вона є групою, яку позначатимемо $G_s$.

| * | а | б | в | г | д | е |
|---|---|---|---|---|---|---|
| а | е | д | г | в | б | а |
| б | в | г | д | е | а | б |
| в | б | а | е | д | г | в |
| г | д | е | а | б | в | г |
| д | г | в | б | а | е | д |
| е | а | б | в | г | д | е |

.

З таблиці множення випливає, що роль одиниці відіграє елемент "е". Крім того, оскільки таблиця множення не симетрична відносно головної діагоналі, то операція множення не є комутативною.

Оберненими елементами для елементів "а,б,в,г,д" є елементи "а,г,в,б,д" відповідно. Отже, елементи "а,в,д" обернені самі до себе, а елементи "б" і "г" – взаємно обернені.

Залишається показати, що операція множення задовольняє закон асоціативності, тобто $\forall x, y, z \in A\ x(yz) = (xy)z$. Це зводиться до розгляду окремих випадків, яких за основним правилом комбінаторики буде $6^3 = 216$. Але для одиничного елемента немає потреби розглядати випадки, оскільки вони очевидним чином випливають безпосередньо з таблиці множення. Отже, кількість випадків зменшується до $5^3 = 125$.

Розглянемо випадки виразів, які включають елементи, що обернені самі до себе: а(а$x$) = (аа)$x$ = е$x$ = $x$. Оскільки а$x$ = е, д, г, в, б, а, якщо $x$ = а, б, в, г, д, е відповідно, то рівність лівої і правої частин встановлено. Аналогічна перевірка і для елементів "в, д".

Решту випадків і доведення того, що оберненим елементом до елемента $xy$ буде елемент $y^{-1}x^{-1}$, пропонуємо читачеві. ♠

Нехай $H$ – деяка підгрупа групи $G$ і $a$ – довільний елемент із $G$. Множина $aH = \{ah | h \in H\}$ називається *лівим суміжним класом групи $G$ за підгрупою $H$*, заданим елементом $a$. Зрозуміло, що $a \in$ $\in aH$, оскільки $e \in H$, і якщо $b \in aH$, то $bH = aH$. Дійсно, $b \in aH$ означає що $b = ah$, де $h \in H$. Але тоді для довільних $h_1, h_2 \in H$ маємо $bh_1 = (ah)h_1 = a(hh_1) \in aH$, тобто $bH \subseteq aH$. Нехай тепер $b = ah \in aH$, тоді для довільного $h_2 \in H$ дістаємо

$$ah_2 = (bh^{-1})h_2 = b(h^{-1}h_2) \in bH,$$

тобто $aH \subseteq bH$. Отже, $aH = bH$.

Звідси випливає, що ліві суміжні класи $H, a_1H, a_2H, \ldots, a_nH, \ldots$ є класами розбиття групи $G$. Отже, підгрупа $H$ задає на $G$ відношення еквівалентності $R:\ aRb \Leftrightarrow aH = bH$.

Задання групи $G$ у вигляді об'єднання класів $H \cup a_1H \cup a_2H \cup \cup a_3H \cup \cdots \cup a_nH \cup \cdots$ називається *лівостороннім розкладом групи* $G$ за підгрупою $H$. Аналогічно будується і правосторонній розклад групи $G$ за підгрупою $H$. Правосторонній і лівосторонній розклади групи складаються з одного і того ж числа класів. У цьому легко переконатися, задаючи відображення $f : G \to G$ так, що $f(a) = a^{-1}$.

Якщо ж кількість суміжних класів у розкладі групи $G$ за підгрупою $H$ скінченна, то підгрупа $H$ називається підгрупою **скінченного індексу**, а кількість класів – **індексом** підгрупи $H$ у групі $G$.

Підгрупа $H$ називається **нормальним дільником**, або **інваріантною підгрупою** групи $G$, якщо лівосторонній розклад $G$ за $H$ збігається з правосторонім розкладом $G$ за $H$. Інакше кажучи, для довільного $a \in G$ виконується рівність $aH = Ha$ або $aHa^{-1} = H$. Очевидно, що нормальними дільниками довільної групи буде одинична підгрупа, яка складається лише з одиниці групи, і сама група. Ці нормальні дільники називаються *тривіальними*.

Розклад групи $G$ за підгрупою $H \subseteq G$, як було показано вище, задає відношення еквівалентності на $G$. Оскільки розклад може бути лівостороннім і правостороннім, то цих еквівалентностей буде дві – $R_1$ і $R_2$. Означення цих еквівалентностей мають вигляд: $\forall a, b \in G$

$$(a, b) \in R_1 \Leftrightarrow a^{-1}b \in H \text{ і } (a, b) \in R_2 \Leftrightarrow ab^{-1} \in H.$$

Еквівалентності $R_1$ і $R_2$, як неважко переконатися, є конгруентностями [10] і якщо підгрупа $H$ – нормальний дільник групи $G$, то обидві конгруентності збігаються, тобто $R_1 = R_2$. Обернене твердження теж виконується: якщо $R$ – відношення конгруентності на $G$, то клас розбиття, якому належить одиниця групи, є її нормальним дільником. У загальному випадку справедлива така теорема.

**Теорема 49.** *У довільній групі $G$ її конгруентності знаходяться у взаємно однозначній відповідності з її нормальними діль-*

*никами.*

**Приклад 2.9.2.** Повернемося до групи $G_s$ із прикладу 2.9.1. Безпосередньо з таблиці множення цієї групи можна помітити, що вона має підгрупу $H$, носій якої складається з елементів "е,б,г". Дійсно, ця множина елементів замкнута відносно операцій групи:

| * | е | б | г |
|---|---|---|---|
| е | е | б | г |
| б | б | г | е |
| г | г | е | б |

.

Із симетричності таблиці множення випливає закон комутативності для цієї підгрупи. Лівосторонній розклад групи $G_s$ за підгрупою $H$ має вигляд:

$$G_s = H \cup \text{а}H = \{\text{е,б,г,}\} \cup \{\text{а, д, в}\} = \{\text{е,а,б,в,г,д}\},$$

оскільки елементи д, в є елементами а$H$ (у чому легко переконатися: а$H$ = {а, д, в} = = д$H$ = {в, а, д} = в$H$ = {д, в, а}).

Неважко переконатися, що правосторонній розклад збігатиметься з лівостороннім розкладом. Звідси випливає, що підгрупа $H$ – нормальний дільник групи $G_s$, оскільки а$H$а = в$H$в = д$H$д = $H$. ♠

Нехай $G$ і $G'$ – групи, $e$ і $e'$ – їхні одиничні елементи відповідно, а $h$ – гомоморфізм групи $G$ у групу $G'$. Тоді множина $K = \{a \in G | h(a) = e'\}$ називається **ядром гомоморфізму** $h$ і позначається символом $ker(h)$. Неважко довести, що $ker(h)$ – нормальний дільник групи $G$.

Якщо група скінченна, то кількість її елементів називається *порядком групи* і, очевидно, всі її підгрупи теж будуть скінченними. Нехай $G$ – скінченна група $n$-го порядку і $H$ – її підгрупа $k$-го порядку. Тоді в розкладі $G$ за $H$ довільний суміжний клас складається точно із $k$ елементів і, отже, $n = kj$. Звідси випливає така теорема.

**Теорема 50 (Лагранжа)** . *Порядок і індекс довільної підгрупи скінченної групи є дільниками порядку групи.*

Якщо група $G$ породжується одним своїм елементом $a \in G$, то така група називається **циклічною**. Степені елемента $a$ не обов'язково повинні бути різними. Елемент $a$ групи $G$ називається **елементом скінченного порядку**, якщо існують такі цілі числа $k$ і $l$, що $a^k = a^l$, тобто $a^{k-l} = e$. Найменший показник серед усіх

таких показників елемента $a$ називається *порядком елемента* $a$. Якщо таких чисел $k$ і $l$ не існує, то $a$ має нескінченний порядок.

**Приклади циклічних груп**. а) Для $n \geq 1$ адитивна група лишків $\mathcal{Z}_n$ за модулем $n$ є циклічною групою. Породжуючим елементом цієї групи, очевидно, є одиниця.

б) Мультиплікативна група $\mathcal{Z}_5$ породжується елементами 2 і 3, оскільки

$$2^1 = 2,\ 2^2 = 4,\ 2^3 = 3,\ 2^4 = 1,$$
$$3^1 = 3,\ 3^2 = 4,\ 3^3 = 2,\ 3^4 = 1.$$

Елемент 4 не може бути породжуючим, оскільки $4^x \equiv 2 \pmod 5$ не має розв'язку у цій групі. ♠

Якщо $a$ – елемент скінченного порядку, то $a^0 = e, a, a^2, \ldots \ldots, a^{n-1}$ будуть різними елементами групи. Якщо дано $a^k$, де $k > n$, то $k = qn + r,\ 0 \leq r < n$, і $a^{qn+r} = (a^n)^q \cdot a^r = a^r$. Отже, скінченний порядок елемента збігається з порядком його циклічної групи.

**Наслідок 6**. *Порядок довільного елемента скінченної групи є дільником порядку групи.*

**Наслідок 7**. *Довільна скінченна група, порядок якої є простим числом, буде циклічною.*

Дійсно, на підставі простоти порядку групи вона повинна збігатися із циклічною підгрупою, породженою довільним її елементом, відмінним від одиниці. ■

Група називається *простою*, якщо вона не має нетривіальних нормальних дільників.

Якщо група $H$ – нормальний дільник групи $G$, то множина суміжних класів складає групу. Дійсно, якщо $H, a_1H, a_2H, \ldots$ – розбиття групи $G$ і для довільного елемента $a$ із $G$ виконуються рівності $aH = Ha$ і $H \cdot H = H$, то неважко довести, що:

1) $(aH \cdot bH) \cdot cH = aH \cdot (bH \cdot cH)$;

2) $aH \cdot H = H \cdot aH = aH = Ha$, тобто $H$ відіграє роль одиниці;

3) $aH \cdot (a^{-1})H = (aHa^{-1}) \cdot H = H \cdot H = H$;

4) $aH \cdot bH = (Ha) \cdot (bH) = (H(ab)) \cdot H = abH \cdot H = Hab$.

Іншими словами, довільний нормальний дільник $H$ групи $G$ задає деяке відношення конгруентності $R$ на $G$. Фактор-група за

цим відношенням конгруентності позначається $G/H$ і називається **фактор-групою групи** $G$ за нормальним дільником $H$.

Якщо група $G$ – скінченна і $H$ – її нормальний дільник, то одержуємо такий наслідок.

**Наслідок 8**. *Порядок групи $G/H$ дорівнює індексу підгрупи $H$ у групі $G$ і є дільником порядку групи $G$.*

### 2.9.2. Групи підстановок

Розглянемо приклад некомутативної групи, яка відіграє важливу роль у теорії груп, – групу підстановок деякої скінченної множини $M = \{1, 2, , \ldots, n\}$. **Підстановкою** називається бієктивне відображення множини $M$ на себе.

Сукупність усіх підстановок множини $M$ складає групу. Дійсно, роль одиниці в цій групі відіграє тотожна підстановка, яка залишає на місці кожний елемент із $M$. Далі, якщо елемент $a$ переходить в елемент $f(a)$, то $f^{-1}(f(a)) = a$ і $f^{-1}$ теж буде підстановкою на підставі взаємної однозначності $f$. $f^{-1}$ буде відігравати роль оберненого елемента для $f$. З попереднього відомо, що добуток відображень є операцією асоціативною. Отже, всі підстановки множини $M$ складають групу, яка називається **симетричною групою на множині** $M$.

**Приклад 2.9.3.** Якщо множина $M$ скінченна і складається з $n$ елементів, то симетрична група на $M$, яка називається **симетричною групою $n$-го степеня**, буде скінченною і матиме порядок $n!$. Підстановки скінченних множин задають у вигляді таблиць відповідностей. Наприклад, якщо $M = \{1, 2, 3, 4, 5, 6\}$, то підстановки $f$ і $f_1$ множини $M$, відповідні значення яких $\{4, 3, 1, 5, 2, 6\}$ і $\{6, 3, 5, 2, 4, 1\}$, матимуть вигляд

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 3 & 1 & 5 & 2 & 6 \end{pmatrix} \quad f_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 6 & 3 & 5 & 2 & 4 & 1 \end{pmatrix}.$$

Множення підстановок $f \cdot f_1$ є суперпозицією цих відображень і тоді

$$f \cdot f_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 3 & 1 & 5 & 2 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 6 & 3 & 5 & 2 & 4 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 5 & 6 & 4 & 3 & 1 \end{pmatrix}.$$

Нехай маємо підстановки $f_1, f_2, f_3$, які задані наведеними нижче таблицями. Для прикладу покажемо, що $(f_1 \cdot f_2) \cdot f_3 = f_1 \cdot (f_2 \cdot f_3)$:

$$f_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 3 & 5 & 2 & 6 & 1 \end{pmatrix}, \qquad f_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 5 & 2 & 6 & 1 & 4 \end{pmatrix},$$

$$f_3 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 6 & 3 & 5 & 1 & 4 & 2 \end{pmatrix}.$$

Тоді

$$(f_1 \cdot f_2) \cdot f_3 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 6 & 4 & 1 & 5 \end{pmatrix}, \quad f_1 \cdot (f_2 \cdot f_3) = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 6 & 4 & 1 & 5 \end{pmatrix}.$$

Розглянемо приклад задання групи $G'$ всіх підстановок множини $M = \{1, 2, 3\}$ за допомогою таблиці Келі. У цьому випадку група $G'$ – скінченна алгебра 6-го порядку і її елементами є підстановки

$$f_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \quad f_2 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}, \quad f_3 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix},$$

$$f_4 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix}, \quad f_5 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}, \quad f_6 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}.$$

Таблиця множення цієї алгебри набуває вигляду

| $*$ | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 4 | 5 | 6 |
| 2 | 2 | 3 | 1 | 6 | 4 | 5 |
| 3 | 3 | 1 | 2 | 5 | 6 | 4 |
| 4 | 4 | 5 | 6 | 1 | 2 | 3 |
| 5 | 5 | 6 | 4 | 3 | 1 | 2 |
| 6 | 6 | 4 | 5 | 2 | 3 | 1 |

,

де $h(f_i) = i,\ 1 = 1, 2, 3, 4, 5, 6$.

Таблиця операції взяття оберненого елемента вже міститься в таблиці множення, тому наводити її немає сенсу.

Пропонується переконатися самостійно, що отримана група ізоморфна групі $G$ з прикладу 2.9.1, якщо відображення $\varphi : G \to G'$ є таким:

$$\varphi(\text{а}) = 4,\ \varphi(\text{б}) = 2,\ \varphi(\text{в}) = 6,\ \varphi(\text{г}) = 3,\ \varphi(\text{д}) = 5,\ \varphi(\text{е}) = 1.\ \spadesuit$$

Окремим випадком підстановки є **інволюція**.

**Означення 57.** *Підстановка $p$ скінченної множини $S$ називається інволюцією, якщо $p = p^{-1}$ або $p(p(x)) = x$ для довільного $x \in S$.*

**Приклад інволюції**. Підстановка $p : S \to S$, де $S = \{1, 2, 3, 4, 5\}$,

$$p = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 2 & 5 & 1 & 3 \end{pmatrix},$$

є інволюцією. Другим прикладом інволюції є функція додавання за модулем 2. $\spadesuit$

Важливість груп підстановок випливає з такої теореми.

**Теорема 51 (Келі).** *Для довільної групи $G = (A, \Omega)$ існує ізоморфна їй група підстановок $G' = (M, \Omega)$ на відповідній множині $M$ [10].*

У зв'язку з теоремою Келі розглянемо будову групи підстановок детальніше. Нехай $M$ – деяка $n$-елементна множина і $f$ – її підстановка:

$$f = \begin{pmatrix} 1 & 2 & \dots & n \\ i_1 & i_2 & \dots & i_n \end{pmatrix},$$

де $i_j \in M, j = 1, 2, \dots, n$. Якщо $f$ не тотожна підстановка, то для деякого $i \in M$ буде $f(i) = j$ і $i \neq j$. Знайдемо значення $f(j) = k$, а потім значення $f(k) = l$ і т. д. Зрозуміло, що не більше, ніж через $n$ кроків у цій послідовності число $i$ зустрінеться знову. Нехай $i \to j \to k \to \dots \to m \to i$ означає пройдений шлях обчислень і якщо кількість різних елементів у цій послідовності дорівнює $s$, то говорять, що підстановка $f$ має *цикл довжини* $s$. Цикл довжини 2 називається **транспозицією**. Довільний цикл записується у вигляді

$$z_1 = (ijkl \cdots m),$$

де останнім є число $m$, яке переходить при відображенні $f$ у число $i$, з якого починається цикл. Очевидно також, що цикл $z$ може починатися з довільного числа $j$ цього циклу. Наприклад, запис циклу $z$ міг би бути таким:

$$z_1 = (kl \cdots mij).$$

Якщо $f(i) = i$, то маємо цикл $i \to i$, або у введеній нотації цикл $(i)$ довжини 1. Цикли довжини 1 є нерухомими точками підстановок.

Якщо довжина циклу $z_1$ дорівнює $|M| = n$, то говорять, що підстановка $f$ складається з єдиного циклу довжини $n$ і такий цикл називається **повним циклом**. У протилежному випадку візьмемо довільне число з $M$, яке не ввійшло в цикл $z_1$ довжини $s_1 < n$, і починаючи з нього, побудуємо таким же способом другий цикл $z_2$ довжини $s_2$. Зазначимо, що цикли $z_1$ і $z_2$ не мають спільних елементів на підставі того, що відображення $f$ взаємно однозначне.

Якщо цикли $z_1$ і $z_2$ не вичерпують усієї множини $M$, то будуємо третій цикл $z_3$ довжини $s_3$ і т. д. У результаті отримуємо цикли

$$z_1 z_2 \cdots z_r, \tag{2.53}$$

які вичерпують усі елементи множини $M$, і довільні два цикли не мають спільних елементів. Запис підстановки $f$ у вигляді (2.53) називається циклічним розкладом підстановки $f$, тобто $f = z_1 z_2 \cdots z_r$, причому порядок слідування циклів у такому поданні неістотний. Циклічний розклад підстановки часто називають *добутком циклів* підстановки.

Звідси випливає, що кожна така підстановка індукує відношення еквівалентності $R$ на множині $M$:

$$xRy \Leftrightarrow (\exists k \in \mathcal{Z}) y = f^k(x).$$

Дійсно воно рефлексивне, оскільки $x = f^0(x) = \varepsilon(x)$, де $\varepsilon$ – тотожна підстановка.

Симетричність випливає з того, що коли $y = f^k(x)$, тобто $xRy$, то $x = f^{-k}(y)$. А це означає, що $yRx$.

Транзитивність $R$ випливає з того, що коли $xRy$ і $yRz$, тобто $y = f^k(x)$ і $z = f^m(y)$, то $z = f^{k+m}(x)$. А це означає, що $xRz$.

Відношення $R$ розбиває множину $M$ на класи еквівалентності:

$$M = M_1 \cup M_2 \cup \cdots \cup M_k, \tag{2.54}$$

де $M_i \cap M_j = \emptyset$, коли $i \neq j$, $i, j = 1, 2, \ldots, k$.

**Означення 58.** *Класи еквівалентності $M_i$ за відношенням $R$ називаються* **орбітами** *підстановки $f$.*

Очевидно, $f(M_1) = M_1, \ldots, f(M_k) = M_k$ і звуження $f_1, \ldots, f_k$ підстановки $f$ на орбіти $M_1, \ldots, M_k$ теж будуть підстановками.

**Означення 59.** *Звуження $f_i$ підстановки $f$ на множину $M_i$ називається* **циклом** *цієї підстановки.*

Цикл є підстановкою степеня $m_i = |M_i|$ на множині $M_i$, тобто $f^{m_i}(x) = x,\ x \in M_i$, $i = 1, 2 \ldots, k$, і його можна записати у вигляді

$$(x, f_i(x), f_i^2(x), \ldots, f_i^{m_i-1}(x)).$$

Це дає змогу розкласти довільну підстановку $f$ на цикли

$$(x_1, f_1(x_1), \ldots, f_1^{m_1-1}(x_1)) \ldots (x_k, f_k(x_k), \ldots, f_k^{m_k-1}(x_k)).$$

Розширимо підстановки $f_1, f_2, \ldots, f_k$ до підстановок степеня $n$, покладаючи $f_i(x) = x$, якщо $x \in M \setminus M_i$. Тоді розклад підстановки $f$ можна подати у вигляді добутку циклів

$$f = f_1 f_2 \cdots f_k.$$

Цей запис однозначний із точністю до циклічних перестановок елементів у циклах і не залежить від порядку співмножників.

**Означення 60.** *Підстановка $f$ на множині $M$ має порядок $p$, якщо $f^p = \varepsilon$ і це число $p$ найменше серед усіх таких чисел.*

**Теорема 52.** *Порядок $p$ підстановки $f$ дорівнює найменшому спільному кратному довжин циклів у розкладі $f$.*

*Доведення.* Якщо розклад підстановки $f$ має вигляд $f = f_1 f_2 \cdots \cdots f_k$, то

$$f^p = f_1^p f_2^p \cdots f_k^p.$$

Оскільки $f_i^{m_i}(x) = x$ $(i = 1, \ldots, k)$, то $p = \text{НСК}(m_1, \ldots, m_k)$. ■

**Означення 61.** *Підстановка $f$ на множині $M$ належить цикловому класу $\{1^{\alpha_1} 2^{\alpha_2} \cdots n^{\alpha_n}\}$, якщо вона має $\alpha_1$ циклів довжини 1, $\alpha_2$ циклів довжини 2,..., $\alpha_n$ циклів довжини $n$ $(1 \cdot \alpha_1 + 2 \cdot \alpha_2 + ; ; + \cdots + n \cdot \alpha_n = n)$.*

Нехай $P(\alpha_1, \alpha_2, \ldots, \alpha_n)$ – кількість підстановок у цикловому класі $\{1^{\alpha_1} 2^{\alpha_2} \cdots n^{\alpha_n}\}$.

**Теорема 53.**

$$P(\alpha_1, \alpha_2, \ldots, \alpha_n) = \frac{n!}{1^{\alpha_1} 2^{\alpha_2} \cdots n^{\alpha_n} \alpha_1! \alpha_2! \cdots \alpha_n!}. \tag{2.55}$$

*Доведення.* Розглянемо розклад підстановки $f$ із циклічного класу $\{1^{\alpha_1}2^{\alpha_2}\cdots n^{\alpha_n}\}$ у вигляді добутку циклів

$$f=(x_1)(x_2)\cdots(x_{\alpha_1})(y_1,z_1)(y_2,z_2)\cdots(y_{\alpha_2},z_{\alpha_2})\cdots.$$

Довільну підстановку із цього класу можна одержати шляхом усіх можливих перестановок елементів при збереженні дужок. Циклічні перестановки елементів усередині дужок і перестановки, які переводять повністю елементи з однієї дужки в іншу з такою самою кількістю елементів, очевидно, не генерують нових підстановок. Число таких підстановок дорівнює $\{1^{\alpha_1}2^{\alpha_2}\cdots n^{\alpha_n}\}$. Помноживши цю кількість на число різних підстановок у цикловому класі $\{1^{\alpha_1}2^{\alpha_2}\cdots n^{\alpha_n}\}$, отримуємо кількість всіх підстановок $n!$. Отже,

$$P(\alpha_1,\alpha_2,\ldots,\alpha_n)1^{\alpha_1}2^{\alpha_2}\cdots n^{\alpha_n}\alpha_1!\alpha_2!\cdots\alpha_n!=n!.$$

Звідси отримуємо

$$P(\alpha_1,\alpha_2,\ldots,\alpha_n)=\tfrac{n!}{1^{\alpha_1}2^{\alpha_2}\cdots n^{\alpha_n}\alpha_1!\alpha_2!\cdots\alpha_n!}.\ \blacksquare$$

Нехай $C(n,k)$ означає кількість підстановок степеня $n$ $n$-елементної множини, які мають $k$ циклів. З попередньої теореми випливає, що

$$C(n,k)=\sum_{1\alpha_1+\cdots+n\alpha_n=n,\ \alpha_1+\cdots+\alpha_n=k,\ \alpha_i\geq 0}=\tfrac{n!}{1^{\alpha_1}2^{\alpha_2}\cdots n^{\alpha_n}\alpha_1!\alpha_2!\cdots\alpha_n!}.$$

Можна показати, що $|C(n,k)|=S_n^k$, де $S_n^k$ – числа Стірлінга першого роду.

**Застосування груп підстановок у криптографії**. Нехай $X=\{a,b,c,\ldots$ $\ldots,x,y,z\}$ – алфавіт англійської мови, літери якого лінійно упорядковані звичайним чином:

| $a$ | $b$ | $c$ | $d$ | $e$ | $f$ | $g$ | $h$ | $i$ | $j$ | $k$ | $l$ | $m$ | $n$ | $o$ | $p$ | $q$ | $r$ | $s$ | $t$ | $u$ | $v$ | $w$ | $x$ | $y$ | $z$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

Розглянемо приклади шифрування, які ґрунтуються на техніці підстановок.

а) **Шифр Цезаря**. Цей шифр відомий із давніх часів і приписують його авторство Юлію Цезарю – римському імператору. Цей шифр ґрунтується на заміні кожної літери алфавіту новою літерою цього ж алфавіту, яка розміщена на три позиції правіше.

| a | b | c | d | e | f | g | h | i | j | k | l | m |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D | E | F | G | H | I | J | K | L | M | N | O | P |
| n | o | p | q | r | s | t | u | v | w | x | y | z |
| Q | R | S | T | U | V | W | X | Y | Z | A | B | C |

Дано *відкритий текст:* “meeting will in twelve”
*криптограма:* PHHWLQJ ZLOO LQ WZHOYH

Зауважимо, що алфавіт “закручений” так, що після літери Z наступною буде йти літера A.

Якщо кожній літері припишемо числовий відповідник ($n(a) = 0, n(b) = 1, \ldots$), то цей шифр можна подати у такому вигляді. Кожна літера відкритого тексту $p$ замінюється літерою тексту зашифрованого $q$ на підставі правила

$$q = f(p) = n(p) + 3 \pmod{26}.$$

Зсув літер в алфавіті може мати довільну величину, а це означає, що загальний вигляд алгоритму є таким:

$$q = f(p) = n(p) + k \pmod{26},$$

де $k \in \{1, 2, \ldots, 26\}$. Алгоритм розшифрування досить простий:

$$p = g(q) = n(q) - k \pmod{26}.$$

Якщо відомо, що даний текст зашифрований шифром Цезаря, то його криптоаналіз не складає труднощів. Цей криптоаналіз можна виконати як методом частотного аналізу, так і методом простого перебору, випробовуючи 25 можливих ключів.

Нижче на рис. 2.9.1 показано результати застосування методу перебору до зашифрованого тексту. У даному випадку відкритий текст дістаємо на третьому кроці перебору.

| Ключ | PHHWLQJ | ZLOO | LQ | WZHOYH |
|---|---|---|---|---|
| 1 | oggvkpi | yknn | kp | vygnxg |
| 2 | nffujog | xjmm | jo | uxfmwf |
| 3 | meeting | will | in | twelve |
| 4 | lddshmc | vhkk | hm | svdkud |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 25 | ... | ... | ... | ... |

Рис. 2.9.1. Криптоаналіз шифру Цезаря методом перебору

Застосування методу перебору стало можливим, оскільки:

1) відомий алгоритм шифрування і розшифрування;

2) існує тільки 25 можливих ключів;

3) мова відкритого тексту відома і легко розпізнається.

Метод підстановки стає практичним, якщо існує великий простір для вибору ключів. Наприклад, американський стандартний алгоритм DES використовує 56-бітовий ключ, що дає простір вибору $2^{56} > 7 \cdot 10^{16}$ можливих ключів.

Істотною є також третя риса. Якщо не відома мова відкритого тексту, то можемо не розпізнати результати розшифрування. Більше того, криптограма може бути яким-небудь способом скорочена або стиснена, а це додаткові перешкоди на шляху розшифрування. Якщо стиснемо файл, а потім його зашифруємо простим шифром підстановки, то ВТ може бути не розпізнаний цим методом.

Криптоаналіз шифру Цезаря можна ускладнити, якщо поміняти звичний порядок літер в алфавіті. Наприклад, нехай порядок літер змінено таким чином:

| **a** | *b* | *c* | **d** | *e* | *f* | **g** | *h* | *i* | **j** | *k* | *l* | **m** | *n* | *o* | **p** | *q* | *r* | **s** | *t* | *u* | **v** | *w* | *x* | **y** | *z* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | d | g | j | m | p | s | v | y | b | e | h | k | n | q | t | w | z | c | f | i | l | o | r | u | x |

.

Тоді шифрограма буде такою: KMMFYNS OYHH YN FOMHLM і її криптоаналіз стає складнішим.

б) **Блоковий спосіб шифрування** з використанням техніки підстановок. Шифрування слова $p = y_1 y_2 \dots y_m$ у алфавіті зі звичним порядком літер виконується таким чином:

– слово $p$ розбивається на блоки по $t$ символів у кожному блоці;

– кожний символ у блоці перетворюється за допомогою підстановок $\alpha_1, \alpha_2, \dots, \alpha_t$, кожна з яких має вигляд

$$\alpha_i(k) = k + j_i \pmod{26},$$

де $k$ – номер літери в алфавіті $X$, а $k + j_i \pmod{26}$ – її відповідник в алфавіті $X$ при підстановці $\alpha_i$, $i = 1, 2, \dots, t$.

Тоді, коли $t = 3, j_i = 3, j_2 = 7, j_3 = 10$ $\alpha_1(k) = k + 3$, $\alpha_2(k) = k + 7$, $\alpha_3(k) = k + 10$, слово $p = mee\ tin\ gwi\ lli\ ntw\ elv\ e$ перетворюється до слова

$$q = \ plo\ wpx\ jds\ oss\ qag\ hsf\ h.$$

Дійсно, літера $m$ має номер $k = 12$ і $\alpha_1(k) = 12 + 3 = 15$, а це номер літери $p$ в алфавіті $X$, літера $e$ має номер $k = 4$ і $\alpha_2(k) = 4 + 7 = 11$, а це номер літери $l$ в алфавіті $X$, літера $e$ має номер $k = 4$ і $\alpha_3(k) = 4 + 10 = 14$, а це номер літери $o$ в алфавіті $X$, літера $t$ має номер $k = 19$ і $\alpha_1(k) = 19 + 3 = 22$, а це номер літери $w$ в алфавіті $X$ і т. д.

Розшифрування виконується очевидним чином:

$$\alpha_i^{-1}(k) = \begin{cases} k - j_i, & \text{якщо } k - j_i \geq 0, \\ k - j_i + 26, & \text{якщо } k - j_i < 0. \end{cases}$$

Наприклад, літери $o$ і $g$ мають такі відповідники:

$\alpha_3^{-1}(14) = 14 - 10 = 4$, а це літера $e$, $\alpha_3^{-1}(6) = 6 - 10 + 26 = 22$, а це літера $w$.

Перевагою такого способу шифрування є те, що частота входження літер у текст шифрограми прихована, а це значно ускладнює криптоаналіз такого тексту. ♠.

**Абелеві групи**. Ці групи отримуємо шляхом додавання закону комутативності до множини тотожностей групи.

**Означення 62.** *Група* $G(A,\Omega)$ *називається* **абелевою групою**, *якщо, крім тотожних співвідношень, які визначають її як групу, ця множина включає закон комутативності для операції множення.*

Довільне слово із $G$ можна записати у вигляді

$$a_1^{n_1}a_2^{n_2}\cdots a_k^{n_k}, \tag{2.56}$$

де всі $a_j \in A$ попарно різні, $n_j$ – цілі числа, $j = 1,2,\ldots,k$. Запис слова у вигляді (2.56) називають *мультиплікативним*, але часто для запису слів абелевої групи використовують адитивний запис – у вигляді суми:

$$n_1 \cdot a_1 + n_2 \cdot a_2 + \cdots + n_k \cdot a_k. \tag{2.57}$$

Числа $n_j$ у цьому виразі називаються **коефіцієнтами**. Додавання слів вигляду (2.57) визначається як додавання коефіцієнтів при однакових елементах $a_i \in A$, а роль одиниці у такій формі запису відіграє **нуль** – нульовий елемент, тобто вираз вигляду (2.57), у якого всі коефіцієнти дорівнюють нулю. Якщо деякі $a_j$ з $A$ у виразі (2.57) відсутні, то вважається, що коефіцієнт $n_j$ при цьому $a_j$ дорівнює нулю.

**Приклади абелевих груп.** 1. Множина $M = \{-1,1\}$ є групою відносно звичайного множення чисел. Дійсно, $1\cdot 1 = 1,\ (-1)\cdot 1 = 1\cdot(-1) = -1,\ (-1)\cdot(-1) = 1$. Отже, множина $M$ замкнута відносно операції множення. Роль одиниці виконує елемент 1. Ця множина також замкнута і відносно операції взяття оберненого (елементи 1 і $-1$ обернені до самих себе). Очевидно, що операція множення асоціативна і комутативна.

2. Нехай $\mathcal{Z}_n = \{0,1,\ldots,n-1\},\ n > 1$. Задамо на множині $\mathcal{Z}_n$ операцію додавання $\oplus$ таким чином:

$$k \oplus l = \begin{cases} k+l, & \text{якщо} \quad k+l < n; \\ k+l-n, & \text{якщо} \quad k+l \geq n, \end{cases}$$

де $+$ і $-$ звичайні операції додавання і віднімання цілих чисел, $k,l \in \mathcal{Z}_n$. Множина $\mathcal{Z}_n$ з такою операцією додавання складає абелеву групу $n$-го порядку. Дійсно, роль нуля відіграє елемент 0, роль оберненого елемента до даного елемента $k$ – елемент $l$ такий, що $k+l = n$. Очевидно, що операція додавання комутативна на підставі комутативності операції додавання цілих чисел. Доведення асоціативності операції пропонується як проста вправа.

Група $\mathcal{Z}_n$ називається **групою лишків за модулем** $n$ і виникає в результаті розбиття множини цілих чисел $\mathcal{Z}$ на класи, де в один клас попадають ті і тільки ті

числа, які при діленні на $n > 1$ дають однакові остачі. Ця група зустрічалася раніше у розгляді порівнянь. Якщо позначити $rest(m, n)$ остачу від ділення числа $m$ на $n$, то операцію додавання $\oplus$ можна визначити іншим способом:

$$k \oplus l = rest(k + l, n).$$

3. Множина цілих чисел $\mathcal{Z}$ складає адитивну групу, але $\mathcal{Z}$ не є групою відносно операції множення, оскільки операція ділення у множині $\mathcal{Z}$ (операція взяття оберненого) не завжди визначена.

Множина $\mathcal{Z}_O$ цілих парних чисел складає адитивну групу і є підгрупою групи $\mathcal{Z}$. Узагалі адитивною підгрупою буде довільна множина цілих чисел, які кратні деякому заданому цілому числу $n$.

Множина непарних цілих чисел не буде групою, оскільки вона не замкнута відносно операції додавання.

4. Циклічна група $G$, тобто група, породжена одним елементом, наприклад $a$, складається з елементів вигляду $a^n$, де $n \in \mathcal{Z}$, $a^0 = e, a^1 = a$. Множення елементів визначається як додавання степенів, тобто $a^k \cdot a^l = a^{k+l}$.

**Теорема 54.** *Довільна нескінченна циклічна група ізоморфна адитивній групі цілих чисел $\mathcal{Z}$. Довільна скінченна циклічна група $n$-го порядку ізоморфна групі лишків за модулем $n$ – $\mathcal{Z}_n$.*

*Доведення.* У першому випадку відображення $h$ задається $h(a^n) = n$, а в другому – $h(a^k) = rest(k, n)$.

Покажемо, що $h$ – ізоморфізм для першого випадку.

а1) $h(a^m \cdot a^n) = h(a^{m+n}) = m + n$;

а2) $h(a^{-n}) = -n$, і оскільки $a^n$ обернений до $a^{-n}$, то умови ізоморфізму виконуються;

а3) $h(a^0) = h(e) = 0$.

Залишається показати, що $h$ – взаємно однозначне. Для цього необхідно зауважити, що із $a^m = a^n$ випливає $a^{m-n} = a^0 = e$, тобто $m = n$. ■

Другий випадок пропонується розглянути як вправу. ♠

## 2.9.3. Арифметика на основі абелевих груп

Нехай задана деяка скінченна множина цілих чисел, наприклад, $N_5 = \{0, 1, 2, 3, 4\}$. Оскільки ми хочемо побудувати адитивну абелеву групу, то ця множина обов'язково повинна включати 0. Щоб $N_5$ перетворити на групу $GN_5$, необхідно коректно задати значення для операції додавання з одним з елементів групи, скажімо з 1. Дійсно, оскільки $a + 0 = a$ для довільного $a \in GN_5$, то перший рядок таблиці додавання елементів групи визначений (табл. 2.9.1), а на підставі комутативності (оскільки $GN_5$ абелева) – і перший стовпчик цієї таблиці. Нехай, наприклад, задано $0 + 1 = 1,\ 1 + 1 = 4,\ 1+4 = 2,\ 1+2 = 3,\ 1+3 = 0$. Таке задання коректне, оскільки

гарантується єдиність результату (але єдиність результату, як буде показано нижче, не достатня умова гарантії коректності). Тепер послідовно знаходимо результати додавання з елементом 4, оскільки 4=1+1:

$$4 + 2=(1+1)+2=1+(1+2)=1+3=0, \quad 4+3=(1+1)+3=1+(1+3)=1,$$
$$4+4=(1+1)+4=1+(1+4)=1+2=3.$$

Далі знаходимо значення 4+1=2 і обчислюємо операцію додавання з елементом 2:

$$2+2=(1+4)+2=1+(4+2)=1+0=1, \quad 2+3=(1+4)+3=1+(4+3)=1+1=4,$$
$$2+4=(1+4)+4=1+(4+4)=1+3=0.$$

Потім знаходимо значення 2+1=3 і обчислюємо операцію додавання з елементом 3:

$$3+2=(1+2)+2=1+(2+2)=1+1=4, \quad 3+3=(1+2)+3=1+(2+3)=1+4=2,$$
$$3+4=(1+2)+4=1+(2+4)=1+0=1.$$

Оскільки 3+1=0, то на цьому процес побудови закінчується, тому що для 0 вже результати операції визначені.

Заносимо ці значення в табл. 2.9.2 і на цьому закінчуємо побудову групи $GN_5$.

Таблиця 2.9.1

| + | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 |
| 1 | 1 | 4 | 3 | 0 | 2 |
| 2 | 2 | 3 | | | |
| 3 | 3 | 0 | | | |
| 4 | 4 | 2 | | | |

,

Таблиця 2.9.2

| + | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 |
| 1 | 1 | 4 | 3 | 0 | 2 |
| 2 | 2 | 3 | 1 | 4 | 0 |
| 3 | 3 | 0 | 4 | 2 | 1 |
| 4 | 4 | 2 | 0 | 1 | 3 |

,

Таблиця 2.9.3

| + | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 |
| 1 | 1 | 3 | 0 | 4 | 2 |
| 2 | 2 | 0 | 4 | 1 | 3 |
| 3 | 3 | 4 | 1 | 2 | 0 |
| 4 | 4 | 2 | 3 | 0 | 1 |

.

Аналогічно можна задати і довільну іншу групу $GN_5$. Дійсно, задамо такий рядок таблиці додавання:

$$1+0 = 1,\ 1 + 1 = 3,\ 1+2=0,\ 1+3=4,\ 1+4=2.$$

За цим рядком знаходимо групу з таблицею додавання 2.9.3.

Для визначення групи можна взяти довільний її елемент. Наприклад, визначимо групу $GN_5$, елементами якої є 0, 2, 3, 5, 6, за допомогою додавання з елементом 3:

Таблиця 2.9.4

| + | 0 | 3 | 5 | 6 | 2 |
|---|---|---|---|---|---|
| 0 | 0 | 3 | 5 | 6 | 2 |
| 3 | 3 | 5 | 6 | 2 | 0 |
| 5 | 5 | 6 | | | |
| 6 | 6 | 2 | | | |
| 2 | 2 | 0 | | | |

,

Таблиця 2.9.5

| + | 0 | 3 | 5 | 6 | 2 |
|---|---|---|---|---|---|
| 0 | 0 | 3 | 5 | 6 | 2 |
| 3 | 3 | 5 | 6 | 2 | 0 |
| 5 | 5 | 6 | 2 | 0 | 3 |
| 6 | 6 | 2 | 0 | 3 | 5 |
| 2 | 2 | 0 | 3 | 5 | 6 |

.

Для побудови групи $GN_k$ мало вимагати тільки однозначності операції додавання. Якщо визначити додавання у групі як $0+1=$ $=1,\ 1+1=0,\ 1+2=3,\ 1+3=4,\ 1+4=2$, то, обчислюючи 1+3, отримаємо

$$1+3=1+(1+2)=(1+1)+2=0+2=2\neq 4,$$

що не збігається з визначеним вище. Зазначимо, що так визначена операція додавання не охоплює весь цикл елементів групи, тому що має елемент скінченного порядку $2<5$ (1+1=0).

Усі три групи, побудовані вище, циклічні на підставі теореми Лагранжа (вони мають порядок 5). У перших двох групах твірним був елемент 1, а в третій групі – елемент 3. Неважко переконатися, що всі три групи ізоморфні. Дійсно, бієктивне відображення для перших двох груп має вигляд

$$f(0)=0,\ f(1)=1,\ f(4)=3,\ f(2)=4,\ f(3)=2,$$

а ізоморфізм першої і третьої груп визначається таким відображенням:

$$f(0)=0,\ f(1)=3,\ f(4)=5,\ f(2)=6,\ f(3)=2.$$

Поставимо у відповідність операції додавання з елементом групи $a_1$, за допомогою якого визначається група, підстановку

$$f_{a_1}=\begin{pmatrix} 0 & a_1 & a_2 & \dots & a_{k-1} \\ a_1 & a_{i_1} & a_{i_2} & \dots & a_{i_k} \end{pmatrix}.$$

Ця підстановка означає, що $f_{a_1}(0)=0+a_1=a_1, f_{a_1}(a_1)=a_1+$ $+a_1=a_{i_1}, f_{a_1}(a_{i_1})=a_{i_1}+a_1=a_{i_j}, f_{a_1}(a_{i_j})=a_{i_j}+a_1=a_{i_l}$ і т. д.

Назвемо групу $GN_k$ **повноциклічною**, якщо підстановка $f_{a_1}$ є повним циклом довжини $k$. Справедлива така теорема.

**Теорема 55.** *Усі скінченні повноциклічні абелеві групи одного порядку ізоморфні між собою.*

*Доведення* випливає з того, що всі повноциклічні групи циклічні, а скінченні циклічні групи одного порядку ізоморфні.

**Пряма сума абелевих груп** – це операція, яка дозволяє будувати абелеві групи більших порядків з абелевих груп менших порядків.

**Означення 63** *Прямою (зовнішньою) сумою адитивних абелевих груп $G_1, \ldots, G_m$ називається абелева група $G = G_1 \oplus \cdots \oplus G_m$, яка складається зі всіх послідовностей $(a_1, \ldots, a_m)$, де $a_i \in G_i$, з операцією додавання $(a_1, \ldots, a_m) \oplus (b_1, \ldots, b_m) = (a_1 +_1 b_1, \ldots, a_m +_m b_m)$.*

**Приклад прямої суми абелевих груп**. Нехай маємо групи лишків $\mathcal{Z}_3$ і $\mathcal{Z}_4$ за модулем 3 і 4 відповідно. Тоді абелева група $G = \mathcal{Z}_3 \oplus \mathcal{Z}_4$ має носій, елементам якого поставлені у відповідність числа:

(0,0) – 0, (1,1) – 1, (2,2) – 2, (0,3) – 3, (1,0) – 4, (2,1) – 5,
(0,2) – 6, (1,3) – 7, (2,0) – 8, (0,1) – 9, (1,2) – 10, (2,3) – 11.

Операція додавання в $G$ має вигляд $(a, b) \oplus (c, d) = ((a + c) \pmod 3, (b + d) \pmod 4)$, а таблиця додавання за такої відповідності набуває вигляду

| $\oplus$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 0 |
| 2 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 0 | 1 |
| 3 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 0 | 1 | 2 |
| 4 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 0 | 1 | 2 | 3 |
| 5 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 0 | 1 | 2 | 3 | 4 |
| 6 | 6 | 7 | 8 | 9 | 10 | 11 | 0 | 1 | 2 | 3 | 4 | 5 |
| 7 | 7 | 8 | 9 | 10 | 11 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 8 | 8 | 9 | 10 | 11 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 9 | 9 | 10 | 11 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 10 | 10 | 11 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 11 | 11 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

.

Це таблиця додавання групи лишків $\mathcal{Z}_{12}$ за модулем 12, а відображення $\varphi(x) = (x \pmod 3, x \pmod 4)$ – ізоморфізм груп $\mathcal{Z}_{12}$ і $G$ на підставі китайської теореми про остачі. ♠

Далі будуть розглянуті інші застосування груп, зокрема, для розв'язання проблеми передачі ключів.

### 2.9.4. Кільця

Ця алгебра будується шляхом розширення сигнатури операцій абелевої групи та множини тотожних співвідношень для цих операцій. До множини операцій додається операція множення, яка пов'язується з операцією додавання законами дистрибутивності.

**Означення 64.** *Універсальна алгебра $\mathcal{Q} = G(A, \Omega)$ називається* **кільцем**, *якщо вона*

а) *абелева група відносно додавання;*

б) *групоїд відносно множення;*

в) *задовольняє закон дистрибутивності, тобто для довільних її елементів $a, b, c \in A$ виконується*

$$a(b + c) = (ab) + (ac), \qquad (a + b)c = (ac) + (bc).$$

Це означає, що $\Omega$ включає чотири операції: бінарні операції додавання і множення, унарну операцію взяття оберненого відносно операції додавання і нульарну операцію, яка фіксує нульовий елемент абелевої групи кільця. Цей нульовий елемент називається **нулем** кільця.

Множення у кільці зводиться, на підставі законів дистрибутивності, до множення елементів з $A$, яке виконується за правилом множення слів у групоїді.

Із законів для операцій кільця випливають співвідношення, які дає твердження 6.

**Твердження 6.** *У довільному кільці $K$ для довільних його елементів $a, b, c$ справедливі такі закони:*

а) $a(b - c) = ab - ac$, $(b - c)a = ba - ca$;

б) $a0 = 0a = 0$; в) $(-a)b = a(-b) = -ab$, $(-a)(-b) = ab$.

*Доведення.* а) На підставі комутативності й асоціативності операції додавання можна записати $c+(b-c) = b$. Домноживши обидві частини цього рівняння зліва на $a$, дістаємо

$$a(c + (b - c)) = ac + a(b - c) = ab.$$

Звідси, $(ac + a(b - c)) - ac = ab - ac = a(b - c)$, знову ж таки на підставі комутативності й асоціативності операції додавання.

Аналогічно доводиться і другий закон.

б) Нехай $b$ – довільний елемент кільця. Тоді, на підставі доведеного вище пункту а), маємо $a0 = a(b - b) = ab - ab = 0$.

в) *Доведення* пропонується як вправа.

Зауважимо, що обернене твердження до пункту б) в довільному кільці не виконується, тобто існують такі кільця, в яких є відмінні від нуля елементи $a, b$, добуток яких дорівнює нулю ($ab = 0$). Якщо такі елементи в кільці є, то вони називаються **дільниками нуля**.

**Означення 65.** *Кільце називається* **асоціативним (комутативним)**, *якщо його операція множення асоціативна* (*комутативна*), *і називається* **кільцем з одиницею**, *коли воно має одиничний елемент відносно операції множення.*

*Кільце називається* **асоціативно-комутативним**, *якщо воно асоціативне і комутативне одночасно.*

Множення в асоціативному кільці виконується за правилом множення слів у напівгрупі, а сама напівгрупа називається **мультиплікативною напівгрупою** асоціативного кільця.

Елементи асоціативно-комутативного кільця з множиною вільних твірних $X$ є поліномами від елементів із $X$ із цілими коефіцієнтами. Тому таке кільце часто називають просто **кільцем поліномів над** $X$.

**Приклади кілець.** 1. Множина цілих чисел, кратних числу $k \geq 2$, асоціативно-комутативне кільце без одиниці.

2. Прикладом асоціативно-комутативного кільця з одиницею є **кільце лишків** $\mathcal{Z}_n$ **за модулем** числа $n$. Дійсно, нехай $m \neq 0$ – фіксоване ціле число.

Визначивши операції додавання і множення так, що $\forall x, y \in \{0, 1, \ldots, n-1\}$

$$x \oplus y = \begin{cases} x + y & \text{якщо} \quad x + y < n, \\ x + y - n & \text{якщо} \quad x + y \geq n, \end{cases} \qquad x \odot y = \begin{cases} xy & \text{якщо} \quad xy < n, \\ rest(xy, m) & \text{якщо} \quad xy \geq n, \end{cases}$$

дістаємо кільце лишків з одиницею за модулем $n$. Нулем кільця служить 0, одиницею – 1. Закони асоціативності і комутативності для операцій додавання і множення випливають з виконуваності цих законів у множині цілих чисел $\mathcal{Z}$.

Наприклад, якщо $n = 6$, то введені операції можна подати у вигляді табл. 2.9.6, 2.9.7.

Таблиця 2.9.6

| $\oplus$ | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 2 | 3 | 4 | 5 | 0 |
| 2 | 2 | 3 | 4 | 5 | 0 | 1 |
| 3 | 3 | 4 | 5 | 0 | 1 | 2 |
| 4 | 4 | 5 | 0 | 1 | 2 | 3 |
| 5 | 5 | 0 | 1 | 2 | 3 | 4 |

,

Таблиця 2.9.7

| $\odot$ | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 | 5 |
| 2 | 0 | 2 | 4 | 0 | 2 | 4 |
| 3 | 0 | 3 | 0 | 3 | 0 | 3 |
| 4 | 0 | 4 | 2 | 0 | 4 | 2 |
| 5 | 0 | 5 | 4 | 3 | 2 | 1 |

.

Як видно з табл. 2.9.7, це кільце має дільники нуля 2, 3, 4.

3. Прикладом асоціативного, але не комутативного кільця з одиницею $GL(n)$, є множина квадратних матриць розмірності $n \times n$ над множиною дійсних чисел $D$. З лінійної алгебри відомо, що відносно операції додавання $GL(n)$ – абелева група, а відносно операції множення – напівгрупа з одиницею.

Нулем у цьому кільці є нульова матриця 0, а одиницею – одинична матриця $E$. Це кільце з дільниками нуля, у чому легко переконатися перемножуючи ненульові матриці

$$A = \begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 0 \\ 2 & 1 \end{pmatrix}.$$

4. Множина квадратних матриць, елементами яких є раціональні числа, відносно звичайної операції додавання й операції множення $A \cdot B = \frac{1}{2}(AB + BA)$ – комутативне неасоціативне кільце з одиницею.

5. Множина векторів тривимірного векторного простору відносно звичайної операції додавання векторів і відносно операції векторного добутку векторів є прикладом неасоціативного і некомутативного кільця. ♠

Асоціативно-комутативне кільце без дільників нуля називається **областю цілісності**.

Областями цілісності, як легко переконатися, будуть кільця цілих і раціональних чисел – $\mathcal{Z}$ і $\mathcal{Q}$. Областю цілісності буде також і кільце поліномів.

Дійсно, нехай $ACR$ – довільне асоціативно-комутативне кільце. Розглянемо всі можливі поліноми вигляду

$$a_0 + a_1 \cdot x_1^1 + a_2 \cdot x_1^2 + \cdots + a_n \cdot x_1^n,$$

де $n_i \in \mathcal{N}$, відносно невідомого $x_1$ з коефіцієнтами $a_i$ із $ACR$, $i = 1, \ldots, n$. Якщо $a_n \neq 0$, то число $n$ називається *степенем* цього полінома. Визначаючи додавання і множення поліномів звичайним шляхом так, як це прийнято в курсі вищої алгебри, отримуємо **кільце поліномів** $R[x_1]$ від невідомого $x_1$ над кільцем $ACR$. Нулем кільця $R[x_1]$ служить поліном, усі коефіцієнти якого дорівнюють нулю.

Аналогічно можна визначити кільце поліномів $R[x_1, \ldots, x_n]$ від довільного скінченного числа невідомих як кільце поліномів від одного невідомого $x_n$ над кільцем $R[x_1, \ldots, x_{n-1}]$. Справедлива така теорема.

**Теорема 56.** *Якщо $ACR$ – область цілісності, то $R[x_1, \ldots, x_n]$ – теж область цілісності.*

Вище було визначено поняття дільника нуля в кільці. Подібно до поняття дільника нуля існує і поняття дільника одиниці.

**Означення 66.** *Нехай $G$ – довільне кільце і $a, b \in G$. Елемент $a$ називається дільником $b$ або "$a$ ділить $b$" або "$b$ кратне $a$", якщо рівняння $ax{=}b$ має хоча б один розв'язок.*

*Елемент $a \in G$ називається дільником одиниці, якщо він ділить усі елементи кільця $G$.*

Термін дільник одиниці має таке пояснення: якщо рівняння $ax = 1$ має розв'язок, то рівняння $ay = b$ буде мати розв'язок для довільного $b \in G$. Дійсно, якщо $x = c$ – розв'язок рівняння $ax = 1$, то $y = cb$ буде розв'язком рівняння $ay = b$ (що перевіряється простою підстановкою).

Неважко показати, що множина всіх дільників одиниці кільця $G$ є абелевою групою відносно операції множення. Ця група позначається $MGN$ і називається *мультиплікативною групою кільця.* Характеристику цій групі дає така теорема.

**Теорема 57.** *Скінченне асоціативно-комутативне кільце з одиницею $k$-го порядку, адитивна група якого повноциклічна, має $\varphi(k)$ дільників одиниці, де $\varphi$ – функція Ойлера.*

*Доведення.* Розглянемо $\mathcal{Z}_k$ – кільце лишків за модулем $k$. Неважко переконатися, що адитивна группа цього кільця повноциклічна. Дільниками одиниці в цьому кільці будуть елементи, які взаємно прості з модулем кільця $k$, а кількість таких елементів, як відомо, дорівнює значенню функції Ойлера $\varphi(k)$. На підставі теореми 55 існує ізоморфізм між кільцем лишків $\mathcal{Z}_k$ і кільцем $G_k$, який є

продовженням ізоморфізму між адитивними групами цих кілець. Звідси випливає справедливість твердження теореми. ■

Ілюстрацією теореми 57 є кільце $G_6$, задане наведеними далі табл. 2.9.8 і 2.9.9. Оскільки кільце лишків за модулем 6 має два елементи 1 і 5, які взаємно прості з модулем 6, а цим елементам при ізоморфізмі відповідають елементи 1 і 2, то з табл. 2.9.9 видно, що 1 і 2 – дільники одиниці в цьому кільці.

### 2.9.5. Застосування кілець у криптографії

Під час розгляду абелевих груп був побудований приклад "нетрадиційної арифметики" для операції додавання. Використовуючи ідею такої побудови, її можна розширити і на кільця. Скористаємося для цього прикладом.

Нехай задано рядок додавання з одиницею групи $GN_6$, який будемо називати **визначальним рядком**:

$$0{+}1{=}1, \quad 1{+}1{=}3, \quad 1{+}3{=}5, \quad 1{+}5{=}4, \quad 1{+}4{=}2, \quad 1{+}2{=}0,$$

за яким побудована таблиця додавання (табл. 2.9.8). Алгоритм побудови рядка додавання з одиницею наведено в підрозділі 3.3.

Розглянемо, як довизначається група $GN_6$ до кільця з одиницею. Роль одиниці буде відігравати 1. На підставі аксіом кільця і твердження 6 маємо: для довільного елемента $a$ із $GN_6$ $a \cdot 1 = 1 \cdot a = a$, $a \cdot 0 = 0 \cdot a = 0$.

Таблиця 2.9.8

| + | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 3 | 0 | 5 | 2 | 4 |
| 2 | 2 | 0 | 4 | 1 | 5 | 3 |
| 3 | 3 | 5 | 1 | 4 | 0 | 2 |
| 4 | 4 | 2 | 5 | 0 | 3 | 1 |
| 5 | 5 | 4 | 3 | 2 | 1 | 0 |

,

Таблиця 2.9.9

| · | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 | 5 |
| 2 | 0 | 2 | 1 | 4 | 3 | 5 |
| 3 | 0 | 3 | 4 | 4 | 3 | 0 |
| 4 | 0 | 4 | 3 | 3 | 4 | 0 |
| 5 | 0 | 5 | 5 | 0 | 0 | 5 |

.

Таким чином, два рядки і два стовпчики таблиці множення визначено. Далі, за таблицею додавання і законом дистрибутивності, знаходимо таблицю для операції множення (табл. 2.9.9). Дійсно,

$$3 \cdot 2 = (1+1) \cdot 2 = 2+2 = 4; \qquad 3 \cdot 3 = (1+1) \cdot 3 = 3+3 = 4;$$
$$3 \cdot 4 = (1+1) \cdot 4 = 4+4 = 3, \qquad 3 \cdot 5 = (1+1) \cdot 5 = 5+5 = 0.$$

Далі, оскільки 1+3=5, то дістаємо

$$5 \cdot 2 = (1+3) \cdot 2 = 2 + 3 \cdot 2 = 2 + 4 = 5;$$
$$5 \cdot 3 = (1+3) \cdot 3 = 3 + 3 \cdot 3 = 3 + 4 = 0;$$
$$5 \cdot 4 = (1+3) \cdot 4 = 4 + 3 \cdot 4 = 4 + 3 = 0;$$
$$5 \cdot 5 = (1+3) \cdot 5 = 5 + 3 \cdot 5 = 5 + 0 = 5.$$

За таблицею додавання 5+1 = 4, і це дає змогу знайти значення операції множення з елементом 4, аналогічно знаходимо значення операції множення з елементом 2=1+4 і решту елементів табл. 2.9.9. Оскільки 2+1=0, то це означає, що вся таблиця множення побудована. Із симетричності таблиці випливає, що побудована множина елементів задовольняє закон комутативності. Крім того, легко перевірити, що ця множина задовольняє також закон асоціативності, тобто побудована алгебра $G_6$ – асоціативно-комутативне кільце з одиницею.

Побудова скіненного кільця $G_k$ відбувається таким чином: задати визначальний рядок додавання з одиницею кільця і виконати наступні алгоритми:

**ADD-TAB**$G(1,k)$
0) Задекларувати масив $T_+[k \times k]$.
1) Занести в $T_+$ у перший рядок результати додавання з нулем.
2) Занести в $T_+$ у другий рядок визначальний рядок.
3) Покласти $c = 1$.
4) Взяти в $T_+$ елемент $c' = c + 1$.
5) Для всіх $x \in G_k$ занести в $T_+$ у рядок із номером $c'$ суми $c' + x$;
6) Покласти $c = c'; c' = c + 1$. Якщо $c' = 0$ то СТОП, інакше на крок 5).

**MUL-TAB**$G(1,k)$
0) Задекларувати масив $T_\times[k \times k]$.
1) Занести у $T_\times$ в рядки з номером 0 і 1 результати множення на 0 і на 1.
2) Покласти $c = 1$.
3) Взяти в $T_+$ елемент $c' = c + 1$.
4) Для всіх $x \in G_k$ занести у рядок із номером $c'$ масиву $T_\times$ добутки $c' \cdot x$.
5) Покласти $c = c'; c' = c + 1$. Якщо $c' = 0$, то СТОП, інакше на крок 4).

Правильність цього алгоритму обґрунтовується тим, що елементи $c_1 = 0+1, c_2 = c_1+1, \ldots, c_k = c_{k-1}+1 = 0$ пробігають усі елементи

абелевої групи на підставі того, що рівняння $x + 1 = b$ в групі має єдиний розв'язок.

Складність першого алгоритму $O(k^2 \log k)$, а другого – $O((k \log k)^2)$, оскільки складність виконання операції додавання натуральних чисел не більших $k$ – $O(\log k)$, а складність виконання операції множення таких чисел – $O(\log^2 k)$ [12] і цих операцій $k^2$.

Нехай алфавіт $X$ складається з $m$ символів, які певним чином упорядковані числами з множини $M = \{0, 1, \ldots, m-1\}$, а кільце $G_k = \{a_1, a_2, \ldots, a_k\}$ має $k$ елементів, де $k \geq m$. Визначимо, наприклад, таку сюр'єкцію $g : G \to M$:

$$g(a_i) = \begin{cases} m_1, \text{ якщо } i = 0, \\ g(a_{i-1}) + 1, \text{ якщо } i > 0, \end{cases} \tag{2.58}$$

де $m_1 \in M$ вибирається довільним чином, а номер класу $i$ обчислюється за модулем $m$. Це означає, що $m_1$ потрапляє у клас із номером 0, $m_1 + 1$ – у клас із номером 1 і т. д.

Нехай потрібно зашифрувати повідомлення $b = x_1 x_2 \ldots x_s$, де $x_i \in X$. Повідомленню $b$ відповідає його цифрове зображення $\bar{b} = n_1 n_2 \ldots n_s$, де $n_j = g(a_j)$, а $n_j$ – номер символу $x_i \in X$ із відповідного класу. Сюр'єкція $g$ визначає на елементах кільця $G_k$ відношення еквівалентності $\sim$. Якщо $k = m$, то сюр'єкція буде бієкцією, оскільки класи еквівалентності будуть одноелементними (приклад такої сюр'єкції наводиться нижче). Щоб приховати частоту появи символів у повідомленні, слід вибирати $k > m$, причому $k$ повинно бути достатньо великим для збільшення кількості елементів у класах еквівалентності (елементи у класах відіграють роль гомофонів при шифруванні). Вибираємо ключове слово $p$, яке може бути довільним словом в алфавіті $X$ і яке має цифрове зображення (застосуванням відображення $g$) $\bar{p} = k_1 k_2 \ldots k_r$.

Із цих побудов випливає такий спосіб шифрування з використанням властивостей кільця:

**RG-EN**$(T_+[1, k], g, p)$.

1) Задати рядок додавання з одиницею $T_+[1, k]$ (вибирається довільним чином, але так щоб група кільця була повноциклічною), сюр'єкцію $g$ і $p$.

2) Будуємо кільце (або тільки групу) порядку $k$ ($k \geq |X|$).

3) Побудувати шифрограму $d = d_1 d_2 \dots d_s$ (посимвольно) для $\bar{b} = n_1 n_2 \dots n_s$ (наприклад) за правилом $d_i = k_i + g(a_i)$, де $k_i$ – номер символу ключового слова, $g(a_i)$ – образ номера символу $x_i \in X$ повідомлення $b$, а $+$ – операція додавання кільця $G_k$ (шифрування може використовувати й операцію множення кільця).

4) Вислати абоненту по відкритому каналу шифрограму $d$, а по закритому каналу – трійку $(T_+[1,k], g, p)$.

Розшифрування виконується таким алгоритмом:

**RG-DE**$(d)$.

1) Абонент за рядком $T_+[1,k]$ будує кільце.

2) По отриманій шифрограмі $d$ і ключовому слову $p$ (в їхньому цифровому поданні) для кожної пари відповідних символів розв'язує рівняння $k_i + x_i = d_i$ ($k_i x_i = d_i$) за допомогою таблиць операцій кільця, де $k_i$ – номер символу ключового слова, $d_i$ – символ шифрограми.

3) Знаходить $m_i$ номер класу, який відповідає символу відкритого тексту $x_i$, і таким чином відкриває початковий текст.

Як зазначалося, при шифруванні тексту можна використовувати як таблицю додавання, так і таблицю множення. Ця обставина дозволяє одній парі літер ставити у відповідність елемент таблиці додавання, а наступній парі літер, яка повторюється, – відповідник у таблиці множення (тобто відповідником буде елемент $m = i \cdot j$). Але це можливо лише у випадку, коли аргументи добутку є дільниками одиниці в кільці $G_k$.

Збільшення кількості замін різними відповідниками однієї і тієї ж літери дає можливість використовувати так звані гомофонічні шифри, про які йтиметься далі.

Варто зазначити, що сюр'єкції можна визначати довільним чином, але потрібно при цьому враховувати частотні характеристики символів мови.

**Приклад 2.9.4.** Розглянемо на прикладі адитивної групи кільця $G_{25}$ застосування алгоритму шифрування і розшифрування.

Нехай літери алфавіту англійської мови перенумеровані природним чином:

Таблиця цифрових відповідників символів алфавіту

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i/j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |

Задаючи сюр'єкцію відображенням (2.58) з початковим значенням $m_1 = 7$ (яка в цьому випадку буде бієкцією), знаходимо такі відповідності між символами алфавіту:

Таблиця сюр'єктивних цифрових відповідників символів алфавіту

| 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 | 12 | 14 | 16 | 18 | 20 | 24 | 22 | 23 | 0 | 1 | 6 | 8 | 10 | 2 | 4 | 3 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i/j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |

Зашифруємо в адитивній групі цього кільця текст "UKRPROGTWENTY". Для цього вибираємо ключове слово "IPRSP" і знаходимо пари літер та їхні числові відповідники в таблиці додавання, які містяться на перетині відповідного рядка і стовпчика.

Застосовуючи алгоритми **ADD-TAB-**$G(1{,}25)$ і **MUL-TAB-**$G(1{,}25)$, знаходимо таблиці операцій кільця $G_{25}$, які визначаються підстановкою:

$$f = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \\ 1 & 6 & 4 & 5 & 3 & 7 & 8 & 9 & 10 & 11 & 2 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 24 & 12 & 23 & 0 & 22 \end{pmatrix}.$$

Таблиця додавання кільця $G_{25}$

| $\oplus$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 1 | 1 | 6 | 4 | 5 | 3 | 7 | 8 | 9 | 10 | 11 | 2 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 24 | 12 | 23 | 0 | 22 |
| 2 | 2 | 4 | 9 | 13 | 11 | 15 | 3 | 17 | 5 | 19 | 7 | 21 | 24 | 12 | 22 | 14 | 23 | 16 | 0 | 18 | 1 | 20 | 8 | 10 | 6 |
| 3 | 3 | 5 | 13 | 17 | 15 | 19 | 7 | 21 | 9 | 12 | 11 | 14 | 23 | 16 | 0 | 18 | 1 | 20 | 6 | 24 | 8 | 22 | 2 | 4 | 10 |
| 4 | 4 | 3 | 11 | 15 | 13 | 17 | 5 | 19 | 7 | 21 | 9 | 12 | 22 | 14 | 23 | 16 | 0 | 18 | 1 | 20 | 6 | 24 | 10 | 2 | 8 |
| 5 | 5 | 7 | 15 | 19 | 17 | 21 | 9 | 12 | 11 | 14 | 13 | 16 | 0 | 18 | 1 | 20 | 6 | 24 | 8 | 22 | 10 | 23 | 4 | 3 | 2 |
| 6 | 6 | 8 | 3 | 7 | 5 | 9 | 10 | 11 | 2 | 13 | 4 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 24 | 12 | 22 | 14 | 0 | 1 | 23 |
| 7 | 7 | 9 | 17 | 21 | 19 | 12 | 11 | 14 | 13 | 16 | 15 | 18 | 1 | 20 | 6 | 24 | 8 | 22 | 10 | 23 | 2 | 0 | 3 | 5 | 4 |
| 8 | 8 | 10 | 5 | 9 | 7 | 11 | 2 | 13 | 4 | 15 | 3 | 17 | 18 | 19 | 20 | 21 | 24 | 12 | 22 | 14 | 23 | 16 | 1 | 6 | 0 |
| 9 | 9 | 11 | 19 | 12 | 21 | 14 | 13 | 16 | 15 | 18 | 17 | 20 | 6 | 24 | 8 | 22 | 10 | 23 | 2 | 0 | 4 | 1 | 5 | 7 | 3 |
| 10 | 10 | 2 | 7 | 11 | 9 | 13 | 4 | 15 | 3 | 17 | 5 | 19 | 20 | 21 | 24 | 12 | 22 | 14 | 23 | 16 | 0 | 18 | 6 | 8 | 1 |
| 11 | 11 | 13 | 21 | 14 | 12 | 16 | 15 | 18 | 17 | 20 | 19 | 24 | 8 | 22 | 10 | 23 | 2 | 0 | 4 | 1 | 3 | 6 | 7 | 9 | 5 |
| 12 | 12 | 14 | 24 | 23 | 22 | 0 | 16 | 1 | 18 | 6 | 20 | 8 | 7 | 10 | 9 | 2 | 11 | 4 | 13 | 3 | 15 | 5 | 19 | 21 | 17 |
| 13 | 13 | 15 | 12 | 16 | 14 | 18 | 17 | 20 | 19 | 24 | 21 | 22 | 10 | 23 | 2 | 0 | 4 | 1 | 3 | 6 | 5 | 8 | 9 | 11 | 7 |
| 14 | 14 | 16 | 22 | 0 | 23 | 1 | 18 | 6 | 20 | 8 | 24 | 10 | 9 | 2 | 11 | 4 | 13 | 3 | 15 | 5 | 17 | 7 | 21 | 12 | 19 |
| 15 | 15 | 17 | 14 | 18 | 16 | 20 | 19 | 24 | 21 | 22 | 12 | 23 | 2 | 0 | 4 | 1 | 3 | 6 | 5 | 8 | 7 | 10 | 11 | 13 | 9 |
| 16 | 16 | 18 | 23 | 1 | 0 | 6 | 20 | 8 | 24 | 10 | 22 | 2 | 11 | 4 | 13 | 3 | 15 | 5 | 17 | 7 | 19 | 9 | 12 | 14 | 21 |
| 17 | 17 | 19 | 16 | 20 | 18 | 24 | 21 | 22 | 12 | 23 | 14 | 0 | 4 | 1 | 3 | 6 | 5 | 8 | 7 | 10 | 9 | 2 | 13 | 15 | 11 |
| 18 | 18 | 20 | 0 | 6 | 1 | 8 | 24 | 10 | 22 | 2 | 23 | 4 | 13 | 3 | 15 | 5 | 17 | 7 | 19 | 9 | 21 | 11 | 14 | 16 | 12 |
| 19 | 19 | 21 | 18 | 24 | 20 | 22 | 12 | 23 | 14 | 0 | 16 | 1 | 3 | 6 | 5 | 8 | 7 | 10 | 9 | 2 | 11 | 4 | 15 | 17 | 13 |
| 20 | 20 | 24 | 1 | 8 | 6 | 10 | 22 | 2 | 23 | 4 | 0 | 3 | 15 | 5 | 17 | 7 | 19 | 9 | 21 | 11 | 12 | 13 | 16 | 18 | 14 |
| 21 | 21 | 12 | 20 | 22 | 24 | 23 | 14 | 0 | 16 | 1 | 18 | 6 | 5 | 8 | 7 | 10 | 9 | 2 | 11 | 4 | 13 | 3 | 17 | 19 | 15 |
| 22 | 22 | 23 | 8 | 2 | 10 | 4 | 0 | 3 | 1 | 5 | 6 | 7 | 19 | 9 | 21 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 24 | 18 |
| 23 | 23 | 0 | 10 | 4 | 2 | 3 | 1 | 5 | 6 | 7 | 8 | 9 | 21 | 11 | 12 | 13 | 14 | 16 | 15 | 17 | 18 | 19 | 24 | 22 | 20 |
| 24 | 24 | 22 | 6 | 10 | 8 | 2 | 23 | 4 | 0 | 3 | 1 | 5 | 17 | 7 | 19 | 9 | 21 | 11 | 12 | 13 | 14 | 15 | 18 | 20 | 16 |

Таблиця множення кільця $G_{25}$

| $\odot$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 2 | 0 | 2 | 0 | 9 | 2 | 19 | 9 | 18 | 19 | 0 | 18 | 2 | 9 | 9 | 19 | 19 | 18 | 18 | 0 | 0 | 2 | 2 | 19 | 18 | 9 |
| 3 | 0 | 3 | 9 | 23 | 12 | 4 | 17 | 15 | 20 | 18 | 8 | 6 | 16 | 7 | 1 | 21 | 5 | 22 | 19 | 2 | 24 | 13 | 11 | 14 | 10 |
| 4 | 0 | 4 | 2 | 12 | 11 | 22 | 13 | 10 | 14 | 9 | 23 | 21 | 6 | 24 | 5 | 8 | 17 | 7 | 18 | 19 | 1 | 20 | 15 | 16 | 3 |

| | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | 0 | 5 | 19 | 4 | 22 | 17 | 21 | 24 | 23 | 2 | 3 | 15 | 11 | 20 | 16 | 10 | 6 | 13 | 9 | 18 | 14 | 8 | 7 | 12 | 1 |
| 6 | 0 | 6 | 9 | 17 | 13 | 21 | 10 | 14 | 4 | 18 | 5 | 24 | 7 | 23 | 11 | 1 | 15 | 8 | 19 | 2 | 12 | 3 | 20 | 22 | 16 |
| 7 | 0 | 7 | 18 | 15 | 10 | 24 | 14 | 4 | 6 | 19 | 11 | 23 | 8 | 5 | 13 | 12 | 20 | 1 | 2 | 9 | 17 | 16 | 3 | 21 | 22 |
| 8 | 0 | 8 | 19 | 20 | 14 | 23 | 4 | 6 | 7 | 2 | 13 | 5 | 1 | 11 | 10 | 17 | 3 | 12 | 9 | 18 | 15 | 22 | 16 | 24 | 21 |
| 9 | 0 | 9 | 0 | 18 | 9 | 2 | 18 | 19 | 2 | 0 | 19 | 9 | 18 | 18 | 2 | 2 | 19 | 19 | 0 | 0 | 9 | 9 | 2 | 19 | 18 |
| 10 | 0 | 10 | 18 | 8 | 23 | 3 | 5 | 11 | 13 | 19 | 21 | 16 | 14 | 22 | 24 | 6 | 1 | 4 | 2 | 9 | 7 | 17 | 12 | 20 | 15 |
| 11 | 0 | 11 | 2 | 6 | 21 | 15 | 24 | 23 | 5 | 9 | 16 | 20 | 13 | 3 | 22 | 14 | 7 | 10 | 18 | 19 | 4 | 1 | 8 | 17 | 12 |
| 12 | 0 | 12 | 9 | 16 | 6 | 11 | 7 | 8 | 1 | 18 | 14 | 13 | 17 | 10 | 4 | 20 | 22 | 15 | 19 | 2 | 3 | 24 | 21 | 5 | 23 |
| 13 | 0 | 13 | 9 | 7 | 24 | 20 | 23 | 5 | 11 | 18 | 22 | 3 | 10 | 16 | 21 | 4 | 8 | 14 | 19 | 2 | 6 | 12 | 1 | 15 | 17 |
| 14 | 0 | 14 | 19 | 1 | 5 | 16 | 11 | 13 | 10 | 2 | 24 | 22 | 4 | 21 | 23 | 7 | 12 | 6 | 9 | 18 | 8 | 15 | 17 | 3 | 20 |
| 15 | 0 | 15 | 19 | 21 | 8 | 10 | 1 | 12 | 17 | 2 | 6 | 14 | 20 | 4 | 7 | 16 | 24 | 3 | 9 | 18 | 22 | 5 | 23 | 13 | 11 |
| 16 | 0 | 16 | 18 | 5 | 17 | 6 | 15 | 20 | 3 | 19 | 1 | 7 | 22 | 8 | 12 | 24 | 11 | 21 | 2 | 9 | 23 | 10 | 13 | 4 | 14 |
| 17 | 0 | 17 | 18 | 22 | 7 | 13 | 8 | 1 | 12 | 19 | 4 | 10 | 15 | 14 | 6 | 3 | 21 | 20 | 2 | 9 | 16 | 23 | 24 | 11 | 5 |
| 18 | 0 | 18 | 0 | 19 | 18 | 9 | 19 | 2 | 9 | 0 | 2 | 18 | 19 | 19 | 9 | 9 | 2 | 2 | 0 | 0 | 18 | 18 | 9 | 2 | 19 |
| 19 | 0 | 19 | 0 | 2 | 19 | 18 | 2 | 9 | 18 | 0 | 9 | 19 | 2 | 2 | 18 | 18 | 9 | 9 | 0 | 0 | 19 | 19 | 18 | 9 | 2 |
| 20 | 0 | 20 | 2 | 24 | 1 | 14 | 12 | 17 | 15 | 9 | 7 | 4 | 3 | 6 | 8 | 22 | 23 | 16 | 18 | 19 | 21 | 11 | 5 | 10 | 13 |
| 21 | 0 | 21 | 2 | 13 | 20 | 8 | 3 | 16 | 22 | 9 | 17 | 1 | 24 | 12 | 15 | 5 | 10 | 23 | 18 | 19 | 11 | 4 | 14 | 7 | 6 |
| 22 | 0 | 22 | 19 | 11 | 15 | 7 | 20 | 3 | 16 | 2 | 12 | 8 | 21 | 1 | 17 | 23 | 13 | 24 | 9 | 18 | 5 | 14 | 10 | 6 | 4 |
| 23 | 0 | 23 | 18 | 14 | 16 | 12 | 22 | 21 | 24 | 19 | 20 | 17 | 5 | 15 | 3 | 13 | 4 | 11 | 2 | 9 | 10 | 7 | 6 | 1 | 8 |
| 24 | 0 | 24 | 9 | 10 | 3 | 1 | 16 | 22 | 21 | 18 | 15 | 12 | 23 | 17 | 20 | 11 | 14 | 5 | 19 | 2 | 13 | 6 | 4 | 8 | 7 |

За цими таблицями будуємо шифрограму:

| I | P | R | S | P | U | K | R | P | R | O | G | T | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| U | K | R | P | R | O | G | t | w | e | n | t | y | |
| 12 | 22 | 0 | 1 | 22 | 8 | 14 | 0 | 22 | 0 | 24 | 19 | 6 | $\bar{p}\bar{b}$ |
| + | + | + | + | + | + | + | + | + | + | + | + | + | . |
| 8 | 14 | 0 | 22 | 0 | 24 | 19 | 6 | 2 | 15 | 20 | 6 | 3 | $\bar{b}$ |
| = | = | = | = | = | = | = | | = | = | = | = | = | |
| 18 | 21 | 0 | 23 | 22 | 0 | 5 | 6 | 8 | 15 | 14 | 12 | 7 | $d$ |

Розшифрування відбувається у зворотному напрямку. Абоненту, якому адресована ця шифрограма, відомий ключ таблиці додавання і ключове слово "IPRSP". Виписуємо

– цифрову шифрограму і під нею цифрові значення ключового слова;

– за першим символом $a$ ключового слова знаходимо символ, який є першим символом шифрограми в таблиці додавання групи (у прикладі за символом $k_1 = 12$ знаходимо значення $n_1 = 18$, тобто розв'язуємо рівняння $k_1 + x = n_1$);

– за знайденим значенням у стовпчику, що відповідає цьому значенню $n_1$, знаходимо віповідник (у прикладі це число $x = 8$, якому відповідає символ $u$);

– такий цикл повторюється доки не буде знайдено весь текст повідомлення.

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 18 | 21 | 0 | 23 | 22 | 0 | 5 | 6 | 8 | 15 | 14 | 12 | 7 | $d$ |
| 12 | 22 | 0 | 1 | 22 | 8 | 14 | 0 | 22 | 0 | 24 | 19 | 6 | $\bar{p}\bar{b}$ |
| 8 | 14 | 0 | 22 | 0 | 24 | 19 | 6 | 2 | 15 | 20 | 6 | 3 | $\bar{b}$ |
| U | K | R | P | R | O | G | T | W | E | N | T | Y | $b$ |

. ♠

Зупинимося коротко на властивостях породжуючих елементів мультиплікативної групи кільця. Неважко переконатися, що породжуючими елементами мультиплікативної групи кільця $G_{25}$ є елементи 5, 6, 8,12,13,15,22,24. Це випливає з такої теореми.

**Теорема 58.** *Якщо мультиплікативна група дільників одиниці асоціативно-комутативного кільця з одиницею має твірний елемент $g$, то твірними елементами цієї групи будуть елементи $g^j$ такі, що НСД$(j, \varphi(k)) = 1$ [12].*

Дійсно, в нашому прикладі мультиплікативної групи кільця $G_{25}$ маємо

$$\begin{array}{lllll} \mathbf{6^1 = 6}, & 6^2 = 10, & \mathbf{6^3 = 5}, & 6^4 = 21, & 6^5 = 3, \\ 6^6 = 17, & \mathbf{6^7 = 8}, & 6^8 = 4, & \mathbf{6^9 = 13}, & 6^{10} = 23, \\ \mathbf{6^{11} = 22}, & 6^{12} = 20, & \mathbf{6^{13} = 12}, & 6^{14} = 7, & 6^{15} = 14, \\ 6^{16} = 11, & \mathbf{6^{17} = 24}, & 6^{18} = 16, & \mathbf{6^{19} = 15}, & 6^{20} = 1, \end{array}$$

а також

$$\begin{array}{lllll} \mathbf{5^1 = 5}, & 5^2 = 17, & \mathbf{5^3 = 13}, & 5^4 = 20, & 5^5 = 14, \\ 5^6 = 16, & \mathbf{5^7 = 6}, & 5^8 = 21, & \mathbf{5^9 = 8}, & 5^{10} = 23, \\ \mathbf{5^{11} = 12}, & 5^{12} = 11, & \mathbf{5^{13} = 15}, & 5^{14} = 10, & 5^{15} = 3, \\ 5^{16} = 4, & \mathbf{5^{17} = 22}, & 5^{18} = 7, & \mathbf{5^{19} = 24}, & 5^{20} = 1, \end{array}$$

З наведених таблиць і теореми 58 випливає, що елементи 1, 3, 4, 7, 10, 11, 14, 16, 17, 20, 21, 23 не можуть бути твірними мультиплікативної групи кільця (ці елементи є твірними підгруп цієї групи). Але на підставі циклічності мультиплікативної групи в ній можна застосувати функцію дискретного логарифма для шифрування повідомлень і передачі ключів. Наприклад, порівняння $6^x = 20 \pmod{25}$ має розв'язок $x = 12$, а $5^x = 24 \pmod{25}$ має розв'язок $x = 19$. Основною перешкодою використання дискретного логарифма в кільцях є те, що не завжди його мультиплікативна група буде циклічною. Наприклад, у кільці лишків за модулем $k$=16, маємо $\varphi(k) = 8$ і дільниками одиниці будуть елементи 1,3,5,7,9,11,13,15. Ці елементи мають порядок 2 або 4 і тому группа дільників одиниці не буде циклічною. Це одна з причин того, що у криптографії застосовують скінченні поля а не кільця, оскільки у скінченному полі його мультиплікативна група завжди циклічна. Ця обставина дозволяє використовувати у скінченних полях функцію дискретного логарифма. Враховуючи, що порядок мультиплікативної групи кільця дорівнює $\varphi(k)$, то найкращий порядок кільця $k$ буде тоді, коли $\varphi(k)$ просте число. У цьому випадку мультиплікативна група кільця проста і тому на підставі теореми Лагранжа буде циклічною. Але числами, які задовольняють цю умову є 3, 4 і 6 і інших таких чисел не існує. Дійсно, якщо $k = p_1^{r_1} p_2^{r_2} \cdots p_s^{r_s}$ – канонічний розклад числа $k$ на прості множники, то

$$\varphi(k) = p_1^{r_1-1}(p_1 - 1)p_2^{r_2-1}(p_2 - 1) \cdots p_s^{r_s-1}(p_s - 1)$$

і тому, коли $k > 6$, значення $\varphi(k)$ не є простим числом. Виникає питання: які умови повинен задовольняти порядок кільця, щоб його мультиплікативна група була циклічною групою великого порядку? На підставі теореми 55 вичерпну відповідь на це питання дає така теорема.

**Теорема 59 (Гаусса)**. *Мультиплікативна група кільця $\mathcal{Z}_k$ буде циклічною тоді і тільки тоді, коли k дорівнює 2, 4, $p^m$ або $2p^m$, $m \geq 1$, $p$ – непарне просте число* [*24*].

Із цієї теореми випливає, що мультиплікативна група кільця $G_{25}$, яке ізоморфне кільцю $\mathcal{Z}_{25}$, буде циклічною, оскільки $k = 25 = 5^2$ задовольняє умові теореми 59. Таким чином, щоб мультиплікативна група кільця була циклічною і мала великий порядок, потрібно вибрати порядок кільця $k = p^m$, де $m \geq 1$, $p$ – непарне просте число. Наприклад, якщо $k = 5^{19}$, то його мультиплікативна циклічна група буде мати порядок $\varphi(k) = 4 \cdot 5^{18} = 4 \cdot 1953125 \cdot 1953125 > 2^{41}$. Для простої криптосистеми з одноразовим ключем такого порядку достатньо для обміну повідомленнями. Отже, в кільцях теж можна використовувати функцію дискретного логарифма, якщо вибирати порядок кільця на підставі теореми 59. Для порівняння, якщо поле має порядок $k = 5^{19}$, то його мультиплікативна група матиме порядок $k = 5^{19} - 1$, але побудова такого поля потребує значно більших затрат часу і пам'яті, ніж побудова кільця. Зокрема, у розглянутому прикладі 2.9.4 мультиплікативна група кільця $G_{25}$ матиме на підставі теореми 57 вісім породжуючих елементів, оскільки $\varphi(20) = 8$ (два елементи 5 і 6 були наведені вище).

На підставі теореми 55 таблиці кільця можна не будувати, а скористатися ізоморфізмом $f : \mathcal{Z}_{25} \to G_{25}$. Дійсно, ізоморфне відображення має вигляд

| | | | | |
|---|---|---|---|---|
| $f(0)=0$, | $f(5)=2$, | $f(10)=9$, | $f(15)=19$, | $f(20)=18$, |
| $f(1)=1$, | $f(6)=4$, | $f(11)=11$, | $f(16)=21$, | $f(21)=20$, |
| $f(2)=6$, | $f(7)=3$, | $f(12)=13$, | $f(17)=12$, | $f(22)=24$, |
| $f(3)=8$, | $f(8)=5$, | $f(13)=15$, | $f(18)=14$, | $f(23)=22$, |
| $f(4)=10$, | $f(9)=7$, | $f(14)=17$, | $f(19)=16$, | $f(24)=23$. |

Дільниками нуля в кільці лишків $\mathcal{Z}_{25}$ є елементи 0,5,10,15,20. Цим елементам при ізоморфізмі в кільці $G_{25}$ відповідають елементи

$I_0 = \{0, 2, 9, 18, 19\}$, які утворюють ідеал. А це дає можливість розглядати фактор-кільце за цим ідеалом. Два елементи $x$ і $y$ кільця порівнюються за ідеалом $I$, якщо $x - y \in I$. Тоді фактор-кільце за ідеалом $I_0$ має такі елементи (класи):

$$I_0 = \{0, 2, 9, 18, 19\}, \quad I_1 = \{1, 4, 11, 20, 21\}, \quad I_2 = \{3, 6, 12, 13, 24\},$$
$$I_3 = \{5, 8, 14, 15, 22\}, \quad I_4 = \{7, 10, 16, 17, 23\}.$$

Таблиці додавання і множення (табл. 2.9.10 і 2.9.11, де елементи зображуються індексами класів) цього фактор-кільця набувають вигляду

Таблиця 2.9.10

| + | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 |
| 1 | 1 | 2 | 3 | 4 | 0 |
| 2 | 2 | 3 | 4 | 0 | 1 |
| 3 | 3 | 4 | 0 | 1 | 2 |
| 4 | 4 | 0 | 1 | 2 | 3 |

,

Таблиця 2.9.11

| · | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 |
| 2 | 0 | 2 | 4 | 1 | 3 |
| 3 | 0 | 3 | 1 | 4 | 2 |
| 4 | 0 | 4 | 3 | 2 | 1 |

.

Оскільки фактор-кільце має простий порядок 5, то воно є полем.

Решта елементів кільця $G_{25}$ – дільники одиниці, які утворюють мультиплікативну абелеву групу цього кільця. Отже, для знаходження добутку, наприклад, елементів 6 і 7, знаходимо добуток їхніх відповідників 2 і 9 у кільці лишків $18 = 18 \pmod{25}$. Тоді в кільці $G_{25}$ відповідником є елемент $f(18)$=14. Отже, $6 \cdot 7 = 14$ в кільці $G_{25}$.

Для повноти картини розглянемо випадок, коли мультиплікативна група кільця не буде циклічною. Це випливає з теореми 60.

**Теорема 60.** *Якщо порядок мультиплікативної групи кільця $k = pq$, де $p \neq q$ і $p, q$ – непарні прості числа, то ця группа не буде циклічною.*

*Доведення.* Покажемо, що порядок довільного елемента мультиплікативної групи $MGN_k$ кільця є дільником НСК$(p-1, q-1)$, де НСК означає найменше спільне кратне. Нехай $a \in MGN_k$. На підставі малої теореми Ферма (теорема 32) маємо: $a^{p-1} \equiv 1 \pmod{p}$ і $a^{q-1} \equiv 1 \pmod{q}$, а звідси випливає, що $a^r \equiv 1 \pmod{p}$ і $a^r \equiv 1 \pmod{q}$, де $r$=НСК$(p-1q-1)$. Отже, число $a^r - 1$ кратне числам $p$ і $q$. На підставі простоти чисел $p$ і $q$ дістаємо $a^r - 1$ кратне $k = p \cdot q$. А це

означає, що $r$ кратне порядку числа a. Звідси на підставі парності чисел $p-1$ і $q-1$ дістаємо, що $\text{НСК}(p-1, q-1) < \varphi(k) = (p-1)(q-1)$. Отже, у групі $MGN_k$ не існує елемента, порядок якого $\varphi(k)$, а це означає відсутність у ній породжуючого елемента. ■

**Шифри гомофонічні**. Одноалфавітні шифри, як було показано вище, ламаються методом частотного аналізу. Запобігти такому криптоаналізу можна відображенням однієї літери в декілька її образів, які називають *гомофонами*. Кількість гомофонів для кожної літери повинна бути пропорціональна частоті появи цієї літери у явному тексті. Якщо гомофони використати ротаційно, то можна сподіватися, що частота появи літер не буде ідентична і це призведе до неможливості використання частотного криптоаналізу. Ідею застосування гомофонів приписують Гауссу. Він вважав, що гомофонічний шифр не можна зламати, але це не так. Гомофонічний шифр ілюструє приклад 2.9.5.

У гомофонічних шифрах частотний аналіз стає неможливим, але частота появи комбінацій сусідніх літер дає можливість застосувати цей тип аналізу. В англійській мові маємо 676 різних двознаків, з них 18 становить більше 25 % тексту. Таким чином, гомофони теж проявляють частоту появи літер і атака методом частотного аналізу стає можливою. Подібна ситуація із частотою появи в текстах тризнаків і чотиризнаків.

**Приклад 2.9.5.** Нехай символи алфавіту англійської мови упорядковані як у прикладі 2.9.4, а кільце має порядок $k = 49 = 7^2$. Значення рядка $T_+[1, 49]$ задані такою підстановкою:

$$f = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \\ 25 & 26 & 27 & 28 & 29 & 30 & 31 & 32 & 33 & 34 & 35 & 36 & 37 & 38 & 39 & 40 & 41 & 42 & 43 & 44 & 45 & 46 & 47 & 48 & \\ \hline 1 & 5 & 34 & 45 & 37 & 7 & 41 & 10 & 38 & 35 & 17 & 20 & 40 & 43 & 44 & 36 & 47 & 2 & 26 & 46 & 39 & 31 & 32 & 30 & 27 \\ 28 & 0 & 42 & 18 & 22 & 25 & 24 & 23 & 48 & 11 & 12 & 8 & 6 & 9 & 33 & 14 & 13 & 29 & 15 & 19 & 4 & 16 & 21 & 3 & \end{pmatrix}.$$

Задамо сюр'єкцію відображенням (2.58) з початковим значенням $m_1 = 7$ і модулем $m = 25$, тоді отримуємо такі класи еквівалентності за правилом: порядковий номер класу елемента $m_i$ дорівнює $i$ за модулем 25.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 10 | 17 | 2 | 34 | 11 | 20 | 39 | 33 | 48 | 3 | 45 | 4 | 37 | 6 | 41 | 13 | 43 | 15 | 36 | 8 | 38 | 9 | 35 | 12 |
| 40 | 14 | 44 | 19 | 46 | 16 | 41 | 21 | 31 | 24 | 27 | 42 | 29 | 22 | 32 | 23 | 30 | 25 | 28 | 18 | 26 | 0 | 1 | 5 | |
| a | b | c | d | e | f | g | h | i/j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |

Тепер символу алфавіту можна ставити у відповідність різні значення з класу еквівалентності, які відповідають цьому символу, і виконувати шифрування за допомогою наведених вище алгоритмів. Дійсно, використовуючи ізоморфізми $\varphi : \mathcal{Z}_{49} \to G_{49}$ і $\varphi^{-1} : G_{49} \to \mathcal{Z}_{49}$, дістаємо такі таблиці відповідностей:

$\varphi : \mathcal{Z}_{49} \to G_{49}$

| | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | |
| 49 | 1 | 5 | 7 | 10 | 17 | 2 | 34 | 11 | 20 | 39 | 33 | 48 | 3 | 45 | 4 | 37 | 6 | 41 | 13 | 43 | 15 | 36 | 8 | 38 |
| 9 | 35 | 12 | 40 | 14 | 44 | 19 | 46 | 16 | 47 | 21 | 31 | 24 | 27 | 42 | 29 | 22 | 32 | 23 | 30 | 25 | 28 | 18 | 26 | 0 |

$\varphi^{-1} : G_{49} \to \mathcal{Z}_{49}$

| | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | |
| 49 | 1 | 6 | 13 | 15 | 2 | 17 | 3 | 23 | 25 | 4 | 8 | 27 | 19 | 29 | 21 | 33 | 5 | 47 | 31 | 9 | 35 | 41 | 43 | 37 |
| 45 | 48 | 38 | 46 | 40 | 44 | 36 | 42 | 11 | 7 | 26 | 22 | 16 | 24 | 10 | 28 | 18 | 39 | 20 | 30 | 14 | 32 | 34 | 12 | |

Наприклад, для ключового слова **KUKU** і тексту **KUKURIKU**, який потрібно зашифрувати, дістаємо такий зашифрований текст:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| K | U | K | U | K | U | K | U | |
| K | U | K | U | R | I | K | U | |
| 48 | 36 | 24 | 18 | 24 | 36 | 48 | 36 | $\bar{p}b$ |
| + | + | + | + | + | + | + | + | , |
| 24 | 18 | 24 | 18 | 13 | 31 | 48 | 36 | $\bar{b}$ |
| = | = | = | = | = | = | = | = | |
| 0 | 43 | 9 | 25 | 34 | 20 | 38 | 30 | $d$ |

де символи, які повторюються, приховані. ♠

**Надійність алгоритму шифрування.** Основним чинником, який має вплив на надійність розглянутого алгоритму, є кількість сюр'єкцій $g : G \to M$, $|M| = m$.

У випадку $k = m$ таких сюр'єкцій буде $m!$ і тоді приведений алгоритм – це деяка модифікація шифру Віженера. Для цього шифру відомі ефективні способи його криптоаналізу [1, 13].

У випадку $k > m$ кожна сюр'єкція $g : G_k \to M$ породжує $m = |X|$ класів еквівалентності з $t = \lfloor k/m \rfloor$ елементами в класі і $m!$ способами встановлення відповідності між цими класами і символами алфавіту. Тоді кількість способів зашифрувати повідомлення буде $S = D(k, m)(t(t + 1)/2)$, де $D(k, m)$ – число сюр'єкцій $k$-елементної множини на $m$-елементну множину. Відомо, що значення $D(k, m)$ обчислюється за формулою $D(k, m) = \sum_{j=0}^{m} (-1)^j C_m^j (m - j)^k$[37]. Навіть при невеликих значеннях $k$ і $m$ значення $S$ росте досить швидко, наприклад, для $k = 9, m = 4$ дістаємо $S = 186480 \cdot 3 > 2^{19}$. Якщо тепер до цього числа додати множником кількість способів вибору ключового слова, а ця кількість дорівнює $m^r$, де $m = |X|$, $r$ – довжина ключового слова $p$, то

у результаті отримуємо таке число способів зашифрувати повідомлення – $m^r \cdot S = m^r D(k,m) \cdot (t(t+1)/2)$, де $r$ – довжина ключового слова $p, m = |X|, k = |G_k|$, а $t(t+1)/2$ – кількість різних пар, які можна побудувати із $t$ елементів класу еквівалентності. Вибираючи порядок кільця досить великим, дістаємо можливість повністю приховати частоту входження символів у повідомленні. У такому способі шифрування методи частотного аналізу і гамування [1, 13] значно ускладнюються.

З наведених алгоритмів **RG-EN** і **RG-DE** випливає, що заміна початкового значення $m_1$ сюр'єкції $g$ (наприклад, у відображенні (2.58)), ключового слова $p$ і рядка $T_+[1,k]$, за якими будується кільце, досить проста. Крім того, порядок $k$ кільця завжди можна підібрати таким, щоб він був кратним $m = |X|$ (що не завжди можна зробити для поля). Залишається описати спосіб передачі трійки $(T_+[1,k], g, p)$. Рядок $T_+[1,k]$ задається вектором $(1, a_1, a_2, \ldots, a_{k-1})$, де $a_i = a_{i-1} + 1$, $i = 1, \ldots, k-1$. Сюр'єкція задається парою $(m_1, m_{i-1} + 1)$ (модуль $k$ відомий із рядка $T_+[1,k]$). Ключове слово $p$ передається вектором $(\bar{p}) = (k_1, k_2, \ldots, k_r)$ .

Хоча приведена оцінка способів побудови шифрованого тексту має високий порядок зростання і може використовуватися в системі з одноразовим ключем, але така система ще повинна мати спосіб автентифікації абонентів.

**Використання моделі математичного сейфа** для автентифікації абонентів [8, 17].

Неформально під математичним сейфом розуміють таку систему $Z = \{z_1, z_2, \ldots, z_n\}$ взаємнопов'язаних між собою засувів таких, що коли виконується поворот ключа в одному із засувів, то такий же поворот виконується і у всіх засувах, які пов'язані з даним.

Математичний сейф може задаватися або за допомогою прямокутної матриці або за допомогою графа (орієнтованого або неорієнтованого). Якщо сейф задається матрицею, то її елементи відповідають засувам, а значення цих елементів – позиціям засувів, тобто у вигляді матриці $Z = ||z_{ij}||, i = 1, \ldots, m, j = 1, \ldots, n$. Цей спосіб називають *матричним сейфом*. А коли сейф задається графом, то його вершини відповідають засувам. Цей спосіб називають *графо-*

*вим сейфом*. У матричному сейфі кожний засув $z_{ij}$ пов'язаний із тими засувами, які розміщені в $i$-му рядку і в $j$-му стовпці, а в графовому сейфі із засувом, який міститься у вершині $v_i$, пов'язаними вважаються засуви, які знаходяться в суміжних вершинах із вершиною $v_i$.

Кожний із засувів може перебувати в одній із позицій, і всіх таких позицій скінченне число: $0, 1, \ldots, k-1$. Засув відкритий, якщо він знаходиться в позиції 0. У довільній іншій позиції засув вважається закритим. Початковий стан сейфа визначається матрицею $A = ||a_{ij}||$. Якщо в якомусь засуві виконується поворот ключа, то всі засуви, які пов'язані з даним засувом, збільшують свої позиції на одиницю за модулем $k$.

*Необхідно розв'язати таку задачу*. Враховуючи початкові стани засувів сейфа, знайти таку послідовність засувів і число поворотів ключа в них, щоб сейф перейшов у положення відкритого, тобто коли всі засуви знаходяться в позиції 0.

Розглянемо матричний сейф. Нехай матриця $X = ||x_{ij}||$ – шуканий розв'язок задачі, де $x_{ij}$ дорівнює числу поворотів ключа в засуві $z_{ij}$. Позначимо $x = (x_{11}, \ldots, x_{1n}, x_{21}, \ldots, x_{2n}, \ldots, x_{m1}, \ldots, x_{mn})$ вектор-стовпець, отриманий із матриці $X$ послідовним записом її рядків. Аналогічно з матриці $A$ початкових станів засувів отримуємо вектор-стовпець $a$. Тоді розв'язок задачі про математичний сейф зводиться до системи лінійних неоднорідних рівнянь (СЛНР) $Bx + a \equiv 0 \pmod{k}$ у кільці $G_k$, де матриця $B$ розмірності $mn \times mn$.

*Модифікуємо задачу*: будемо шукати розв'язок задачі про МС у вигляді розв'язання СЛНР вигляду $Bx + a \equiv c \pmod{k}$, у скінченному кільці типу $G_k$. Це означає, що сейф відкривається, коли позиції засувів дорівнюють значенням вектора $c$. Отже, розв'язання задачі про математичний сейф зводиться до пошуку розв'язків системи лінійних рівнянь, у скінченному кільці $G_k$, яке побудоване описаним методом.

**Приклад 2.9.6.** Нехай математичний сейф заданий матрицею $A$ в кільці $G_{25}$, за якою будується система лінійних рівнянь $Bx + a \equiv c \pmod{25}$, де

$$A = \begin{pmatrix} 9 & 7 & 12 & 4 & 5 & 21 \\ 5 & 4 & 2 & 17 & 20 & 20 \end{pmatrix}$$

і $c = (16, 12, 4, 10, 21, 20, 7, 5, 2, 20, 2, 7)$. Тоді вектор

$$a = (9, 7, 12, 4, 5, 21, 5, 4, 2, 17, 20, 20),$$

вектор

$$a - c = (21, 12, 11, 6, 12, 18, 23, 22, 0, 14, 21, 13),$$

а матриця $B$, за якою будується система лінійних рівнянь $Bx + a - c \equiv 0 \pmod{25}$, набуває вигляду

$$B = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 21 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 12 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 11 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 6 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 12 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 18 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 23 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 22 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 14 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 21 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 13 \end{pmatrix},$$

в якій вектор $a - c$ представлено останнім її стовпцем.

Використовуючи ізоморфізм кілець, розв'язуємо (наприклад, методом TSS) відповідну систему в кільці $\mathcal{Z}_{25}$, а потім знову за оберненим ізоморфізмом знаходимо розв'язок системи, над кільцем $G_{25}$ [33] $d = (21, 12, 19, 24, 23, 8, 13, 11, 12, 1, 11, 5)$, який висилається абоненту, що прислав рядок $T_+[1, 25]$ і вектор $c$. Якщо надісланий вектор $d$ відкриває сейф (що означає легальність отримувача), то далі відправник виконує крок 4 алгоритму **GM-EN**$(T_+[1, k], g, \mathrm{p})$ і висилає абоненту дозвіл на подальшу роботу. Отриманий дозвіл для отримувача означає легальність відправника і він отримує доступ до шифрограми. ♠

З властивостей кільця $G_k$ випливає, що задача про математичний сейф буде сумісною, коли серед розв'язків системи $Bx + a \equiv \equiv 0 \pmod{k}$ є хоча б один розв'язок, остання координата якого не є дільником нуля, і не буде мати розв'язку, коли елементи матриці $A$ є дільниками нуля. Останнє випливає з того, що дільники нуля утворюють ідеал кільця, а це достатня умова того, що всі розв'язки будуть мати останню координату дільником нуля, яка не матиме оберненого. Формування матриці сейфа з потрібними властивостями робить одна зі сторін обміну повідомленнями.

Використання моделі МС дозволяє організувати процесс автентифікації абонентів (причому обох), які беруть участь в обміні повідомленнями. Дійсно, автентифікацію можна організувати таким чином: послати абоненту пару $(T_+[1,k],\mathrm{c})$; отримати від нього розв'язок, який повинен відкривати сейф. При цьому кільце, в якому розв'язується задача про МС не обов'язково має бути кільцем $G_k$, воно може бути довільним скінченним кільцем або полем. Якщо цей розв'язок відкриває сейф, то абонент легальний, інакше абонент нелегальний. Отримання підтвердження легальності абонента за отриманим розв'язком дозволяє дістати доступ абонентам до параметрів алгоритма **RG-EN** для виконання шифрування.

Враховуючи сказане, наведемо алгоритми шифрування та розшифрування повідомлень у такій криптосистемі.

**RGM-EN**$(T_+[1,k_1],c)$
1) Вислати закритим каналом вектор $c$ і визначальний рядок $T_+[1,k_1]$, кільця $G_{k_1}$.
2) Вислати відкритим каналом вектор $a$.
3) Чекати на отримання розв'язку $d$ задачі про МС у кільці $G_{k_1}$.
4) Якщо $d$ відкриває сейф, то виконати крок 5), інакше СТОП("абонент нелегал").
5) Виконати алгоритм **GM-EN**$(T_+[1,k],g,p)$ і вислати абоненту дозвіл на роботу.

**RGM-DE**$(T_+[1,k_1],c)$
1) Побудувати кільце $G_{k_1}$ за отриманим рядком $T_+[1,k_1]$.
2) За векторами $a$ і $c$ знайти розв'язок $d$ системи $Bx+a\equiv c \pmod{k_1}$.
3) Вислати розв'язок $d$ абоненту, який прислав параметри $T_+[1,k_1],c$.
4) Якщо отримано дозвіл на роботу, то виконати алгоритм **GM-DE**$(T_+[1,k],g,p)$.

У цьому варіанті слабким місцем автентифікації є те, що матриця $B'$ системи має стандартний вигляд, а це спрощує роботу криптоаналітика. Щоб уникнути цього недоліку, краще задавати математичний сейф на графах, оскільки в такому випадку структура матриці залежить від топології графа і вже не буде мати регулярного вигляду. Топологія графа передається вектором списків суміжності вершин графа, довжина якого не перевищує $2|E|$, де $|E|$ – кількість ребер графа.

**Приклад 2.9.7**. Розглянемо математичний сейф, який задано неорієнтованим графом, зображеним на рис. 2.9.2. Список суміжних вершин цього графа має вигляд

[**1**:2,4,11; **2**:8; **3**:5,9; **4**:7,10; **5**:12; **6**:7,9; **7**:8; **8**:2:7; **9**:12; **10**:11],

а початкові позиції засувів у вершинах графа такі, як у матриці $A$ з прикладу 2.9.6:

$$A = \begin{pmatrix} 21 & 12 & 11 & 6 & 12 & 18 \\ 23 & 22 & 0 & 14 & 21 & 13 \end{pmatrix}.$$

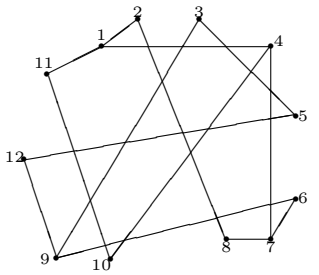



Рис. 2.9.2. Граф, який задає сейф

Матриця системи $Bx + a \equiv 0 \pmod{25}$ для заданого таким графом сейфа набуває вигляду (останній стовпець матриці представляє вектор $a$):

$$B = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 21 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 12 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 11 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 6 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 12 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 18 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 23 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 22 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 14 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 21 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 13 \end{pmatrix}.$$

Розв'язком цієї системи в кільці $G_{25}$ є вектор (16,6,12,13,23,23,22,3,5,17,11,1,21), який після нормалізації (множення на коефіцієнт 11, оскільки $11 \cdot 21 = 1$) дає розв'язок $s = (7, 24, 13, 3, 17, 17, 8, 6, 15, 10, 20, 11, 1)$. Цей розв'язок відкриває сейф і подальша робота йде за алгоритмами **RGM-EN** і **RGM-DE**. ♠

Таким чином, у цьому варіанті криптоаналітику серед іншого невідома також структура матриці $B$, що значно ускладнює його роботу.

Можливі також різні варіації і з матрицею сейфа, одна з яких наведена у прикладі 2.9.8 [17].

**Приклад 2.9.8.** Нехай задача про МС розв'язується в кільці за модулем 27 із матрицею початкових позицій засувів

$$B = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

і з такою комбінацією засувів, яка відкриває сейф,

$$c = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}.$$

Модифікована матриця $A$ системи має вигляд

$$\left|\begin{array}{ccccccccc} 1 & 1 & 1 & 3 & 0 & 0 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 3 & 0 & 0 & 2 & 0 \\ 1 & 1 & 1 & 0 & 0 & 3 & 0 & 0 & 2 \\ 1 & 0 & 0 & 3 & 3 & 3 & 2 & 0 & 0 \\ 0 & 1 & 0 & 3 & 3 & 3 & 0 & 2 & 0 \\ 0 & 0 & 1 & 3 & 3 & 3 & 0 & 0 & 2 \\ 1 & 0 & 0 & 3 & 0 & 0 & 2 & 2 & 2 \\ 0 & 1 & 0 & 0 & 3 & 0 & 2 & 2 & 2 \\ 0 & 0 & 1 & 0 & 0 & 3 & 2 & 2 & 2 \end{array}\right|.$$

Це означає, що поворот ключа у засувах першого рядка змінює стани засувів на одну позицію, поворот у засувах другого рядка змінює стани засувів на три позиції, а третього рядка – стани засувів на дві позиції. Тоді СЛНДР $Ax+a \equiv c \pmod{27}$ набуває вигляду

$$Ax + a = c \left\{\begin{array}{ccccccccccc} 1 & 1 & 1 & 3 & 0 & 0 & 2 & 0 & 0 & 1 & = & 1 \\ 1 & 1 & 1 & 0 & 3 & 0 & 0 & 2 & 0 & 2 & = & 2 \\ 1 & 1 & 1 & 0 & 0 & 3 & 0 & 0 & 2 & 0 & = & 3 \\ 1 & 0 & 0 & 3 & 3 & 3 & 2 & 0 & 0 & 0 & = & 4 \\ 0 & 1 & 0 & 3 & 3 & 3 & 0 & 2 & 0 & 0 & = & 5 \\ 0 & 0 & 1 & 3 & 3 & 3 & 0 & 0 & 2 & 0 & = & 6 \\ 1 & 0 & 0 & 3 & 0 & 0 & 2 & 2 & 2 & 0 & = & 7 \\ 0 & 1 & 0 & 0 & 3 & 0 & 2 & 2 & 2 & 0 & = & 8 \\ 0 & 0 & 1 & 0 & 0 & 3 & 2 & 2 & 2 & 0 & = & 9 \end{array}\right. \pmod{27}.$$

Розв'язком цієї системи є вектор $x = (4, 5, 18, 20, 20, 7, 24, 24, 24, 18)$, який не відкриває сейф. Для того, щоб цей розв'язок відкривав сейф, необхідно його перетворити таким чином:

– перші три координати, які відповідають коефіцієнтам 1, залишаються незмінними;

– другі три координати, які відповідають коефіцієнтам 3, множаться на 3 за модулем 27;

– треті три координати, які відповідають коефіцієнтам 2, множаться на 2 за модулем 27.

У результаті дістаємо розв'язок $a = (4, 5, 18, 6, 6, 21, 21, 21, 9)$, який відкриває сейф. Дійсно,

$$\begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \to (x_{11} = 4) \begin{pmatrix} 5 & 6 & 4 \\ 4 & 0 & 0 \\ 4 & 0 & 0 \end{pmatrix} \to (x_{12} = 5) \begin{pmatrix} 10 & 11 & 9 \\ 4 & 5 & 0 \\ 4 & 5 & 0 \end{pmatrix} \to (x_{13} = 18) \to$$

$$\to \begin{pmatrix} 1 & 2 & 0 \\ 4 & 5 & 18 \\ 4 & 5 & 18 \end{pmatrix} \to (x_{21} = 6) \begin{pmatrix} 7 & 2 & 0 \\ 10 & 11 & 24 \\ 10 & 5 & 18 \end{pmatrix} \to (x_{22} = 6) \begin{pmatrix} 7 & 8 & 0 \\ 16 & 17 & 3 \\ 10 & 11 & 18 \end{pmatrix} \to$$

$$\to x_{23} = 21) \begin{pmatrix} 7 & 8 & 21 \\ 10 & 11 & 24 \\ 10 & 11 & 12 \end{pmatrix} \to (x_{31} = 21) \begin{pmatrix} 1 & 8 & 21 \\ 4 & 11 & 24 \\ 4 & 5 & 6 \end{pmatrix} \to$$

$$\to (x_{32} = 21) \begin{pmatrix} 1 & 2 & 21 \\ 4 & 5 & 24 \\ 25 & 26 & 0 \end{pmatrix} \to (x_{33} = 9) \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} .\spadesuit$$

## 2.9.6. Побудова скінченних кілець

Стандартним способом побудови алгебр вищих порядків з алгебр нижчих порядків є застосування операції прямого добутку алгебр. Покажемо, як ця операція діє на скінченних кільцях.

Нехай $\mathcal{K}_1 = (K_1, \{+_1, \cdot_1\})$ і $\mathcal{K}_2 = (K_2, \{+_2, \cdot_2\})$ скінченні кільця, порядок яких $k_1$ і $k_2$ відповідно. Прямим добутком $\mathcal{K}_1 \times \mathcal{K}_2$ кілець $\mathcal{K}_1$ і $\mathcal{K}_2$ називається алгебра $\mathcal{K} = (K_1 \times K_2, \{\oplus, \odot\})$, операції якої діють покомпонентно, тобто

$$(a, b) \oplus (c, d) = (a +_1 c, b +_2 d), \quad (a, b) \odot (c, d) = (a \cdot_1 c, b \cdot_2 d).$$

Нулем такого кільця є елемент $(0_1, 0_2)$, де $0_1$ – нуль кільця $\mathcal{K}_1$, а $0_2$ – нуль кільця $\mathcal{K}_2$. Якщо кільця $\mathcal{K}_1$ і $\mathcal{K}_2$ мають одиниці, то елемент $(1_1, 1_2)$ – одиниця кільця $\mathcal{K}$.

**Приклад 2.9.9.** Побудувати прямий добуток кілець $\mathcal{Z}_2$ і $\mathcal{Z}_3$.

*Розв'язання.* Кільце $\mathcal{K} = \mathcal{Z}_2 \times \mathcal{Z}_3$ матиме елементи $\{(0,0), (0,1), (0,2), (1,0), (1,1), 1, 2)\}$ з такими таблицями додавання і множення:

| $\oplus$ | (0,0) | (0,1) | (0,2) | (1,0) | (1,1) | (1,2) |
|---|---|---|---|---|---|---|
| (0,0) | (0,0) | (0,1) | (0,2) | (1,0) | (1,1) | (1,2) |
| (0,1) | (0,1) | (0,2) | (0,0) | (1,1) | (1,2) | (1,0) |
| (0,2) | (0,2) | (0,0) | (0,1) | (1,2) | (1,0) | (1,1) |
| (1,0) | (1,0) | (1,1) | (1,2) | (0,0) | (0,1) | (0,2) |
| (1,1) | (1,1) | (1,2) | (1,0) | (0,1) | (0,2) | (0,0) |
| (1,2) | (1,2) | (1,0) | (1,1) | (0,2) | (0,0) | (0,1) |

,

| $\odot$ | (0,0) | (0,1) | (0,2) | (1,0) | (1,1) | (1,2) |
|---|---|---|---|---|---|---|
| (0,0) | (0,0) | (0,0) | (0,0) | (0,0) | (0,0) | (0,0) |
| (0,1) | (0,0) | (0,1) | (0,2) | (0,0) | (0,1) | (0,2) |
| (0,2) | (0,0) | (0,2) | (0,1) | (0,0) | (0,2) | (0,1) |
| (1,0) | (0,0) | (0,0) | (0,0) | (1,0) | (1,0) | (1,0) |
| (1,1) | (0,0) | (0,1) | (0,2) | (1,0) | (1,1) | (1,2) |
| (1,2) | (0,0) | (0,2) | (0,1) | (1,0) | (1,2) | (1,1) |

.

Визначимо відображення $f : K \to \{0, 1, 2, 3, 4, 5\}$:

$$f((0,0)) = 0, \quad f((1,1)) = 1, \quad f((0,2)) = 2, \quad f((1,0)) = 3, \quad f((0,1)) = 4, \quad f((1,2)) = 5.$$

Тоді таблиці операцій кільця $\mathcal{K}$ перетворюються в таблиці операцій кільця лишків $\mathcal{Z}_6$:

| $\oplus$ | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 1 | 2 | 3 | 4 | 5 | 0 |
| 2 | 2 | 3 | 4 | 5 | 0 | 1 |
| 3 | 3 | 4 | 5 | 0 | 1 | 2 |
| 4 | 4 | 5 | 0 | 1 | 2 | 3 |
| 5 | 5 | 0 | 1 | 2 | 3 | 4 |

,

| $\odot$ | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 | 5 |
| 2 | 0 | 2 | 4 | 0 | 2 | 4 |
| 3 | 0 | 3 | 0 | 3 | 0 | 3 |
| 4 | 0 | 4 | 2 | 0 | 4 | 2 |
| 5 | 0 | 5 | 4 | 3 | 2 | 1 |

.

Отже, відображення $f$ є ізоморфізмом кілець $\mathcal{K}$ і $\mathcal{Z}_6$. Варто зауважити, що кільця $\mathcal{Z}_2$ і $\mathcal{Z}_3$ є полями, а кільце $\mathcal{K}$ полем не буде ($\mathcal{K}$ має дільники нуля (0,2), (1,0), (0,1)). ♠

Отриману конструкцію можна розширити на довільні кільця $\mathcal{Z}_m$ і $\mathcal{Z}_n$ за умови, що НСД$(m, n)$=1. Справелива така теорема.

**Теорема 61.** *Нехай* $m = m_1 \cdot m_2 \cdots m_r$, *де* НСД$(m_i, m_j) = 1$, $i \neq j$. *Тоді кільце* $\mathcal{Z}_{m_1} \times \mathcal{Z}_{m_2} \times \cdots \times \mathcal{Z}_{m_r}$ *ізоморфне кільцю* $\mathcal{Z}_m$ [*14*].

Доведення теореми випливає з китайської теореми про остачі.

Із цієї теореми очевидним чином отримуємо такий наслідок.

**Наслідок 9.** *Нехай* $m = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$ *канонічний розклад* $m$ *на прості множники. Тоді кільце* $\mathcal{Z}_m$ *ізоморфне кільцю* $\mathcal{Z}_{p_1^{k_1}} \times \mathcal{Z}_{p_2^{k_2}} \times \times \cdots \times \mathcal{Z}_{p_r^{k_r}}$.

Оскільки кільця в такому добутку примарні, то звідси випливає, що можна використовувати у прямому добутку довільне з кілець для шифрування інформації. Крім того, з примарності кілець $\mathcal{Z}_{p_i^{k_i}}$ випливає, що вони мають єдиний максимальний ідеал, фактор-кільце за яким буде полем.

Додамо до сказаного такі властивості кілець.

**Теорема 62.** *а) Кільце $\mathcal{Z}_n$ буде полем тоді і тільки тоді, коли $n$ просте число.*

*б) У кільці $\mathcal{Z}_n$ елемент $a$ буде дільником одиниці тоді і тільки тоді, коли НСД$(a, n)$=1.*

*в) Якщо $f : R \to Q$ гомоморфізм кілець, то*

*– $f(0_R) = 0_Q$, де $0_R, 0_Q$ – нулі кілець $R$ і $Q$ відповідно;*

*– $f(-a) = -f(a)$; $f(na) = nf(a)$ для всіх $a \in R, n \in \mathcal{Z}$;*

*– $f(a^n) = (f(a))^n$ для всіх $a \in R, n \geq 0, n \in \mathcal{Z}$.*

*– Якщо $R'$ підкільце кільця $R$, то $f(R')$ підкільце кільця $Q$.*

*Якщо $f : R \to Q$ сюр'єктивний гомоморфізм, то*

*а) коли $1_R$ одиниця $R$, то $f(1_R)$ одиниця $Q$;*

*б) коли $R$ кільце з одиницею і $a$ дільник одиниці у $R$, то $f(a)$ дільник одиниці у $Q$ і $f(a^{-1}) = (f(a))^{-1}$;*

*в) коли $R$ комутативне кільце, то і $Q$ комутативне кільце;*

*г) коли $J$ ідеал кільця $R$, то $f(J)$ ідеал кільця $Q$.*

### 2.9.7. Поля

Під час розгляду кілець було видно, що дільники нуля обмежують використання всіх його елементів. Цього не може статися, коли кільце є полем. Поле – це окремий випадок асоціативно-комутативного кільця з одиницею.

**Означення 67.** *Асоціативно-комутативне кільце з одиницею називається* **полем**, *якщо воно як відносно додавання, так і відносно множення є абелевою групою. Група поля відносно додавання називається* **адитивною**, *а група відносно множення –* **мультиплікативною**.

Отже, поле є областю цілісності і, як випливає з теореми 56, кільце поліномів над довільним полем $\mathcal{F}$ є областю цілісності. Одиницею в цій області цілісності служить поліном, коефіцієнти якого всі рівні нулю, крім коефіцієнта $a_0$, який дорівнює $1_{\mathcal{F}}$ – одиниці поля $\mathcal{F}$.

Поле $\mathcal{F}$ називається скінченним, якщо його носій $A$ має скінченне число елементів.

Нехай $a \in \mathcal{F}$, тоді елементи $a, a+a, a+a+a, \ldots$ є елементами поля, які будемо позначати $a, 2a, 3a, \ldots, na, \ldots$ відповідно (множник $n$ не обов'язково повинен бути елементом поля $\mathcal{F}$). Аналогічно елементи $a, a \cdot a, a \cdot a \cdot a, \ldots$ теж є елементами поля і їх позначають $a, a^2, a^3, \ldots, a^n, \ldots$ відповідно. Нехай $a \neq 0$ – елемент поля $\mathcal{F}$.

**Означення 68.** *Якщо існує таке найменше число $n \in \mathcal{N}$, що $n \cdot a = 0$, то $n$ називається адитивним порядком елемента $a$.*

*Якщо існує таке найменше число $n \in \mathcal{N}$, що $a^n = 1$, то $n$ називається мультиплікативним порядком елемента $a$.*

**Теорема 63.** *Усі ненульові елементи поля $\mathcal{F}$ мають один і той же адитивний порядок.*

*Доведення.* Нехай $a, b \in \mathcal{F} \setminus \{0\}$ і їхні адитивні порядки дорівнюють $n$ і $m$ відповідно. Тоді

$$n \cdot b = n \cdot (a \cdot a^{-1}) \cdot b = (n \cdot a) \cdot (a^{-1} \cdot b) = 0 \cdot a^{-1} \cdot b = 0.$$

Звідси випливає, що $m \leq n$. Аналогічно

$$m \cdot a = m \cdot (b \cdot b^{-1}) \cdot a = (m \cdot b) \cdot (b^{-1} \cdot a) = 0 \cdot b^{-1} \cdot a = 0.$$

Звідки $n \leq m$, і тому $m = n$. ■

**Означення 69.** *Якщо в полі $\mathcal{F}$ всі ненульові елементи мають адитивний порядок $n$, то поле $\mathcal{F}$ називається полем характеристики $n$. Якщо такого числа $n$ не існує, то поле називається полем характеристики 0.*

**Теорема 64.** *Характеристика довільного скінченного поля є простим числом.*

*Доведення.* Нехай $\mathcal{F}$ – скінченне поле і $a \in \mathcal{F} \setminus \{0\}$ – довільний його елемент. Тоді в послідовності $a, 2a, 3a, \ldots$ існують такі числа $i, j \in \mathcal{Z}, i < j$, що $ja = ia$ або $(j-i)a = 0$. Отже, поле $\mathcal{F}$ має додатну характеристику, яку позначимо $n$. На підставі того, що поле $\mathcal{F}$ включає принаймні два елементи (0 і 1), то $n \geq 2$. Якщо число $n$ не

є простим, то існують такі числа $k, l \in \mathcal{Z}, 1 < k, l < n$, що $n = kl$. Тоді $0 = n \cdot 1 = (k \cdot l) \cdot 1 = (k \cdot l) \cdot (1 \cdot 1) = (k \cdot 1)(l \cdot 1)$. Оскільки поле є областю цілісності, то або $k \cdot 1 = 0$ або $l \cdot 1 = 0$. Звідси дістаємо, що або $ka1 = (k1)a = 0$, або $la1 = (l1)a = 0$ для всіх $a \in \mathcal{F}$. Але це суперечить означенню характеристики поля $n$. ■

З доведеної теореми випливає такий основний результат для скінченних полів.

**Теорема 65.** *Скінченне поле $F$ має характеристику $p$ і порядок $p^n$ для деякого $n \in \mathcal{N}$.*

*Доведення.* З попередньої теореми випливає, що поле $\mathcal{F}$ має характеристику $p$, де $p$ – просте число. Нехай $|\mathcal{F}| = q$. Якщо $q = p$, то твердження теореми, очевидно, виконується. У протилежному випадку візьмемо елемент $a_1 \in \mathcal{F} \setminus \{0\}$ і покладемо

$$G_1 = \{y : y = m_1 \cdot a_1,\ m_1 \in \mathcal{N},\ 1 \leq m_1 \leq p\}.$$

Розглянемо тепер елемент $a_2 \in \mathcal{F} \setminus G_1$ і побудуємо

$$G_2 = \{z : z = m_1 \cdot a_1 + m_2 \cdot a_2,\ m_1, m_2 \in \mathcal{N},\ 1 \leq m_1, m_2 \leq p\}.$$

Якщо $\mathcal{F} \neq G_2$, то розглядається елемент $a_3 \in \mathcal{F} \setminus G_2$ і т. д. На підставі скінченності поля $\mathcal{F}$ такий процес закінчується і ми отримуємо сукупність множин $G_1, G_2, \ldots, G_n$ для деякого $n \in \mathcal{N}$.

Кожний елемент $a \in \mathcal{F}$ єдиним чином зображується у вигляді

$$a = m_1 \cdot a_1 + m_2 \cdot a_2 + \ldots + m_n \cdot a_n,$$

де $1 \leq m_i \leq p$, для всіх $i = 1, 2, \ldots, n$. (Довести це як корисну вправу). Отже, існує $p^n$ таких виразів і тому $|\mathcal{F}| = p^n$. ■

**Означення 70.** *Поле називається простим, якщо воно не має жодного власного підполя.*

Очевидно, що коли поле має простий порядок, то воно є простим полем і порядок простого поля дорівнює характеристиці цього поля.

### 2.9.8. Побудова скінченних полів

Як випливає з теореми 65, простими полями не вичерпуються скінченні поля, оскільки для довільних простого числа $p$ і натурального числа $n$ існує поле, порядок якого $p^n$. Цей більш загальний вигляд скінченних полів будується за допомогою поліномів.

Нехай $\mathcal{F} = (A, \Omega)$ – поле. Нагадаємо, що поліномом над полем називається вираз

$$f(x) = \sum_{i=0}^{n} a_i x^i,$$

де $n \in \mathcal{N}$, $a_i \in A$, $0 \le i \le n$, а $x$ – символ, який не належить полю $\mathcal{F}$. Коефіцієнт $a_n \neq 0$ називається старшим, якщо $n \neq 0$. Число $n$ називається степенем полінома $f(x)$ і позначається $n = deg(f)$. Якщо старшим коефіцієнтом є $a_0$, то поліном $f(x)$ називається константою, а коли старший коефіцієнт $a_0 = 0$, то поліном $f(x)$ називається нульовим $f(x) = 0$. Множину всіх поліномів над полем $\mathcal{F}$ позначатимемо $F(x)$.

Якщо $f(x), g(x) \in F(x)$, де

$$f(x) = \sum_{i=0}^{n} a_i x^i, \qquad g(x) = \sum_{j=0}^{m} b_j x^j,$$

то

$$f(x) + g(x) = \sum_{k=0}^{\max(m,n)} c_k x^k, \tag{2.59}$$

де

$$c_k = \begin{cases} a_k + b_k, & k = 0, 1, \ldots, \min(m, n), \\ a_k, & k = m+1, \ldots, n, \text{ якщо } m < n, \\ b_k, & k = n+1, \ldots, m, \text{ якщо } n < m, \end{cases}$$

і

$$f(x) \cdot g(x) = \sum_{k=0}^{m+n} c_k x^k, \tag{2.60}$$

де $c_k = \sum\limits_{i+j=k,\ 0 \le i,j \le n}^{m+n} a_i b_j.$

**Означення 71.** *Поліном $g(x) \in F(x)$ називається незвідним над полем $\mathcal{F}$, якщо він має додатний степінь і з рівності $g(x) = f(x) \cdot h(x)$, де $f(x), h(x) \in F(x)$, випливає, що або поліном $f(x)$ є константою, або поліном $h(x)$ є константою. У протилежному випадку поліном називається звідним.*

Побудуємо тепер скінченне поле за допомогою незвідного полінома. Зауважимо, що для поліномів $f(x)$ і $g(x)$, де $f(x) \neq 0$, як і при діленні цілих чисел, справедливий розклад

$$g(x) = f(x) \cdot h(x) + r(x), \tag{2.61}$$

де $g(x), r(x) \in F(x)$ і $deg(r) < deg(f)$.

**Означення 72.** *Нехай $f(x), g(x), h(x) \in F(x)$, де $f(x) \neq 0$, задовольняють умові* (2.61). *Тоді поліном $r(x)$ називається остачею від ділення полінома $g(x)$ на поліном $f(x)$. Цей поліном позначається як $r = g \pmod f$*. Остачі від ділення всіх поліномів із множини $F(x)$ за модулем полінома $f(x)$ називаються поліномами із множини $F(x)$ за модулем полінома f(x). Множину всіх таких поліномів позначимо $\mathcal{F}_f(x)$.

Очевидно, що степені всіх поліномів із $\mathcal{F}_f(x)$ менші $deg(f)$.

**Теорема 66.** *Нехай $\mathcal{F}$ – поле, а f(x) – ненульовий поліном із $F(x)$. Тоді $\mathcal{F}_f(x)$ буде полем тоді і тільки тоді, коли f незвідний над полем $\mathcal{F}$.*

*Доведення.* Спочатку зазначимо, що $\mathcal{F}_f(x)$ є кільцем із нулем та одиницею відносно операцій додавання та множення за модулем полінома $f(x)$. Це випливає із співвідношень (2.59), (2.60) і (2.61). Нулем і одиницею виступають 0 і 1 поля $\mathcal{F}$.

Припустимо, що $\mathcal{F}_f(x)$ є полем і $f = g \cdot h$, де $g$ і $h$ – поліноми з множини $F(x)$, які не є константами. Тоді, оскільки $0 < deg(g) < deg(f)$ і $0 < deg(h) < deg(f)$, то поліноми $g$ і $h$ не є константами у множині $\mathcal{F}_f(x)$, не дивлячись на те, що поліном $f$ є нульовим у полі $\mathcal{F}_f(x)$. Але це суперечить замкнутості операції множення

в мультиплікативній групі поля $\mathcal{F}_f(x)$. Отже, $\mathcal{F}_f(x)$ не може бути полем, а це суперечить незвідності полінома $f(x)$ над полем $\mathcal{F}$.

Нехай поліном $f(x)$ незвідний над полем $\mathcal{F}$. Оскільки $\mathcal{F}_f(x)$ кільце, то для доведення теореми потрібно показати, що для довільного ненульового полінома із $\mathcal{F}_f(x)$ у кільці існує елемент, обернений відносно операції множення. Нехай $r$ – ненульовий поліном із $\mathcal{F}_f(x)$ такий, що НСД$(f,r) = c$. Оскільки $f(x)$ незвідний поліном над полем $\mathcal{F}$ і $deg(r) < deg(f)$, то поліном $c$ має бути константою. Розглянемо поліном $h(x) = c \cdot r(x)$, де $c \in \mathcal{F}$, $h(x) \in \mathcal{F}_f(x)$ і НСД$(f,h)$=1. До поліномів, як і до цілих чисел, застосовний алгоритм Евкліда (алгоритм Евкліда для поліномів детально описаний в [11]). За цим алгоритмом обчислюється поліном $h^{-1} \pmod f \in \mathcal{F}_f(x)$. Крім того, оскільки $c \in \mathcal{F}$, то існує такий елемент $c^{-1} \in \mathcal{F}$, що $r^{-1} = c^{-1} \cdot h^{-1} \in \mathcal{F}_f(x)$. ■

Незвідний поліном $f(x)$ називається **визначальним поліномом** поля $\mathcal{F}_f(x)$.

**Теорема 67.** *Нехай $\mathcal{F}_p$ – скінченне поле p-го порядку, де p – просте число, а f – незвідний поліном над полем $\mathcal{F}_p$ степеня n. Тоді $|\mathcal{F}_f(x)| = p^n$.*

*Доведення.* З означення поля $\mathcal{F}_f(x)$ випливає, що множина $F(x)$ складається із поліномів, степінь яких менший $n = deg(f)$, а їхні коефіцієнти належать полю $\mathcal{F}$. Але таких поліномів буде $p^n$. ■

**Наслідок 10.** *Для кожного простого числа p і кожного $n \in \mathcal{N}$ існує єдине з точністю до ізоморфізму скінченне поле, яке складається з $p^n$ елементів.*

**Приклад 2.9.10.** 1. Нехай $\mathcal{F}_2$ – поле лишків за модулем 2. Поліном $f(x) = x^2 + x + 1$ є незвідним над $\mathcal{F}_2$ і множина $\mathcal{F}_f(x)$ буде полем, яке має $2^2$ елементів. Їхні степені менші 2 і тому довільний елемент $y$ із цього поля має вигляд: $y = b_1 x + b_0$, де $b_i \in \mathcal{F}_2$, $i = 0, 1$. Таблиці Келі для поля $\mathcal{F}_f(x)$ набувають вигляду:

| $\oplus$ | 0 | 1 | $x$ | $x+1$ |
|---|---|---|---|---|
| 0 | 0 | 1 | $x$ | $x+1$ |
| 1 | 1 | 0 | $x+1$ | $x$ |
| $x$ | $x$ | $x+1$ | 0 | 1 |
| $x+1$ | $x+1$ | $x$ | 1 | 0 |

,

| $\odot$ | 0 | 1 | $x$ | $x+1$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | $x$ | $x+1$ |
| $x$ | 0 | $x$ | $x+1$ | 1 |
| $x+1$ | 0 | $x+1$ | 1 | $x$ |

.

2. Одним із важливих прикладів простих полів є скінченні кільця лишків за модулем простого числа $p$. Це випливає з такої теореми.

**Теорема 68.** *Кільце лишків $\mathcal{Z}_p$ за модулем $p$ буде полем тоді і тільки тоді, коли $p$ просте число.*

*Доведення.* Для доведення досить установити існування для кожного $s \in \mathcal{Z}_p, s \neq 0$, оберненого елемента $s^{-1} \in \mathcal{Z}_p$.

Розглянемо елементи $s, 2s, \ldots, (p-1)s$. Усі ці елементи різні і відмінні від нуля, оскільки із $s \neq 0$ випливає $ks \neq 0$ для $k = 1, 2, \ldots, p-1$ на підставі простоти числа $p$. А те що вони різні випливає з того, що коли припустити $ks = ls, k < l, l, k < p$, то $(l-k)s = 0$, а це суперечність. Отже, послідовність елементів $s, 2s, \ldots, (p-1)s$ збігається з послідовністю переставлених яким-небудь чином елементів $1, 2, \ldots, p-1$. Зокрема, знайдеться $s', 1 \leq s' \leq p-1$, для якого $ss' = 1$, тобто $s'$ – обернений елемент до елемента $s$. ■

Поле $\mathcal{Z}_p$ будемо позначати $\mathcal{F}_p$. ♠

З теореми 55 випливає така теорема.

**Теорема 69.** *Кожне скінченне просте поле $p$-го порядку ізоморфне полю лишків $\mathcal{F}_p$ [10].*

А теорема 57 показує, що всі ненульові елементи мультиплікативної групи поля – дільники одиниці. Таким чином, розгляд простих полів вигляду $\mathcal{F}_p$ не обмежує загальності. Звідси випливає, що можна вибирати скінченне просте поле $p$-го порядку довільним, а не обов'язково полем лишків за модулем простого числа $p$. Враховуючи ізоморфізм між такими полями, це не вносить яких-небудь принципових змін, але може складати додаткові перешкоди для криптоаналітика. Наприклад, як просте поле можна взяти поле $\mathcal{F}_5$, операції якого були визначені таблицями додавання і множення у прикладі 2.5.9. Відображення $f(2) = 4, f(3) = 2, f(4) = 3, f(0) = 0, f(1) = 1$ – ізоморфізм між полями $\mathcal{F}_5$ і $\mathcal{F}$.

Алгоритми побудови поля $\mathcal{F}_{p^k}$ можна знайти в роботі [17], які ґрунтуються на використанні незвідних над полем $\mathcal{F}_p$ поліномів. Для перевірки незвідності поліномів користуються *тестом Рабіна* [35], який полягає в такому.

Нехай $l_1, l_2, \ldots, l_n$ — прості дільники числа $k$ і $k/l_i = m_i$. Поліном $P(x)$ степеня $k$ над полем $\mathcal{F}_p$ є незвідним тоді і тільки тоді, коли

- $P(x)|(x^{p^k} - x)$, тобто $P(x)$ дільник $x^{p^k} - x$,

• $\gcd(P(x), x^{p^{m_i}} - x) = 1$, для всіх $1 \leq i \leq k$,
де скорочення gcd означає НСД.

**Приклад 2.9.11.** Нехай маємо поліном $P(x) = x^6 + x^5 + 1$ в полі $\mathcal{F}_2$. Ділення полінома $Q(x) = x^{2^6} = x^{64}$ на цей поліном дає в остачі поліном $R(x) = x$, таким чином $P(x)$ задовольняє першій умові тесту Рабіна. Поліноми $x^{2^{6/2}} - x = x^8 - x$ і $x^{2^{6/3}} - x = x^4 - x$ попарно взаємно прості з поліномом $P(x)$, що на підставі тесту Рабіна поліном $x^6 + x^5 + 1$, незвідний у полі $\mathcal{F}_2$.

Якщо розглядається поліном $P(x) = x^2 + x$ у тому ж полі $\mathcal{F}_2$, то остача від ділення полінома $Q(x) = x^{2^2} = x^4$ на $P(x)$ теж дає в остачі поліном $R(x) = x$. Але, $P(x)$ незвідним не буде, оскільки $x^2 + x = x \cdot (x + 1)$. Дійсно, поліноми $x^{2^{2/2}} - x = x^2 - x$ і $P(x) = x^2 + x$ не взаємно прості, а тому друга умова тесту Рабіна не виконується. ♠

Зі сказаного випливає такий алгоритм перевірки полінома на незвідність:

```
function is-irreducible(P(x))
begin
  if x^{p^k} ≡ x (mod P(x)) then
    begin
      irreducible = true;
      for l_i in factors(deg(P(x))) do
        begin
          if gcd(P(x), x^{p^k/l_i} − x (mod P(x))) ≠ 1 then
            begin
              irreducible = false;  break;
            end;
        end;
        return irreducible;
      end
   else return false;
end;
```

Тут $\gcd(P(x), q(x))$ – функція обчислення НСД поліномів алгоритмом Евкліда, $\deg(P(x))$ – степінь полінома $P(x)$, а $\text{factors}(n)$ – функція, яка знаходить прості дільники числа $n$.

Використання в наведеному алгоритмі бінарного піднесення до степеня для обчислення полінома $x^{p^k}$ дає можливість розв'язати порівняння

$$x^{p^k} \equiv x \pmod{P(x)} \text{ і } \gcd(P(x),\, x^{p^{k/l_i}} - x \pmod{P(x)}) \neq 1$$

за не більш, ніж $2 \cdot \log p^k$ операцій. А використання ефективних опе-

рацій множення і ділення поліномів дозволяє отримати алгоритм, складність якого $O(k^2 \log^2 k \log p \log\log k)$ [35].

Розглянемо ще деякі важливі властивості поля $\mathcal{F}_q$, де $q = p^k$ і $p$ – просте число. Зазначимо, що в полі $\mathcal{F}_q$ існує $q-1$ ненульових елементів, які відносно операції множення утворюють абелеву групу. Будемо позначати цю групу $F_q^*$.

**Твердження 7.** *Порядок довільного елемента $a \in F_q^*$ є дільником числа $q-1$.*

*Доведення.* Нехай $d$ – найменший степінь $a$, для якого $a^d = 1$. Такий степінь $d$ дійсно існує на підставі скінченності множини $F_q^*$: різні степені елемента $a$ не можуть бути всі різними і якщо $a^i = a^j, j > i$, то $a^{j-i} = 1$.

Нехай $S$ означає множину $\{1, a, a^2, \ldots, a^{d-1}\}$ всіх різних степенів елемента $a$. Тоді для довільного $b \in F_q^*$ нехай $bS$ означає клас суміжності, який складається з елементів $ba^j$ (наприклад, $1 \cdot S = S$). Очевидно, що два класи суміжності або збігаються, або не мають спільних елементів. Дійсно, якщо $b_1 a^i \in b_1 S$ і $b_1 a^i \in b_2 S$, тобто $b_1 a^i = b_2 a^j$, то для довільного $b_1 a^m$ із $b_1 S$ маємо $b_1 a^m = b_1 a^{i+m-1} = b_2 a^{j+m-i}$. Оскільки кожний клас суміжності складається з $d$ елементів і є розбиттям множини $F_q^*$, то на підставі теореми Лагранжа $d | q-1$. ■

**Означення 73.** *Твірним елементом скінченного поля $\mathcal{F}_q$ називається елемент $g$, порядок якого дорівнює $q-1$, а це еквівалентно тому, що $1, g, g^2, \ldots, g^{q-2}$ є різними елементами групи $F_q^*$.*

**Теорема 70.** *Довільне скінченне поле $\mathcal{F}_q$ має твірний елемент. Якщо $g$ – твірний елемент $F_q^*$, то $g^j$ буде твірним елементом тоді і тільки тоді, коли НСД$(j, q-1) = 1$. Зокрема, в $F_q^*$ існує $\varphi(q-1)$ різних твірних елементів.*

*Доведення.* Припустимо, що елемент $a \in F_q^*$ має порядок $d$, тобто $a^d = 1$ і жодний менший степінь $a$ не дорівнює 1. Тоді на підставі твердження 7 число $d$ є дільником $q-1$. Оскільки $a^d$ найменший степінь, який дорівнює 1, то всі елементи $a, a^2, \ldots, a^d = 1$ різні.

Покажемо, що елементи, порядок яких $d$, – це всі ті $\varphi(d)$ значень $a^j$, для яких НСД$(j,d)$=1. По-перше, оскільки всі $d$ різних степенів різні, то вони є множиною всіх коренів рівняння $x^d = 1$. Отже, довільний елемент, порядок якого дорівнює $d$, повинен знаходитися серед степенів елемента $a$. Але не кожний степінь $a$ має порядок $d$, оскільки коли НСД$(j,d) = d' > 1$, то елемент $a^j$ матиме менший порядок: числа $d/d'$ і $j/d'$ цілі і тому $(a^j)^{d/d'} = (a^d)^{j/d'} = 1$. Навпаки, покажемо, що елемент $a^j$ при НСД$(j,d) = 1$, має порядок $d$. Дійсно, припустимо, що НСД$(j,d)$ =1 і $a^j$ має порядок $d'' < d$. Тоді $(a^{d''})^j =$ $= (a^{d''})^d = 1$ і, отже, $(a^{d''})^{\text{НСД}(j,d)} = a^{d''} = 1$. Але $a^{d''} \neq 1$, оскільки елемент $a$ має порядок $d$. Таким чином, $a^j$ має порядок $d$ тоді і тільки тоді, коли НСД$(j,d)$ =1.

Це означає, що коли маємо елемент, порядок якого $d$, то існує тільки $\varphi(d)$ елементів, порядок яких дорівнює $d$. Отже, для довільного $d|(q-1)$ існує лише дві можливості: або елементів порядку $d$ немає, або таких елементів $\varphi(d)$.

Але кожний елемент має деякий порядок $d|(q-1)$ і або 0, або $\varphi(d)$ елементів мають порядок $d$. На підставі тотожності Гаусса дістаємо

$$\varphi(q-1) = \sum_{d|(q-1)} \varphi(d) = q-1,$$

і права частина дорівнює кількості елементів в $F_q^*$. Звідси і з того, що кожний елемент має певний порядок, випливає, що для кожного $d|(q-1)$ завжди знайдеться $\varphi(d)$ (і ніколи не 0) елементів, порядок яких дорівнює $d$. Зокрема, існує точно $\varphi(q-1)$ елементів, порядок яких $q-1$. З вищедоведеного випливає, що коли елемент $g$ має порядок $q-1$, то всі інші елементи порядку $q-1$ – це елементи $g^j$, для яких НСД$(j,d) = 1$. ■

**Наслідок 11.** *Для довільного простого числа $p$ існує таке число $g$, що його степені пробігають усі ненульові класи лишків за модулем $p$.*

**Приклад 2.9.12.** Нехай $p = 19$, тоді $p-1 = 18$ і всі лишки за модулем 19 степенів числа 2 породжують усі елементи мультиплікативної групи $F_p^*$:

2, 4, 8, 16, 13, 7, 14, 9, 18, 17, 15, 11, 3, 6, 12, 5, 10,1.

Степені числа $4 = 2^2$ не породжуватимуть усі елементи групи $F_p^*$, оскільки НСД(2,18) $= 2 \neq 1$. Дійсно, маємо

4,16,7,9,17,11,6,5,1,4,16,7,9,17,11,6,5,1.

Як бачимо, серед елементів, породжених степенями елемента 4, деяких елементів немає, наприклад, 2, 3, 8, 12. ♠

Підсумовуючи сказане, додамо такі факти, які стосуються скінченних полів.

**Теорема 71.** *а) Скінченне поле $\mathcal{F}_{p^k}$ єдине з точністю до ізоморфізму.*

*б) Скінченне поле $\mathcal{F}_{p^k}$ має просте підполе $\mathcal{F}_p$. Кожне підполе поля $\mathcal{F}_{p^k}$ має порядок $p^m$, де $m$ – дільник $k$.*

*в) Скінченні поля однакових порядків ізоморфні.*

*г) Якщо $a, b \in \mathcal{F}_{p^k}$, то $(a+b)^{p^n} = a^{p^n} + b^{p^n}$ для довільного $n \geq 0$.*

*д) Лінійне рівняння $ax + b = 0$ в полі має єдиний розв'язок.*

*є) Для полінома $g(x) \in \mathcal{F}_{f(x)}$ і елемента $a \in \mathcal{F}$ остача від ділення $g(x)$ на $x - a$ дорівнює $g(a)$, де $f(x)$ – незвідний поліном над полем $\mathcal{F}_p$.*

### 2.9.9. Застосування полів у криптографії

Розглянемо приклад одночасного застосування методів теорії груп і полів у побудові криптографічної системи. Таке застосування, як і у випадку кілець, пов'язане з використанням двох таблиць – таблиці додавання і таблиці множення елементів поля. Ці таблиці дозволяють використовувати гомофонічні шифри та модель математичного сейфа, подібно до того як це робилося в примарних кільцях.

Зауважимо, що оскільки адитивна група поля $\mathcal{F}_{p^k}$ не є повноциклічною, то для роботи в таких полях потрібно мати таблиці операцій на відміну від примарних кілець.

Нехай, як і раніше, літери алфавіту

$$X = \{a, b, c, d, \ldots, x, y, z\}$$

англійської мови лінійно впорядковані таким чином:

(один із можливих варіантів упорядкування літер алфавіту)

| $a$ | $b$ | $c$ | $d$ | $e$ | $f$ | $g$ | $h$ | $i$ | $j$ | $k$ | $l$ | $m$ | $n$ | $o$ | $p$ | $q$ | $r$ | $s$ | $t$ | $u$ | $v$ | $w$ | $x$ | $y$ | $z$ | - |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |

Додамо до 26 літер англійського алфавіту 27-й символ "-" і побудуємо поле $\mathcal{F}_{3^3}$ за допомогою незвідного полінома $f(x) = x^3+2x^2+1$, що дає нижченаведені таблиці додавання і множення. Остачі від ділення на $f(x)$ в таблицях позначено такими числами:

| $x$ | $x+1$ | $x+2$ | $2x$ | $2x+1$ | $2x+2$ | $x^2$ | $x^2+1$ |
|---|---|---|---|---|---|---|---|
| 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| $x^2+2$ | $x^2+x$ | $x^2+x+1$ | $x^2+x+2$ | $x^2+2x$ | $x^2+2x+1$ | $x^2+2x+2$ | $2x^2$ |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| $2x^2+1$ | $2x^2+2$ | $2x^2+x$ | $2x^2+x+1$ | $2x^2+x+2$ | $2x^2+2x$ | $2x^2+2x+1$ | $2x^2+2x+2$ |
| 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |

**Приклад 2.9.13.** Зашифруємо текст "meeting in twelve" з ключовим словом "welcome".

Користуючись таблицями поля $\mathcal{F}_{3^3}$ дістаємо таку шифрограму:

| w | e | l | c | o | m | e | w | e | l | c | o | m | e | w | e | l |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| m | e | e | t | i | n | g | - | i | n | - | t | w | e | l | v | e |
| 22 | 4 | 11 | 2 | 14 | 12 | 4 | 22 | 4 | 11 | 2 | 14 | 12 | 4 | 22 | 4 | 11 |
| 12 | 4 | 4 | 19 | 8 | 13 | 6 | 26 | 8 | 13 | 26 | 19 | 22 | 4 | 11 | 21 | 4 |
| 7 | 8 | 12 | 18 | 10 | 25 | 1 | 9 | 0 | 21 | 25 | 3 | 7 | 8 | 3 | 25 | 12 |
| h | i | m | s | k | z | b | k | a | v | z | d | h | i | d | z | m |

Розшифрування відбувається таким чином: виписуємо цифри, які відповідають ключовому слову, і цифри шифрограми, тобто

| w | e | l | c | o | m | e | w | e | l | c | o | m | e | w | e | l |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 22 | 4 | 11 | 2 | 14 | 12 | 4 | 22 | 4 | 11 | 2 | 14 | 12 | 4 | 22 | 4 | 11 |
| 7 | 8 | 12 | 18 | 10 | 25 | 1 | 9 | 0 | 21 | 25 | 3 | 7 | 8 | 3 | 25 | 12 |
| 12 | 4 | 4 | 19 | 8 | 13 | 6 | 26 | 8 | 13 | 26 | 19 | 22 | 4 | 11 | 21 | 4 |
| m | e | e | t | i | n | g | - | i | n | - | t | w | e | l | v | e |

У рядку таблиці додавання, який має номер 22, знаходимо значення 7. Далі знаходимо номер стовпчика, який відповідає значенню 7. Це буде номер першої літери тексту явного (в даному випадку це буде число 12, якому відповідає літера $m$). Продовжуючи діяти таким чином, знаходимо явний текст.

Адитивна група поля $\mathcal{F}_{3^3}$:

| $\oplus$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 1 | 1 | 2 | 0 | 4 | 5 | 3 | 7 | 8 | 6 | 10 | 11 | 9 | 13 | 14 | 12 | 16 | 17 | 15 | 19 | 20 | 18 | 22 | 23 | 21 | 25 | 26 | 24 |
| 2 | 2 | 0 | 1 | 5 | 3 | 4 | 8 | 6 | 7 | 11 | 9 | 10 | 14 | 12 | 13 | 17 | 15 | 16 | 20 | 18 | 19 | 23 | 21 | 22 | 26 | 24 | 25 |
| 3 | 3 | 4 | 5 | 6 | 7 | 8 | 0 | 1 | 2 | 12 | 13 | 14 | 15 | 16 | 17 | 9 | 10 | 11 | 21 | 22 | 23 | 24 | 25 | 26 | 18 | 19 | 20 |
| 4 | 4 | 5 | 3 | 7 | 8 | 6 | 1 | 2 | 0 | 13 | 14 | 12 | 16 | 17 | 15 | 10 | 11 | 19 | 22 | 23 | 21 | 25 | 26 | 24 | 19 | 20 | 18 |

| 5 | 5 | 3 | 4 | 8 | 6 | 7 | 2 | 0 | 1 | 14 | 12 | 13 | 17 | 15 | 16 | 11 | 9 | 10 | 23 | 21 | 22 | 26 | 24 | 25 | 20 | 18 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6 | 6 | 7 | 8 | 0 | 1 | 2 | 3 | 4 | 5 | 15 | 16 | 17 | 9 | 10 | 11 | 12 | 13 | 14 | 24 | 25 | 26 | 18 | 19 | 20 | 21 | 22 | 23 |
| 7 | 7 | 8 | 6 | 1 | 2 | 0 | 4 | 5 | 3 | 16 | 17 | 15 | 10 | 11 | 9 | 13 | 14 | 12 | 25 | 26 | 24 | 19 | 20 | 18 | 22 | 23 | 21 |
| 8 | 8 | 6 | 7 | 2 | 0 | 1 | 5 | 3 | 4 | 17 | 15 | 16 | 11 | 9 | 10 | 14 | 12 | 13 | 26 | 24 | 25 | 20 | 18 | 19 | 23 | 21 | 22 |
| 9 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 10 | 10 | 11 | 9 | 13 | 14 | 12 | 16 | 17 | 15 | 19 | 20 | 18 | 22 | 23 | 21 | 25 | 26 | 24 | 1 | 2 | 0 | 4 | 5 | 3 | 7 | 8 | 6 |
| 11 | 11 | 9 | 10 | 14 | 12 | 13 | 17 | 15 | 16 | 20 | 18 | 19 | 23 | 21 | 22 | 26 | 24 | 25 | 2 | 0 | 1 | 5 | 3 | 4 | 8 | 6 | 7 |
| 12 | 12 | 13 | 14 | 15 | 16 | 17 | 9 | 10 | 11 | 21 | 22 | 23 | 24 | 25 | 26 | 18 | 19 | 20 | 3 | 4 | 5 | 6 | 7 | 8 | 0 | 1 | 2 |
| 13 | 13 | 14 | 12 | 16 | 17 | 15 | 10 | 11 | 9 | 22 | 23 | 21 | 25 | 26 | 24 | 19 | 20 | 18 | 4 | 5 | 3 | 7 | 8 | 6 | 1 | 2 | 0 |
| 14 | 14 | 12 | 13 | 17 | 15 | 16 | 11 | 9 | 10 | 23 | 21 | 22 | 26 | 24 | 25 | 20 | 18 | 19 | 5 | 3 | 4 | 8 | 6 | 7 | 2 | 0 | 1 |
| 15 | 15 | 16 | 17 | 9 | 10 | 11 | 12 | 13 | 14 | 24 | 25 | 26 | 18 | 19 | 20 | 21 | 22 | 23 | 6 | 7 | 8 | 0 | 1 | 2 | 3 | 4 | 5 |
| 16 | 16 | 17 | 15 | 10 | 11 | 9 | 13 | 14 | 12 | 25 | 26 | 24 | 19 | 20 | 18 | 22 | 23 | 21 | 7 | 8 | 6 | 1 | 2 | 0 | 4 | 5 | 3 |
| 17 | 17 | 15 | 16 | 11 | 9 | 10 | 14 | 12 | 13 | 26 | 24 | 25 | 20 | 18 | 19 | 23 | 21 | 22 | 8 | 6 | 7 | 2 | 0 | 1 | 5 | 3 | 4 |
| 18 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| 19 | 19 | 20 | 18 | 22 | 23 | 21 | 25 | 26 | 24 | 1 | 2 | 0 | 4 | 5 | 3 | 7 | 8 | 6 | 10 | 11 | 9 | 13 | 14 | 12 | 16 | 17 | 15 |
| 20 | 20 | 18 | 19 | 23 | 21 | 22 | 26 | 24 | 25 | 2 | 0 | 1 | 5 | 3 | 4 | 8 | 6 | 7 | 11 | 9 | 10 | 14 | 12 | 13 | 17 | 15 | 16 |
| 21 | 21 | 22 | 23 | 24 | 25 | 26 | 18 | 19 | 20 | 3 | 4 | 5 | 6 | 7 | 8 | 0 | 1 | 2 | 12 | 13 | 14 | 15 | 16 | 17 | 9 | 10 | 11 |
| 22 | 22 | 23 | 21 | 25 | 26 | 24 | 19 | 20 | 18 | 4 | 5 | 3 | 7 | 8 | 6 | 1 | 2 | 0 | 13 | 14 | 12 | 16 | 17 | 15 | 10 | 11 | 9 |
| 23 | 23 | 21 | 22 | 26 | 24 | 25 | 20 | 18 | 19 | 5 | 3 | 4 | 8 | 6 | 7 | 2 | 0 | 1 | 14 | 12 | 13 | 17 | 15 | 16 | 11 | 9 | 10 |
| 24 | 24 | 25 | 26 | 18 | 19 | 20 | 21 | 22 | 23 | 6 | 7 | 8 | 0 | 1 | 2 | 3 | 4 | 5 | 15 | 16 | 17 | 9 | 10 | 11 | 12 | 13 | 14 |
| 25 | 25 | 26 | 24 | 19 | 20 | 18 | 22 | 23 | 21 | 7 | 8 | 6 | 1 | 0 | 2 | 4 | 5 | 3 | 16 | 17 | 15 | 10 | 11 | 9 | 13 | 14 | 12 |
| 26 | 26 | 24 | 25 | 20 | 18 | 19 | 23 | 21 | 22 | 8 | 6 | 7 | 2 | 0 | 1 | 5 | 3 | 4 | 17 | 15 | 16 | 11 | 9 | 10 | 14 | 12 | 13 |

Мультплікативна група поля $\mathcal{F}_{33}$:

| $\odot$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 2 | 0 | 2 | 1 | 6 | 8 | 7 | 3 | 5 | 4 | 18 | 20 | 19 | 24 | 26 | 25 | 21 | 23 | 22 | 9 | 11 | 10 | 15 | 17 | 16 | 12 | 14 | 13 |
| 3 | 0 | 3 | 6 | 9 | 12 | 15 | 18 | 21 | 24 | 11 | 14 | 17 | 20 | 23 | 26 | 2 | 5 | 8 | 19 | 22 | 25 | 1 | 4 | 7 | 10 | 13 | 16 |
| 4 | 0 | 4 | 8 | 12 | 16 | 11 | 24 | 19 | 23 | 20 | 21 | 25 | 5 | 6 | 1 | 17 | 9 | 13 | 10 | 14 | 15 | 22 | 26 | 18 | 7 | 2 | 3 |
| 5 | 0 | 5 | 7 | 15 | 11 | 13 | 21 | 26 | 19 | 2 | 4 | 6 | 17 | 10 | 12 | 23 | 25 | 18 | 1 | 3 | 8 | 16 | 9 | 14 | 22 | 24 | 20 |
| 6 | 0 | 6 | 3 | 18 | 24 | 21 | 9 | 15 | 12 | 19 | 25 | 22 | 10 | 16 | 13 | 1 | 7 | 4 | 11 | 17 | 14 | 2 | 8 | 5 | 20 | 26 | 23 |
| 7 | 0 | 7 | 5 | 21 | 19 | 26 | 15 | 13 | 11 | 1 | 8 | 3 | 22 | 20 | 24 | 16 | 14 | 9 | 2 | 6 | 4 | 23 | 18 | 25 | 17 | 12 | 10 |
| 8 | 0 | 8 | 4 | 24 | 23 | 19 | 12 | 11 | 16 | 10 | 15 | 14 | 7 | 3 | 2 | 22 | 18 | 26 | 20 | 25 | 21 | 17 | 13 | 9 | 5 | 1 | 6 |
| 9 | 0 | 9 | 18 | 11 | 20 | 2 | 19 | 1 | 10 | 17 | 26 | 8 | 25 | 7 | 16 | 6 | 15 | 24 | 22 | 4 | 13 | 3 | 12 | 21 | 14 | 23 | 5 |
| 10 | 0 | 10 | 20 | 14 | 21 | 4 | 25 | 8 | 15 | 26 | 6 | 16 | 1 | 11 | 18 | 12 | 22 | 5 | 13 | 23 | 3 | 24 | 7 | 17 | 2 | 9 | 19 |
| 11 | 0 | 11 | 19 | 17 | 25 | 6 | 22 | 3 | 14 | 8 | 16 | 24 | 13 | 21 | 5 | 18 | 2 | 10 | 4 | 12 | 23 | 9 | 20 | 1 | 26 | 7 | 15 |
| 12 | 0 | 12 | 24 | 20 | 5 | 17 | 10 | 22 | 7 | 25 | 1 | 13 | 15 | 18 | 3 | 8 | 11 | 23 | 14 | 26 | 2 | 4 | 16 | 19 | 21 | 6 | 9 |
| 13 | 0 | 13 | 26 | 23 | 6 | 10 | 16 | 20 | 3 | 7 | 11 | 21 | 18 | 4 | 17 | 14 | 24 | 1 | 5 | 15 | 19 | 25 | 2 | 12 | 9 | 22 | 8 |
| 14 | 0 | 14 | 25 | 26 | 1 | 12 | 13 | 24 | 2 | 16 | 18 | 5 | 3 | 17 | 19 | 20 | 4 | 15 | 23 | 7 | 9 | 10 | 21 | 8 | 6 | 11 | 22 |
| 15 | 0 | 15 | 21 | 2 | 17 | 23 | 1 | 16 | 22 | 6 | 12 | 18 | 8 | 14 | 20 | 7 | 13 | 19 | 3 | 9 | 24 | 5 | 11 | 26 | 4 | 10 | 25 |
| 16 | 0 | 16 | 23 | 5 | 9 | 25 | 7 | 14 | 18 | 15 | 22 | 2 | 11 | 24 | 4 | 13 | 20 | 6 | 21 | 1 | 17 | 26 | 3 | 10 | 19 | 8 | 12 |
| 17 | 0 | 17 | 22 | 8 | 13 | 18 | 4 | 9 | 26 | 24 | 5 | 10 | 23 | 1 | 15 | 19 | 6 | 14 | 12 | 20 | 7 | 11 | 25 | 3 | 16 | 21 | 2 |
| 18 | 0 | 18 | 9 | 19 | 10 | 1 | 11 | 2 | 20 | 22 | 13 | 4 | 14 | 5 | 23 | 3 | 21 | 12 | 17 | 8 | 26 | 6 | 24 | 15 | 25 | 16 | 7 |
| 19 | 0 | 19 | 11 | 22 | 14 | 3 | 17 | 6 | 25 | 4 | 23 | 12 | 26 | 15 | 7 | 9 | 1 | 20 | 8 | 24 | 16 | 18 | 10 | 2 | 13 | 5 | 21 |
| 20 | 0 | 20 | 10 | 25 | 15 | 8 | 14 | 4 | 21 | 13 | 3 | 23 | 2 | 19 | 9 | 24 | 17 | 7 | 26 | 16 | 6 | 12 | 5 | 22 | 1 | 18 | 11 |
| 21 | 0 | 21 | 15 | 1 | 22 | 16 | 2 | 23 | 17 | 3 | 24 | 9 | 4 | 25 | 10 | 5 | 26 | 11 | 6 | 18 | 12 | 7 | 19 | 13 | 8 | 20 | 14 |
| 22 | 0 | 22 | 17 | 4 | 26 | 9 | 8 | 18 | 13 | 12 | 7 | 20 | 16 | 2 | 21 | 11 | 3 | 25 | 24 | 10 | 5 | 19 | 14 | 6 | 23 | 15 | 1 |
| 23 | 0 | 23 | 16 | 7 | 18 | 14 | 5 | 25 | 9 | 21 | 17 | 1 | 19 | 12 | 8 | 26 | 10 | 3 | 15 | 2 | 22 | 13 | 6 | 20 | 11 | 4 | 24 |
| 24 | 0 | 24 | 12 | 10 | 7 | 22 | 20 | 17 | 5 | 14 | 2 | 26 | 21 | 9 | 6 | 4 | 19 | 16 | 25 | 13 | 1 | 8 | 23 | 11 | 15 | 3 | 18 |
| 25 | 0 | 25 | 14 | 13 | 2 | 24 | 26 | 12 | 1 | 23 | 9 | 7 | 6 | 22 | 11 | 10 | 8 | 21 | 16 | 5 | 18 | 20 | 15 | 4 | 3 | 19 | 17 |
| 26 | 0 | 26 | 13 | 16 | 3 | 20 | 23 | 10 | 6 | 5 | 19 | 15 | 9 | 8 | 22 | 25 | 12 | 2 | 7 | 21 | 11 | 14 | 1 | 24 | 18 | 17 | 4 |

Зашифрувати цей текст можна, користуючись таблицею множення:

| w | e | l | c | o | m | e | w | e | l | c | o | m | e | w | e | l |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| m | e | e | t | i | n | g | - | i | n | - | t | w | e | l | v | e |
| 22 | 4 | 11 | 2 | 14 | 12 | 4 | 23 | 4 | 11 | 2 | 14 | 12 | 4 | 23 | 4 | 11 |
| 12 | 4 | 4 | 19 | 8 | 13 | 6 | 26 | 8 | 13 | 26 | 19 | 22 | 4 | 11 | 21 | 4 |
| 16 | 16 | 25 | 11 | 2 | 18 | 24 | 24 | 23 | 21 | 13 | 7 | 16 | 16 | 1 | 22 | 25 |
| q | q | z | l | c | s | y | y | x | v | n | h | q | q | b | w | z |

. ♠

У скінченних полях, подібно до того, як це робилося у скінченних кільцях, теж можна використати модель математичного сейфа. У цьому випадку система $Ax + \bar{b} = \bar{c}$ повинна мати розв'язок, який відкриває сейф і який залежить від ізоморфізму між скінченними полями одного порядку й упорядкування символів алфавіту. Звідси випливає, що **простір ключів** такої криптосистеми дорівнює кількості способів упорядкування алфавіту, який має 27 елементів, а це 27!=10888864450418352160768000000 способів, та 25! автоморфізмів поля $\mathcal{F}_{3^3}$. Але основною перевагою такої криптосистеми є те, що мультиплікативна група поля циклічна і має максимальний порядок, а це дозволяє повною мірою застосовувати функцію дискретного логарифма.

**Контрольні запитання**

1. Яке відношення називається відношенням конгруентності?
2. Навести означення групи, абелевої групи та підгрупи.
3. Яке найбільше число підгруп може мати група четвертого порядку?
6. Навести означення гомоморфізму (ізоморфізму) груп, кілець, полів, нормального дільника й ідеалу.
7. Навести означення дільника одиниці і дільника нуля.
8. Які властивості мають ці елементи?
9. Навести приклад некомутативного кільця з дільниками нуля.
10. Навести означення області цілісності.
11. Чи буде скінченна область цілісності полем?
12. Навести приклад неасоціативного і некомутативного кільця.
13. Навести приклад комутативного неасоціативного кільця.
14. Чи буде полем фактор-кільце за максимальним ідеалом?
15. Скільки існує ізоморфних скінченних полів, порядок яких $p^n$?

**Задачі і вправи**

1. На множині $A = \{1, 2, 3, 4, 6, 12\}$ визначені бінарні операції $a * b$ =НСД$(a, b)$ і $a \circ b$ =НСК$(a, b)$. Побудувати таблиці Келі для цих операцій. Чи буде ця алгебра кільцем?

2. На множині $A$ задана бінарна операція * з такими властивостями: $(\forall x, y, z \in \in A)$ $x * (y * z) = y * (z * x)$, а із $x * y = x * z$ випливає $y = z$. Довести, що операція * комутативна й асоціативна.

3. На множині дійсних чисел $D$ визначена бінарна операція * з такими властивостями: $(\forall x, y, z \in D)$ $x * (x * x) = x$, $x * (y * z) = x * y + z$. Довести, що $x * x = 0$, $x * 0 = x$, $x * y = x - y$.

4. На множині $(R \setminus \{0\}) \times R$, де $R$ – множина раціональних чисел, визначена операція $(a, b) * (c, d) = (ac, bc + d)$. Довести, що

а) алгебра $G = (\{(R \setminus \{0\}) \times R\}, \{*\})$ буде групою;

б) у таблиці множення скінченної групи кожний елемент зустрічається в кожному рядку і кожному стовпчику рівно один раз;

г) коли циклічна група має порядок $n$ і $d$ – дільник $n$, то група має лише одну підгрупу порядку $d$;

д) коли порядок циклічної групи дорівнює $n$, то вона має $\varphi(n)$ твірних елементів.

5. Знайти всі підгрупи групи $\mathcal{Z}_{12}$ лишків відносно операції додавання за модулем 12. Знайти всі твірні цієї групи. Чи буде ця група циклічною?

Довести, що довільна ненульова підгрупа адитивної циклічної групи є циклічною групою.

6. Чи може група бути ізоморфною своїй власній підгрупі? Навести приклад і відповідне обґрунтування.

7. Нехай $G$ – група і $a$ – її фіксований елемент. Якщо $x$ – довільний елемент групи $G$, то визначимо відображення $f : G \to G$ формулою $f(x) = axa^{-1}$. Довести, що $f$ – автоморфізм групи $G$.

8. Довести, що довільна скінченна група, яка складається з $n$ елементів, ізоморфна симетричній групі підстановок деякої множини $M$ із $n$ елементів.

9. а) Побудувати групу підстановок, яка ізоморфна циклічній групі четвертого порядку $G = \{e, a, a^2, a^3\}$;

б) Перевірити, чи виконуються такі рівності в цій групі підстановок $f_2^2 = f_3,\ f_3^2 = e,\ f_2^3 = f_4,\ f_2 f_4 = e$, де $f_1 = e$ – тотожна підстановка, а $f_2,\ f_3,\ f_4$ – решта елементів групи підстановок.

10. Довести, що скінченна напівгрупа з одиницею і законом (лівого або правого) скорочення є групою.

11. Довести, що підгрупа $H$ групи $G$ індексу 2 – нормальний дільник групи $G$.

12. Нехай група $G = \{e, g_1, g_2, \ldots\}$ і $x \in G$ – довільний елемент групи. Довести, що множина $S = \{e, xg_1x^{-1}, xg_2x^{-1}, \ldots\}$ включає всі елементи групи $G$.

13. Нехай $R$ і $S$ – підгрупи групи $G$. Визначимо добуток двох підгруп $R$ і $S$ таким чином: $R \cdot S = \{rs | r \in R\ i\ s \in S\}$. Довести, що

а) множина $R \cdot S$ буде підгрупою групи $G$ тоді і тільки тоді, коли $R \cdot S = S \cdot R$;

б) якщо одна із підгруп ($R$ чи $S$) – нормальний дільник групи $G$, то $R \cdot S = S \cdot R$ буде підгрупою групи $G$.

14. Побудувати абелеві групи, які визначені такими рядками додавання:

а) $0+1=1, 1+1=5, 1+2=6, 1+3=0, 1+4=2, 1+5=4, 1+6=3$;

б) $0+1=1, 1+1=3, 1+2=0, 1+3=4, 1+4=6, 1+5=2, 1+6=5$;

в) $0+1=1, 1+1=3, 1+2=4, 1+3=2, 1+4=5, 1+5=6, 1+6=0$.

Відповідь часткова для груп а) і б):

| $\oplus$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 1 | 3 | 0 | 4 | 6 | 2 | 5 |
| 2 | 2 | 0 | 5 | 1 | 3 | 6 | 4 |
| 3 | 3 | 4 | 1 | 6 | 5 | 0 | 2 |
| 4 | 4 | 6 | 3 | 5 | 2 | 1 | 0 |
| 5 | 5 | 2 | 6 | 0 | 1 | 4 | 3 |
| 6 | 6 | 5 | 4 | 2 | 0 | 3 | 1 |

;

| $\oplus$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 1 | 5 | 6 | 0 | 2 | 4 | 3 |
| 2 | 2 | 6 | 1 | 4 | 0 | 3 | 5 |
| 3 | 3 | 0 | 4 | 6 | 5 | 1 | 2 |
| 4 | 4 | 2 | 0 | 5 | 3 | 6 | 1 |
| 5 | 5 | 4 | 3 | 1 | 6 | 2 | 0 |
| 6 | 6 | 3 | 5 | 2 | 1 | 0 | 4 |

.

Побудувати кільця за знайденими групами.

д) Пояснити неможливість побудови групи за такого означення операції додавання: $0+1=1, 1+1=2, 1+2=0, 1+3=4, 1+4=3$.

15. Знайти розклад $(a+b)^3$ в некомутативному кільці.

Побудувати таблиці операцій фактор-кільця кільця $KGN_{25}$ з пункту 2.9.5 за ідеалом $J_0 = \{0, 2, 9, 18, 19\}$ і показати, що отримане кільце буде полем.

Відповідь: [0]=$J_0 = \{0, 2, 9, 18, 19\}$, [1]=$\{1 + J_0\} = \{1, 4, 11, 20, 21\}$, [3]=$\{3 + J_0\} = \{3, 6, 12, 13, 24\}$, [5]=$\{5 + J_0\} = \{5, 15, 14, 8, 22\}$, [7]=$\{7 + J_0\} = \{7, 12, 13, 10, 23\}$. Таблиці операцій:

| + | 0 | 1 | 3 | 5 | 7 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 3 | 5 | 7 |
| 1 | 1 | 3 | 5 | 7 | 0 |
| 3 | 3 | 5 | 7 | 0 | 1 |
| 5 | 5 | 7 | 0 | 1 | 3 |
| 7 | 7 | 0 | 1 | 3 | 5 |

,

| * | 0 | 1 | 3 | 5 | 7 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 3 | 5 | 7 |
| 3 | 0 | 3 | 7 | 1 | 6 |
| 5 | 0 | 5 | 1 | 7 | 3 |
| 7 | 0 | 7 | 5 | 3 | 1 |

.

Порядок фактор-кільця 5 і тому воно буде полем.

16. Побудувати поле, порядок якого $3^2$, над полем $\mathcal{F}_3$ лишків за модулем 3.

17. Знайти підстановку $x$ із рівняння $f \cdot x \cdot f_1 = f_2$, якщо

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 7 & 3 & 2 & 1 & 6 & 5 & 4 \end{pmatrix}, \quad f_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 1 & 2 & 7 & 4 & 5 & 6 \end{pmatrix},$$

$$f_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 5 & 1 & 3 & 4 & 4 & 7 & 2 \end{pmatrix}.$$

18. Знайти розклад на цикли підстановок

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 2 & 4 & 5 & 1 & 6 \end{pmatrix}, \quad f_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 2 & 1 & 5 & 4 & 6 \end{pmatrix}.$$

19. Довести, що коли підстановка $f$ має розклад на цикли $s_1 s_2 \cdots s_m$, то підстановка $f^k$ матиме розклад на цикли $s_1^k s_2^k \cdots {}_m^k$.

20. Довести, що коли цикл $s$ має довжину $k$, то $s^k = \varepsilon$, де $\varepsilon$ – тотожна підстановка. Користуючись цією властивістю, знайти $f^{100}$, де

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 3 & 5 & 4 & 1 & 7 & 10 & 2 & 6 & 9 & 8 \end{pmatrix}.$$

Відповідь: (див. теорему 52) $f^{100} = f$.

21. Розглянемо множину цілих чисел $\mathcal{Z}$ з такими операціями: $m \oplus n = m + n - 1$ і $m \odot n = m + n - mn$, тобто алгебру $G = (\mathcal{Z}, \Omega = \{\oplus, \odot\})$.

а) Довести, що $G$ є кільцем;

б) Знайти нульовий і одиничний елементи цього кільця;

в) Чи утворює множина непарних чисел підкільце кільця $G$?

г) Чи має це кільце дільники нуля? Знайти ідеали цього підкільця.

22. Нехай $G = (B(U), \Omega = \{\div, \cap\})$, де $B(U)$ – булеан множини $U$, а $\div$ і $\cap$ – операції симетричної різниці і перетину множин відповідно.

а) Довести, що $G$ – комутативне кільце з одиницею;

б) Знайти вигляд оберненого елемента для даного елемента $A \subseteq U$;

в) Побудувати таблиці Келі для операцій цього кільця, якщо $U = \{1, 2\}$, і знайти його ідеали.

23. Нехай $G$ – кільце з одиницею і $a, b \in G$ – дільники одиниці цього кільця. Довести, що

а) $(ab)^{-1} = b^{-1}a^{-1}$; б) одиниця кільця не є дільником нуля;

в) коли $a$ дільник одиниці, то і $-a$ дільник одиниці.

24. Скільки дільників одиниці і нуля має кільце лишків: а) $\mathcal{Z}_8$; б) $\mathcal{Z}_{15}$; в) $\mathcal{Z}_{23}$?

25. Довести, що множина дільників одиниці кільця утворює абелеву групу відносно операції множення.

26. Довести, що коли кільце $K$ з одиницею і $J$ його ідеал, то фактор-кільце $K/J$ теж має одиницю.

27. Довести, що в полі $\mathcal{F}_{p^k}$ має місце розклад $(a+b+\cdots+c)^p = a^p + b^p + \cdots + c^p$.

28. Довести або спростувати, що множина відображень вигляду $f_{a,b}(x) = a + bx$, де $a, b, x \in \mathcal{F}_p$, утворює групу відносно операції суперпозиції відображень.

29. Довести, що множина матриць

$$\begin{pmatrix} a & b \\ -b & a \end{pmatrix},$$

де $a, b \in \mathcal{F}_3$, утворює поле порядку 9, а його мультиплікативна група циклічна і має порядок 8.

30. Довести теорему 62.

## 2.10. Системи обміну ключами

Познайомившись з основними теоретико-числовими й алгебраїчними поняттями, розглянемо алгоритми обміну інформацією між абонентами, які ґрунтуються на властивостях чисел, груп, кілець і полів. Перш за все, розглянемо розв'язання проблеми обміну таємними ключами між абонентами.

Розглянемо алгоритми, які розв'язують цю проблему і вважаються безпечними. Це алгоритми Діффі-Хеллмана, Шаміра та Ель-Гамаля [29], які в певному сенсі призвели в кінці 70-х рр. минулого століття до революції у криптографії. Ці алгоритми ґрунтуються на властивості дискретного логарифма, який є прикладом *односторонньої функції* (див. приклади $NPC$-проблем на стор. 59).

Функція дискретного алгоритму діє в полі $\mathcal{F}_q$ і має вигляд $y = g^x \pmod q$, де $q = p^n$, $p$ – деяке просте число, а $x$ – ціле число з множини $\{1, 2, \ldots, q-1\}$. Обернена функція має вигляд $x = \log_g y \pmod q$ і називається *дискретним логарифмом*. Далі для простоти будемо розглядати поле $\mathcal{F}_p$, тобто коли $n = 1$. Для забезпечення складності обчислення числа $x$ за умови використання кращих комп'ютерів нині використовують числа, які мають розміри не менше 1024 біт.

Покажемо, що обчислення значення $y$ виконується досить швидко, на прикладі обчислення числа $g^{16} \pmod p$. Запишемо цей вираз таким чином: $g^{16} \pmod p = (((g^2)^2)^2)^2 \pmod p$, звідки бачимо, що обчислення значення даної функції виконується всього за 4 операції множення. А при послідовному обчисленні для цього потрібно було б виконати 15 таких операцій.

Розглянемо детальніше алгоритм обчислення числа $y$ (див. алгоритми з пункту 2.5.9). Для цього введемо величину $t = \lfloor \log_2 x \rfloor$ – цілу частину $\log_2 x$, і обчислимо числа ряду

$$g,\ g^2,\ g^4,\ g^8, \ldots, g^{2^t} \pmod p. \tag{2.62}$$

У цьому ряду кожне число знаходиться шляхом множення попереднього числа на самого себе за модулем $p$. Запишемо показник степеня $x$ у вигляді двійкового числа: $x = (x_t x_{t-1} \ldots x_1 x_0)$. Тоді

число

$$y = \prod_{i=0}^{t} [g^{x_i 2^i} (\mathrm{mod}\ p)]. \tag{2.63}$$

Наприклад, якщо потрібно обчислити число $3^{100} \pmod 7$, то знаходимо $t = \lfloor \log_2 100 \rfloor = 6$. Обчислюємо числа ряду (2.62):

$$\begin{array}{ccccccc} g & g^2 & g^4 & g^8 & g^{16} & g^{32} & g^{64} \\ 3 & 2 & 4 & 2 & 4 & 2 & 4 \end{array}.$$

Двійкове число для показника має вигляд: $100_2 = 1100100$ і, виконуючи обчислення за формулою (2.63), дістаємо

$$\begin{array}{cccccccc} g^{64} & g^{32} & & & g^4 & & & \\ 4 & 2 & 1 & 1 & 4 & 1 & 1 & = 4. \end{array}$$

Для цього обчислення знадобилося лише 8 операцій множення (6 операцій для обчислення елементів першого ряду та 2 операції для обчислення елементів другого ряду).

Часову характеристику складності обчислення $y = g^x \pmod p$ дає таке твердження.

**Твердження 8.** *Кількість операцій множення, необхідних для обчислення значення $y = g^x \pmod p$ описаним методом, не перевищує $2 \log_2 x$.*

*Доведення.* Для обчислення чисел ряду (2.63) потрібно $t$ множень, для обчислення значення $y$ за наведеною формулою теж потрібно не більше $t$ множень. Оскільки $t = \lfloor \log_2 x \rfloor < \log_2 x$, то вказана оцінка справедлива. ■

Для обчислення значень оберненої функції дискретного логарифма невідомі ефективні алгоритми. Одним із методів обчислення її значень є метод "крок немовляти, крок гіганта", який описано далі. Цей метод потребує $2\sqrt{p}$ операцій множення і нижче в таблиці показано деякі результати таких підрахунків. Звідси можна зробити висновок, що при великих значеннях числа $p$ функція дискретного логарифма дійсно буде односторонньою, якщо для її обчислення використовується метод "крок немовляти, крок гіганта".

| Кількість десяткових знаків числа $p$ | Обчислення $y$ ($2\log p$ множ.) | Обчислення $x$ ($2\sqrt{p}$ множ.) |
|---|---|---|
| 12 | $2 \cdot 40 = 80$ | $2 \cdot 10^6$ |
| 60 | $2 \cdot 200 = 400$ | $2 \cdot 10^{30}$ |
| 90 | $2 \cdot 300 = 600$ | $2 \cdot 10^{45}$ |

.

Нехай суперкомп'ютер виконує множення двох 90-розрядних чисел у часі $10^{-14}$ секунд (для сучаних комп'ютерів цей час не досяжний). Тоді для обчислення $y$ такому комп'ютеру потрібен час $600 \cdot 10^{-14} = 6 \cdot 10^{-12}$ секунд, а для обчислення значення $x$ – час $10^{45} \cdot 10^{-14} = 10^{31}$ секунд.

### 2.10.1. Протокол обміну ключами Діффі-Хеллмана

Спочатку в цьому протоколі всі учасники домовляються про те, яке буде використовуватися поле $\mathcal{F}_p$ і породжуючий елемент $1 < g < p-1$ його мультиплікативної групи. Нехай вибране поле $\mathcal{F}_p$, де $p$ – велике просте число і породжуючий елемент $g$ (про спосіб вибору елементів $p$ і $g$ буде сказано нижче):

$$g \pmod p,\ g^2 \pmod p,\ \ldots,\ g^{p-1} \pmod p.$$

Числа $p$ і $g$ відомі всім абонентам.

**ПРОТОКОЛ ОБМІНУ КЛЮЧАМИ ДІФФІ-ХЕЛЛМАНА**

**Вхід**: пара $(p, g)$, де $p$ – велике просте число, а $g$ – породжуючий елемент групи поля $\mathcal{F}_p$ $(1 < g < p-1)$.

**Вихід**: елемент групи $k$, яким необхідно обмінятися абонентам $A$ і $B$.

1. $A$ генерує елемент $a \in [1, p-1)$, обчислює число $g_a = g^a \pmod p$ і висилає його абоненту $B$.

2. $B$ генерує елемент $b \in [1, p-1)$, обчислює число $g_b = g^b \pmod p$ і висилає його абоненту $A$.

3. $A$ обчислює число $k = g_b^a \pmod p$.

4. $B$ обчислює число $k = g_a^b \pmod p$.

Таким чином обмін ключем $k$ між абонентами $A$ і $B$ відбувся.

Розглянемо роботу цього протоколу на прикладі обміну ключем між трьома абонентами $A, B, C$. Абоненти $A, B, C$ вибирають великі приватні (таємні) числа $X_A, X_B, X_C$, в той час як числа $p, g$ відомі

всім абонентам $A, B, C$. Кожний абонент обчислює число $Y_X$, яке висилає всім абонентам. Число $Y_X$ обчислюється таким чином:

$Y_A = g^{X_A} \pmod p$, $Y_B = g^{X_B} \pmod p$, $Y_C = g^{X_C} \pmod p$.

Звідси дістаємо таку таблицю:

| Абонент | Ключ приватний | Ключ відкритий |
|---|---|---|
| $A$ | $X_A$ | $Y_A$ |
| $B$ | $X_B$ | $Y_B$ |
| $C$ | $X_C$ | $Y_C$ |

.

Нехай $A$ хоче передати $B$ повідомлення. Для цього він висилає до $B$ ключ $Y_A$, за допомогою якого шифрується повідомлення.

Оскільки інформація про $p$ і $g$ відома всім абонентам, то $A$ висилає до $B$ відкритим каналом інформацію про те, що він хоче вислати повідомлення. Потім $A$ обчислює $Z_{AB} = (Y_B)^{X_A} \pmod p$.

Жодна особа, крім $A$, такого обчислення не може виконати, оскільки $X_A$ є ключем приватним.

Абонент $B$ обчислює число $Z_{BA} = (Y_A)^{X_B} \pmod p$.

Обґрунтування такого способу дає така теорема.

**Теорема 72.** $Z_{AB} = Z_{BA}$.

*Доведення.* На підставі властивостей мультиплікативної групи поля $\mathcal{F}_p$, отримуємо

$Z_{AB} = (Y_B)^{X_A} \pmod p = (g^{X_B})^{X_A} \pmod p = (g^{X_A})^{X_B} \pmod p =$
$= (Y_A)^{X_B} \pmod p = Z_{BA}$. ■

Основні властивості протоколу Діффі-Хеллмана:

– $A$ і $B$ отримали одне і те саме число $Z = Z_{AB} = Z_{BA}$;

– особі небажаній числа $X_A$ і $X_B$ не відомі і вона не має можливості обчислити число $Z$ (принаймні за розумний відрізок часу).

**Вибір елемента** $g$. Як було сказано, стійкість протоколу Діффі-Хеллмана до зламання ґрунтується на складності функції дискретного логарифма. Аби ця стійкість була високою, належить вибирати просте число $p$ таким, щоб число $p-1$ мало великий простий дільник $p'$ (великий означає $p' > 2^{160}$). Вибір числа $p$ можна виконати так:

$$p = 2r + 1 \text{ або } p - 1 = 2r.$$

де $r$ – теж просте число. Якщо число $p$ вибрано таким чином, то елемент $g$ може бути довільним елементом, що задовольняє нерівності: $1 < g < p-1$ і $g^r \not\equiv 1 \pmod{p}$. Елемент $g$ не обов'язково повинен бути породжуючим елементом усієї мультиплікативної групи поля $\mathcal{F}_q$. Необхідно тільки, щоб він був породжуючим її підгрупи, порядок якої великий, наприклад, $p'$.

**Приклад 2.10.1.** Нехай $p = 23 = 2 \cdot 11 + 1$, тобто $r = 11$. Вибираємо $g$. Якщо $g = 3$, то $3^{11} \equiv 1 \pmod{23}$ і тоді $g = 3$ не задовольняє умові вибору. Нехай $g = 5$, тоді $5^{11} \equiv 22 \pmod{23}$ і тому $g = 5$ є шуканим елементом.

Тепер кожний абонент обчислює приватний ключ. Припустимо, що були вибрані числа $X_A = 7, X_B = 13$. Обчислюємо

$Y_A = 5^7 \pmod{23} = 17$, $Y_B = 5^{13} \pmod{23} = 21$.

Якщо $A$ і $B$ вирішили згенерувати спільний ключ, то $A$ обчислює

$$Z_{AB} = (Y_B)^{X_A} \pmod{p} = (21)^7 \pmod{23} = 10,$$

а $B$ обчислює

$$Z_{BA} = (Y_A)^{X_B} \pmod{p} = (17)^{13} \pmod{23} = 10.$$

Отже, $A$ і $B$ мають спільний ключ, який не передавався відкритими каналами. ♠

### 2.10.2. Протокол обміну ключами Шаміра

Цей протокол був першим шифром, який давав можливість обмінюватися повідомленнями відкритими лініями зв'язку для абонентів, які не мають ніяких захищених каналів і секретних ключів і, можливо, ніколи не бачилися. Описана вище система Діффі-Хеллмана дає можливість лише сформувати секретне слово, а передача повідомлення потребує певного шифру, в якому це секретне слово діятиме як ключ.

Шифр Шаміра має такий вигляд. Нехай два абоненти $A$ і $B$ зв'язані між собою лінією зв'язку. $A$ хоче передати повідомлення $m$ абоненту $B$ так, щоб його ніхто не зміг прочитати. $A$ вибирає випадково велике просте число $p$ і відкрито передає його $B$. Потім $A$ вибирає два числа $c_A$ і $d_A$ такі, що

$$c_A \cdot d_A \equiv 1 \pmod{(p-1)}. \tag{2.64}$$

Ці числа $A$ тримає в секреті і нікому їх не передає. $B$ теж вибирає два числа $c_B$ і $d_B$ такі, що

$$c_B \cdot d_B \equiv 1 \pmod{(p-1)}. \tag{2.65}$$

і теж тримає їх у секреті.

Після цього $A$ передає своє повідомлення $m$, використовуючи триступеневий протокол. Якщо $m < p$ ($m$ розглядається як число), то повідомлення $m$ передається відразу, а якщо $m \geq p$, то повідомлення подається у вигляді блоків $m_1, m_2, \ldots, m_t$, де всі $m_i < p$, а далі передаються послідовно $m_1, m_2, \ldots, m_t$. Зазначимо, що для кодування кожного $m_i$ краще вибирати випадково нові пари $(c_A, d_A)$ і $(c_B, d_B)$. У протилежному випадку стійкість системи знижується. Отже, основним є випадок $m < p$, який розглянемо детальніше. Протокол обміну в шифрі є таким:

**крок 1**. $A$ обчислює число $x_1 \equiv m^{c_A} \pmod{p}$, де $m$ – початкове повідомлення, і передає його $B$;

**крок 2**. $B$, отримавши $x_1$, обчислює число

$$x_2 \equiv x_1^{c_B} \pmod{p} \tag{2.66}$$

і передає його $A$;

**крок 3**. $A$ обчислює число $x_3 \equiv x_2^{d_A} \pmod{p}$ і передає його $B$;

**крок 4**. $B$, отримавши $x_3$, обчислює число

$$x_4 \equiv x_3^{d_B} \pmod{p}. \tag{2.67}$$

**Теорема 73 (про властивість протоколу Шаміра).**

*а) $x_4 = m$, тобто дійсно $B$ отримав від $A$ початкове повідомлення.*

*б) зловмисник не може прочитати передане повідомлення.*

*Доведення.* а) Відомо, що довільне число $e \geq 0$ можна подати у вигляді $e = k(p-1) + r$, де $r \equiv e \pmod{(p-1)}$. Тоді на підставі малої теореми Ферма дістаємо

$$x^e \pmod{p} = x^{k(p-1)+r} \pmod{p} = (1^k x^r) \pmod{p} = x^{e \pmod{(p-1)}} \pmod{p}. \tag{2.68}$$

Тепер справедливість пункту а) випливає з такої послідовності рівностей:

$$x_4 \equiv x_3^{d_B} \pmod p = (x_2^{d_A})^{d_B} \pmod p = (x_1^{c_B})^{d_A d_B} \pmod p = \\ = (m^{c_A})^{c_B d_A d_B} \pmod p = m^{c_A c_B d_A d_B} \pmod p = \\ = m^{(c_A c_B d_A d_B)} \pmod{(p-1)} \pmod p = m.$$

Передостання рівність випливає із (2.68), а остання – із (2.64) і (2.65).

б) Доведення ґрунтується на припущенні, що для зловмисника, який намагається прочитати $m$, не існує ефективнішої стратегії, ніж наступна стратегія. Спочатку він знаходить число $c_B$ із (2.66), потім число $d_B$ і, нарешті, обчислює $x_4 = m$ за формулою (2.67). Але для реалізації цієї стратегії зловмисник мусить розв'язати задачу дискретного логарифма (2.66), що практично неможливо зробити при великому значенні $p$. ■

**Метод вибору чисел $c_A$ і $d_A$**, які задовольняють (2.64) і (2.65), опишемо тільки для дій абонента $A$, оскільки дії абонента $B$ аналогічні.

Число $c_A$ вибирається випадково так, щоб воно було взаємно простим із числом $p-1$ (шукати потрібно серед непарних чисел, оскільки число $p-1$ парне). Потім обчислюється число $d_A$ за допомогою узагальненого алгоритму Евкліда, який був наведений вище.

**Приклад 2.10.2.** Нехай $A$ хоче передати $B$ повідомлення $m = 10$. $A$ вибирає числа $p = 23$, $c_A = 7$ (НСД(7,22) $= 1$) і обчислює $d_A = 19$ ($7 \cdot d_A \equiv 1 \pmod{22}$, звідки $d_A = 19$).

Аналогічно $B$ вибирає число $c_B = 5$ (яке взаємно просте із числом 22) і число $d_B = 9$ ($5 \cdot d_B \equiv 1 \pmod{22}$, звідки $d_B = 9$).

Реалізується протокол Шаміра:

*крок 1.* $x_1 \equiv 10^7 \pmod{23} = 14$;

*крок 2.* $x_2 \equiv 14^5 \pmod{23} = 15$;

*крок 3.* $x_3 \equiv 15^{19} \pmod{23} = 19$;

*крок 4.* $x_4 \equiv 19^9 \pmod{23} = 10$.

Таким чином, $B$ отримав повідомлення $m = 10$. ♠

### 2.10.3. Протокол обміну ключами Ель-Гамаля

Нехай абоненти $A, B, C$ хочуть обмінюватися між собою повідомленнями через відкритий канал зв'язку. Такий обмін можна виконувати за допомогою протоколу, запропонованого Ель-Гамалем, який дає можливість передавати повідомлення за допомогою лише

одного пересилання.

Для абонентів $A, B, C$ вибирається велике просте число $p$ і число $g$ (число $g$ вибирається так, як у протоколі Діффі-Хеллмана). Числа $p$ і $g$ висилаються всім абонентам. Після отримання цих чисел кожен абонент вибирає приватне число $c_i,\ 1 < c_i < p-1$, і обчислює число $d_i = g^{c_i} \pmod p$. Результати обчислень наведено в таблиці:

| Абонент | Ключ приватний | Ключ відкритий |
|---|---|---|
| $A$ | $c_A$ | $d_A$ |
| $B$ | $c_B$ | $d_B$ |
| $C$ | $c_C$ | $d_C$ |

.

Покажемо, як абонент $A$ передає повідомлення $m$ абоненту $B$. Припустимо, що $m < p$ (якщо $m < p$, то повідомлення передається зразу, а коли $m > p$, то $m$ ділиться на частини $m = m_1, m_2, \ldots, m_k$, де $m_i < p$ – прості числа $(i = 1, 2, \ldots, k)$ і висилається кожна із частин із власними $c_i$ і $d_i$). Реалізація такого способу виконується таким чином:

**крок 1.** Абонент $A$ вибирає довільним чином число $k,\ 1 \leq k < p-2$, обчислює числа $r = g^k \pmod p,\ e = m \cdot d_B^k \pmod p$ і пересилає пару $(r, e)$ абоненту $B$;

**крок 2.** Абонент $B$, отримавши пару $(r, e)$, обчислює число $m' = e \cdot r^{p-1-c_B} \pmod p$.

**Теорема 74.** *а) $m = m'$;*

*б) особа небажана, знаючи числа $p, g, d_B$ і $e$, не може обчислити $m$ (принаймні за розумний проміжок часу).*

*Доведення.* а) Підставляючи відповідні значення, дістаємо

$$m' = m(g^{c_B})^k (g^k)^{p-1-c_B} \pmod p = m g^{k(p-1)} \pmod p.$$

На підставі теореми Ферма $g^{k(p-1)} \pmod p = 1^k = 1$. Звідси випливає, що $m' = m \cdot g^{k(p-1)} \pmod p = m$.

б) Зловмисник не має змоги обчислити $k$ для виразу $r = g^k \pmod p$, оскільки це функція дискретного логарифма. Отже, він не має змоги обчислити $m$, бо число $m$ множилося на невідоме цій особі число. Крім того, він не має змоги виконати дії абонента $B$, тому що не знає числа $c_B$ (обчислення $c_B$ теж є проблемою

обчислення дискретного логарифма). ■

**Приклад 2.10.3.** Нехай $m = 15$ висилається абонентом $A$ до $B$. Вибираємо $p = 23$ і $g = 5$ (так само як у попередньому прикладі). Нехай абонент $B$ вибрав число $c_B = 13$ і обчислив $d_B = 5^{13} \pmod{23} = 21$.

Абонент $A$ вибрав число $k = 7$ і обчислив

$r = 5^7 \pmod{23} = 17,\ e = 15 \cdot 21^7 \pmod{23} = 15 \cdot 10 \pmod{23} = 12.$

Тепер $A$ висилає до $B$ зашифроване повідомлення (17, 12). $B$ обчислює

$m' = 12 \cdot 17^{23-1-13} \pmod{23} = 12 \cdot 17^9 \pmod{23} = 27 \cdot 7 \pmod{23} = 15.$ ♠

Потрібно зауважити, що в цьому шифрі довжина шифрограми вдвічі перевищує довжину повідомлення, але для передачі повідомлення потрібен лише один сеанс зв'язку.

### Задачі і вправи

1. Знайти всі варіанти вибору параметра $g$ в системі Діффі-Хеллмана для $p = 13$.

2. Знайти секретні ключі $Y_A, Y_B$ і спільний ключ $Z_{AB}$ для системи Діффі-Хеллмана з параметрами:

а) $p = 29,\ g = 5,\ X_A = 5,\ X_B = 7$; б) $p = 19,\ g = 3,\ X_A = 5,\ X_B = 7$;
в) $p = 23,\ g = 7,\ X_A = 3,\ X_B = 4$; г) $p = 17,\ g = 3,\ X_A = 10,\ X_B = 5$;
д) $p = 11,\ g = 2,\ X_A = 4,\ X_B = 8$.

3. Для шифру Шаміра із заданими параметрами $p, c_A, c_B$ знайти параметри, яких бракує, і описати передачу повідомлення $m$ від $A$ до $B$:

а) $p = 19,\ c_A = 5,\ c_B = 7,\ m = 4$; б) $p = 23,\ c_A = 15,\ c_B = 7,\ m = 6$;
в) $p = 17,\ c_A = 11,\ c_B = 5,\ m = 9$; г) $p = 23,\ c_A = 9,\ c_B = 3,\ m = 17$;
д) $p = 17,\ c_A = 3,\ c_B = 13,\ m = 9$.

4. Для шифра Ель-Гамаля із заданими параметрами $p, g, c_B, k$ знайти параметри, яких бракує, і описати передачу повідомлення $m$ до $B$:

а) $p = 19,\ g = 2,\ c_B = 5,\ k = 7,\ m = 5$;
б) $p = 23,\ g = 5,\ c_B = 8,\ k = 10,\ m = 10$;
в) $p = 19,\ g = 2,\ c_B = 11,\ k = 4,\ m = 10$;
г) $p = 23,\ g = 7,\ c_B = 3,\ k = 15,\ m = 5$;
д) $p = 17,\ g = 3,\ c_B = 10,\ k = 5,\ m = 10$.

### Лабораторні роботи

1. Написати і відлагодити набір програм, які реалізують базові алгоритми:
   а) НСД$(m, n)$ (бінарний і розширений бінарний алгоритми);
   б) $a^x \pmod{m}$ (алгоритм піднесення до степеня);
   в) $a^{-1} \pmod{m}$ (алгоритм обчислення оберненого елемента до $a$).
2. Написати програму, яка реалізує систему:
   а) Діффі-Хеллмана для значень $p = 30303,\ g = 2$;
   б) Шаміра для значення $p = 30303$;
   в) Ель-Гамаля для значень $p = 30303,\ g = 2$.

Секретні ключі і необхідні параметри генерувати випадковим чином.

### 2.10.4. Криптоаналіз протоколів обміну ключами

Опишемо два методи криптоаналізу криптографічних протоколів, побудованих на основі функції дискретного логарифма й ефективніших ніж метод послідовних випробовувань ("брутальний метод").

**Метод "крок немовляти, крок гіганта"** був запропонований Шенксом у 1973 р. Наведемо основні кроки цього методу.

**крок 1**. Вибираємо два цілих числа $m$ і $k$ такі, що $m \cdot k > p$, де $p$ – просте число (модуль у конгруентності $y = g^x \pmod p$);

**крок 2**. Обчислюємо два ряди чисел

$$y, gy, g^2y, \ldots, g^{m-1}y \pmod p;$$
$$g^m, g^{2m}, \ldots, g^{km} \pmod p;$$

**крок 3**. Шукаємо числа $i$ і $j$ такі, що $g^{im} = g^j y$.

**Теорема 75.** *Число $x = im - j$ є розв'язком конгруентності $y = g^m \ (mod\ p)$.*

*Доведення* випливає з обчислень за модулем $p$ такої послідовності чисел:

$$g^x = g^{im-j} = \frac{g^{im}}{g^j} = \frac{g^{im}y}{g^j y} = \frac{g^{im}y}{g^{im}} = y.$$

Тепер покажемо, що числа $i$ і $j$ існують. Для цього побудуємо таблицю для чисел типу $x = im - j$:

| $i/j$ | 0 | 1 | 2 | $\cdots$ | $m-1$ |
|---|---|---|---|---|---|
| 1 | $m$ | $m-1$ | $m-2$ | $\cdots$ | 1 |
| 2 | $2m$ | $2m-1$ | $2m-2$ | $\cdots$ | $m+1$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $k$ | $km$ | $km-1$ | $km-2$ | $\cdots$ | $(k-1)m+1$ |

.

Із цієї таблиці видно, що вона включає всі числа від 1 до $km$, а це означає, що вона включає всі числа від 1 до $p$ на підставі нерівності $mk > p$. Звідси випливає, що число $x$, яке задовольняє конгруентності $y = g^x \pmod p$, можна подати у вигляді $x = im - j$ і воно буде завжди знаходитися в цій таблиці. ■

**Приклад 2.10.4.** Нехай маємо рівняння $2^x \pmod{23} = 9$. Виконаємо криптоаналіз описаним методом.

*крок 1.* Вибираємо числа $m, k$: $m = 6,\ k = 4$, оскільки $6 \cdot 4 = 24 > p = 23$;

*крок 2.* Обчислюємо два ряди чисел (крок 2):

(перший ряд) 9, 18, 13, 3, 6, 12;

(другий ряд) 18;

*крок 3.* Оскільки $i = 1$ і $j = 1$ вже відомі, то знаходимо $x = 1 \cdot 6 - 1 = 5$ і $2^5 = 9$ за модулем 23. ♠

Назва цього методу виникла у зв'язку з тим, що у криптографії число $p$ вибирається великим, а тому числа $m$ і $k$ теж будуть великими. У послідовності $y, gy, g^2y, \ldots, g^{m-1}y \pmod{p}$ степінь збільшується на 1 ("крок немовляти"), а в послідовності $g^m, g^{2m}, \ldots\ldots, g^{km} \pmod{p}$ степінь збільшується на $m$ ("крок гіганта"). Справедлива така теорема.

**Теорема 76.** *Часова складність $t$ реалізації описаного методу задовольняє нерівності $t \leq const \cdot \sqrt{p} \log^2 p$.*

**Криптоаналіз на основі обчислення порядку елемента**. Цей спосіб криптоаналізу запропонував Адлеман у 1979 р. Нині цей спосіб (і деякі його варіації) є найкращим для обчислення функції дискретного логарифма.

Нагадаємо, що число $n$ називається $p$-гладким, якщо воно розкладається на прості множники, які менші або дорівнюють $p$. Наприклад, числа $15 = 3 \cdot 5$, $36 = 2^2 \cdot 3^2$, $45 = 5 \cdot 3^2$, $270 = 2 \cdot 3^3 \cdot 5$ є 5-гладкими числами.

**АЛГОРИТМ КРИПТОАНАЛІЗУ**

**крок 1.** Будуємо множину базових чисел $S = \{p_1, p_2, \ldots, p_t\}$, де $p_1, \ldots, p_t$ – перші $t$ простих чисел (про вибір числа $t$ буде сказано далі).

**крок 2.** Для $k = 1, 2, \ldots$ обчислюємо $t + \varepsilon$ ($\varepsilon$ – невелике просте число) $p_t$-гладких чисел вигляду $g^k \pmod{p}$, перевіряючи гладкість шляхом ділення на елементи з множини $S$. Кожне із знайдених $p_t$-гладких чисел записуємо у вигляді добутку базових множників:

$$g^k \pmod{p} = \sum_{i=1}^{t} p_i^{c_i},\ c_i \geq 0 \tag{2.69}$$

(для кожного значення $k$, дістаємо свій набір чисел $c_i$).

**крок 3.** Обчислюємо (за допомогою логарифмів)

$$k = \sum_{i=1}^{t} c_i \cdot \log_g p_i \tag{2.70}$$

для кожного $p_i$-гладкого числа з кроку 2. Будуємо лінійну систему із $t + \varepsilon$ рівнянь типу (2.70) з $t$ невідомими. Невідомими тут виступають величини $\log_g p_i$ і кількість рівнянь на $\varepsilon$ більша від кількості невідомих. Розв'язуючи цю систему, дістаємо значення чисел із множини $S$: $\log_g p_1, \log_g p_2, \ldots, \log_g p_t$.

**крок 4.** Довільним чином вибираємо число $r$, обчислюємо $p_i$-гладке число вигляду $(y \cdot g^t)$:

$$y \cdot g^r \bmod p = \prod_{i=1}^{t} p_i^{c_i},\ c_i \geq 0. \tag{2.71}$$

**крок 5.** Логарифмуючи (2.71), дістаємо остаточний результат

$$x = \log_g y = (\sum_{i=1}^{t} c_i \log_g p_i - r) \pmod{(p-1)}.$$

(величина $r$ віднімається від усієї суми, а не від кожного доданка).

Адлеман показав, що за оптимального значення числа $t$ часова складність алгоритму пропорціональна

$$t_{\text{о.п}} \leq c_1 \cdot 2^{(c_2+o(1))\sqrt[3]{\log p \log\log p}},$$

де $c_1, c_2 > 0$ – сталі величини.

**Вибір числа** $t$. Якщо наосліп вибрати число з нескінченної множини цілих чисел, то з імовірністю $1/2$ воно ділиться на 2, з імовірністю $1/3$ воно ділиться на 3, з імовірністю $1/5$ воно ділиться на 5 і т. д. Тому можна очікувати, що у проміжку від 1 до $p-1$ існує досить багато чисел, у розкладі яких на прості множники знаходяться тільки невеликі прості числа із введеної вище множини $S$. Саме такі числа відшукуються на кроках 2 і 4 наведеного алгоритму. Чим більше число $t$, тобто кількість простих множників у множині $S$, тим менше невдач у пошуку гладких чисел відбуватиметься на кроках 2 і 4. А це означає, що ці кроки будуть виконуватися швидше. Але при великих значеннях $t$ різко зростає складність виконання кроку 3, коли необхідно розв'язувати систему з $t+\varepsilon$ рівнянь. Пошук оптимального значення $t$, яке дає мінімальний загальний час обчислень, як правило, виконується за допомогою чисельних методів. Параметр $\varepsilon$ приймається рівним невеликому цілому числу для того, щоб збільшити ймовірність існування розв'язку системи рівнянь

на кроці 3. Вважається, що при великому значенні числа $p$ значення $\varepsilon$ має порядок 10 і це гарантує існування єдиного розв'язку системи з високою ймовірністю [31]. Якщо система має нескінченно багато розв'язків, то необхідно повернутися до кроку 2 і використати інші значення $k$.

**Приклад 2.10.5.** Знайти розв'язок $x$ із рівняння $37 = 10^x \pmod{47}$.

*Розв'язання.* Маємо $y = 37,\ a = 10,\ p = 47$. Візьмемо множину базових чисел $S = \{2, 3, 5\}$, $t = 3$ і $\varepsilon = 1$, тобто, будуємо систему чотирьох рівнянь із трьома невідомими.

Нехай логарифми чисел із $S$ дорівнюють $u_1, u_2, u_3$ відповідно, наприклад $u_3 = \lg 5 \pmod{47}$. А це означає, що виконано перший крок алгоритму.

*крок 2.* Шукаємо чотири 5-гладких чисел:

$10^1 \pmod{47} = 10 = 2 \cdot 5$, $10^2 \pmod{47} = 6 = 2 \cdot 3$, $10^3 \pmod{47} = 13 = 13$,

$10^4 \pmod{47} = 36 = 2 \cdot 2 \cdot 3 \cdot 3$, $10^5 \pmod{47} = 31$, $10^6 \pmod{47} = 28 = 2 \cdot 2 \cdot 7$,

$10^7 \pmod{47} = 45 = 3 \cdot 3 \cdot 5$.

Отримуємо чотири 5-гладких чисел, що відповідають степеням 1, 2, 4 і 7.

*крок 3.* Будуємо систему рівнянь:

$$S = \begin{cases} 1 = u_1 + u_3 \\ 2 = u_1 + u_2 \\ 4 = 2u_1 + 2u_2 \\ 7 = 2u_2 + u_3 \end{cases}$$

У системі друге і третє рівняння лінійно залежні, а це означає, що пошук був виконаний не даремно.

Розв'язуємо отриману систему і знаходимо: $u_2 = 18, u_3 = 17, u_1 = 30$.

*крок 4.* Нехай $k = 3$. Тоді

$37 \cdot 10^3 \pmod{47} = 37 \cdot 13 \pmod{47} = 11$,

$37 \cdot 10^4 \pmod{47} = 37 \cdot 36 \pmod{47} = 16 = 2 \cdot 2 \cdot 2 \cdot 2$.

*крок 5.* Обчислюємо логарифм

$\lg 37 = 4 \lg 2 - 4 = (4 \cdot 30 - 4) \pmod{46} = 24$.

А це і є шуканий розв'язок початкового рівняння, оскільки $10^{24} \pmod{47} = 37$. ♠

**Задачі і вправи**

1. Методом "крок гіганта, крок немовляти" розв'язати такі порівняння:

$2^x = 21 \pmod{29}$; $3^x = 25 \pmod{31}$; $2^x = 12 \pmod{31}$;
$6^x = 21 \pmod{37}$; $3^x = 11 \pmod{43}$; $8^x = 25 \pmod{19}$.

2. Методом обчислення порядку елемента розв'язати такі рівняння:

$2^x = 24 \pmod{53}$; $2^x = 13 \pmod{59}$; $2^x = 45 \pmod{61}$;
$2^x = 45 \pmod{67}$; $3^x = 22 \pmod{67}$; $7^x = 41 \pmod{71}$.

**Лабораторна робота**

1. Виконати програмну реалізацію методів "крок гіганта, крок немовляти" і обчислення порядку елемента та розв'язати за допомогою цієї реалізації такі рівняння:

$2^x = 24322 \pmod{30203}$; $2^x = 21740 \pmod{30323}$; $2^x = 28620 \pmod{30539}$;
$2^x = 16190 \pmod{30803}$; $5^x = 30994 \pmod{31607}$; $3^x = 13287 \pmod{30323}$.

## 2.11. Генератори випадкових чисел

Як сказано у протоколі 1 з пункту 1.1.4, Аліса вибирає велике випадково згенероване ціле число і повідомляє його Бобу. Яким чином Аліса вибирає це число не говорилося. У криптографічних системах і їхніх протоколах виникає нагальна потреба в генерації великих випадкових чисел і слів, які використовуватимуться як ключі. Тому задача генерування послідовностей випадкових чисел представляє великий інтерес для розробників криптосистем. Більше того, з розвитком криптографічних методів стало зрозумілим, що багато фундаментальних проблем цієї науки тісно пов'язані з генерацією випадкових чисел. Що ж це за випадкові числа і їхні послідовності?

**Випадковою послідовністю чисел у криптографії** називається послідовність символів (найчастіше символами є нулі й одиниці), в якій поява символів рівноймовірна і незалежна.

*Виникає питання:* як генерувати такі числа і їхні послідовності?

Найпопулярнішим способом отримання послідовності випадкових чисел є підкидання монети, але цей спосіб у криптографії не прийнятний, оскільки тут вимагається генерувати великі послідовності випадкових чисел із високою швидкістю. Для такої мети використовують інші фізичні процеси, які мать високу продуктивність і відносно легко реалізуються на комп'ютерних системах. Наприклад, такими фізичними генераторами можуть служити шуми в електромережах та їхніх елементах, лічильники фізичних частинок тощо. У цьому випадку однією з основних задач, які виникають при використанні таких "фізичних" генераторів, є перетворення генерованих ними послідовностей у випадкові числа.

Одним із популярних способів отримання випадкових чисел, не пов'язаним із фізичними процесами, є спосіб, який ґрунтується на обчисленнях. Отримані в такий спосіб послідовності чисел називають **псевдовипадковими числами**, а їх генератори – **псевдогенераторами випадкових чисел**. Ці числа називають псевдовипадковими тому, що вони відрізняються від істинно випадкових чисел. Для істинно випадкових чисел ми не у змозі передбачити значення наступного символу, а в у випадку обчислень така мо-

жливість існує. Саме через указану можливість ці числа називають псевдовипадковими.

**Лінійні конгруентні генератори.** Найпростішими генераторами псевдовипадкових чисел є **лінійні конгруентні генератори (ЛКГ)**. Ці генератори працюють за такою формулою:

$$X_n = (a \cdot X_{n-1} + b) \pmod{m}, \tag{2.72}$$

в якій $X_n$ означає $n$-те число в послідовності, а $X_{n-1}$ – попереднє число в цій послідовності. Величини чисел $a, b$ і $m$ є сталими. Число $a$ називається **множником**, число $b$ – **часткою**, а число $m$ – **модулем**. Початкове значення $X_0$ називається **ключем** або **зерном** ЛКГ. Цей генератор дає повторення чисел не частіше $n \leq m$, де число $n$ називається **періодом** послідовності. Якщо числа $a, b$ і $m$ спеціальним чином підібрані, то такий генератор буде **генератором максимального періоду** і цей період дорівнює $m$. Для цього достатньо взаємної простоти чисел $b$ і $m$, кратності $a - 1$ простому дільнику $m$ та коли $a - 1$ кратне 4 то $m$ кратне 4 [11].

**Приклад 2.11.1**. Нехай у формулі (2.72) $a = 5, b = 12, m = 23, X_0 = 4$ і знайдемо декілька елементів послідовності:

$X_1 = (4 \cdot 5 + 12) \pmod{23} = 9$, $X_2 = (9 \cdot 5 + 12) \pmod{23} = 11$,
$X_3 = (11 \cdot 5 + 12) \pmod{23} = 21$, $X_4 = (21 \cdot 5 + 12) \pmod{23} = 2$,
$X_5 = (2 \cdot 5 + 12) \pmod{23} = 22$, $X_6 = (22 \cdot 5 + 12) \pmod{23} = 7$,
$X_7 = (7 \cdot 5 + 12) \pmod{23} = 1$, $X_8 = (1 \cdot 5 + 12) \pmod{23} = 17$,
$X_9 = (17 \cdot 5 + 12) \pmod{23} = 5$, $X_{10} = (5 \cdot 5 + 12) \pmod{23} = 14$,
$X_{11} = (14 \cdot 5 + 12) \pmod{23} = 13$, $X_{12} = (13 \cdot 5 + 12) \pmod{23} = 8$,
$X_{13} = (8 \cdot 5 + 12) \pmod{23} = 6$, $X_{14} = (6 \cdot 5 + 12) \pmod{23} = 19$,
$X_{15} = (19 \cdot 5 + 12) \pmod{23} = 15$, $X_{16} = (15 \cdot 5 + 12) \pmod{23} = 18$.

Отримана послідовність виглядає неначе й справді випадкова. ♠

**Перевагою ЛКГ** є висока швидкість реалізації, а **недоліком** є те, що вихідні послідовності передбачувані. Дійсно, відомі прості алгоритми, за допомогою яких можна повністю віднайти параметри ЛКГ всього за декількома елементами послідовності, що генерується.

Для використання генератора випадкових чисел у криптографії він повинен задовольняти таким основним вимогам:

*1) період послідовності має бути великим;*

*2) послідовність, яка генерується, має бути такою, що майже не відрізняється від дійсно випадкової послідовності, зокрема,*

*обчислення числа $X_{n+1}$ за відомими попередніми елементами послідовності без знання ключа повинно бути важкою, практично нерозв'язуваною задачею.*

Генератори поліноміальні є узагальненням ЛКГ і мають вигляд

$$X_n = (a \cdot X_{n-1}^2 + b \cdot X_{n-1} + c) \pmod{m} \text{ (другого порядку)},$$

або $X_n = (a \cdot X_{n-1}^3 + b \cdot X_{n-1}^2 + c \cdot X_{n-1} + d) \pmod{m}$ (третього порядку) і так далі.

У лінійних і поліноміальних генераторах максимальний період послідовності (2.72) не перевищує величини модуля $m$. Вибір чисел $a_i, b, m$, за яких досягається максимальне значення періоду, відбувається на основі узагальнення правил вибору для генератора. Наприклад, для генератора другого порядку $x_{n+1} = (a_2 x_n^2 + a_1 x_n + {} + a_0) \pmod{m}$ ці правила є такими:

а) НСД$(a_0, m) = 1$;

б) числа $a_1$ і $a_2$ кратні кожному простому дільнику $p$ модуля $m$;

в) $a_2 \equiv (a_1 - 1) \pmod{2}$, якщо $m$ парне;

г) $a_2$ – парне і $a_2 \equiv (a_1 - 1) \pmod{4}$, якщо $m$ кратне 4;

д) $a_2 \not\equiv 3a_0 \pmod{9}$, якщо $m$ кратне 9.

Наступні спроби покращити роботу генератора послідовності псевдовипадкових чисел зводилися до того, щоб зробити залежним $n$-й член послідовності від декількох попередніх членів. Такого типу генератор був реалізований у адитивному генераторі Фібоначчі. Генерація таким методом виконувалася за формулою $x_{n+1} = (x_n + {} + x_{n-1}) \pmod{m}$, або за формулою $x_{n+1} = (x_{n-p} + x_{n-q}) \pmod{m}$, де в обох формулах модуль $m$ парне число.

Послідовність, згенерована за другою з наведених формул, має максимальний період $2^{\log(m-1)}(2^q - 1)$ у випадку незвідності полінома $x^p + x^q + 1$ в полі $\mathcal{F}_2$.

Наступним поліпшенням адитивного генератора Фібоначчі був мультиплікативний генератор Фібоначчі, де генерація виконувалася за формулою $x_n = (x_{n-p} \cdot x_{n-q}) \pmod{m}$, якщо $m$ – парне, а $x_0, x_1, \ldots, x_{q-1}$ – цілі непарні числа такі, що $x_i \equiv 1 \pmod{4}$, $i = 0, 1, \ldots, q-1$. Максимальний період послідовності, згенерованої таким генератором, дорівнює $2^{\log(m-3)}(2^q - 1)$ за умови незвідності

полінома $x^p + x^q + 1$ в полі $\mathcal{F}_2$.

**Регістр зсуву зі зворотним зв'язком** – це електронна схема, яка генерує числову послідовність, задану рекурентним рівнянням. Така схема складається з двох частин: *регістра зсуву* і *функції зворотного зв'язку* (рис. 2.11.1).

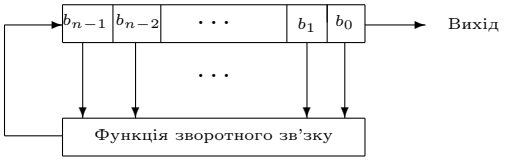



Рис. 2.11.1. Схема зі зворотним зв'язком

Регістр зсуву – це послідовність бітів $b_{n-1}b_{n-2}\dots b_2b_1b_0$, кількість яких називається *довжиною регістра зсуву*. Якщо довжина регістра дорівнює $n$ бітам, то регістр називають $n$-бітовим регістром зсуву.

Під час кожного виходу останнього біта з послідовності решта бітів регістра пересувається на одну позицію праворуч. Значення нового старшого біта обчислюється як функція від решти деяких із бітів регістра. Отримана в такий спосіб послідовність бітів періодична, її період називається *періодом регістра зсуву*.

Найпростішим типом регістра зсуву є *регістр зсуву зі зворотним лінійним зв'язком* (LFSR – Linear Feedback Shift Register). Функція зворотного зв'язку є операцією додавання за модулем 2 (операція XOR) над деякими бітами регістра. Сукупність цих бітів називається *точками знімання*.

**Приклад роботи 4-бітового регістра** зсуву зі зворотним лінійним зв'язком із точками знімання першого та четвертого бітів (рис. 2.11.2).

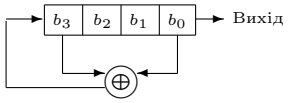



Рис. 2.11.2. 4-бітова схема

Якщо початкове значення регістра дорівнює 1111, то далі схема буде генерувати таку послідовність бітів.

На першому кроці, після застосування операції $\oplus$ дістаємо $b'_3 = b_3 \oplus b_0 = 1 \oplus 1 = 0$, виконується зсув $b_3 \to b_2,\ b_2 \to b_1,\ b_1 \to b_0$. У результаті регістр набуває значення 0111, а на виході буде згенероване значення $b_0 = 1$. Після цього операції повторюються

і результати будуть такими:

| $n$ | Регістр | Вихід | $n$ | Регістр | Вихід |
|---|---|---|---|---|---|
| 0) | 1111 | | 8) | 1001 | 1 |
| 1) | 0111 | 1 | 9) | 0100 | 1 |
| 2) | 1011 | 1 | 10) | 0010 | 0 |
| 3) | 0101 | 1 | 11) | 0001 | 0 |
| 4) | 1010 | 1 | 12) | 1000 | 1 |
| 5) | 1101 | 0 | 13) | 1100 | 0 |
| 6) | 0110 | 1 | 14) | 1110 | 0 |
| 7) | 0011 | 0 | 15) | 1111 | 0 |

.

Таким чином, на виході генерується така послідовність бітів: 111101011001000. ♠

Кількість різних станів регістра $n$-бітового LFSR дорівнює $2^n - 1$ ( а не $2^n$, оскільки коли початкове значення регістра заповнене нулями, то схема генеруватиме нескінченну послідовність з одних нулів, що не має сенсу).

Пройти всі можливі значення регістр може лише за певної послідовності точок знімання.

Прикладами генераторів псевдовипадкових чисел, які використовуються на практиці, є генератори OFB і CTR блокових шифрів.

Нагадаємо, що блоковим шифром називається криптографічна система, в якій початковий відкритий текст розбивається на частини, які називаються **блоками**, а потім кожний блок шифрується окремо (за допомогою одного і того ж ключа, або різних ключів) і висилається отримувачу. Розглянемо один із цих генераторів.

**Генератор OFB.** Назва цього генератора походить від англійських слів Output FeedBack, яке перекладається як обернений зв'язок за виходом. У цьому генераторі на основі певного алгоритму $E_K$ із секретним ключем $K$ і деякого початкового вектора $Y_0$ формується псевдовипадкова послідовність $r$-бітових чисел $z_1, z_2, \ldots, z_k$, яка використовується у формулах

$$y_i = x_i \oplus z_i, \qquad x_i = y_i \oplus z_i \tag{2.73}$$

для шифрування і розшифрування відповідно. Нехай розмір блока дорівнює $n$.

Псевдовипадкова послідовність обчислюється за схемою

$$Y_i = E_K(Y_{i-1}), \quad z_i = r \text{ старших бітів } Y_i,\ 1 \le i \le k$$

(тут $1 \le r \le n$ – параметр методу).

Використовуючи стійкий блоковий шифр, можна отримати стійкий криптогенератор псевдовипадкових чисел, який задовольняє вищенаведені вимоги. А саме, середня довжина періоду псевдовипадкової послідовності (при випадково вибраних $K$ і $Y_0$) складає приблизно $r2^{n-1}$ бітів. Крім того, псевдовипадкова послідовність "непередбачувана" для хакера, оскільки можливість передбачення (обчислення) $z_{i+1}$ на основі $z_1, \ldots, z_i$ означала б нестійкість шифру.

Звернемо увагу на одну особливість, характерну для таких шифрів. Для шифрування кожного окремого повідомлення необхідно використовувати різні ключі $K$ і/або $Y_0$. У протилежному випадку декілька повідомлень будуть шифруватися за допомогою одних і тих самих послідовностей $z$, і такий шифр може бути зламаний. Дійсно, нехай два повідомлення $u_1, u_2, \ldots, u_k$ і $v_1, v_2, \ldots, v_k$ зашифровані за допомогою однієї і тієї ж послідовності $z$. Тоді шифрограми будуть мати вигляд: $u_1 \oplus z_1, u_2 \oplus z_2, \ldots, u_k \oplus z_k$ і $v_1 \oplus z_1, v_2 \oplus z_2, \ldots, v_k \oplus z_k$. Додамо обидві шифрограми і, на підставі рівності $z_i \oplus z_i = 0$, дістанемо послідовність $u_1 \oplus v_1, u_2 \oplus v_2, \ldots, u_k \oplus v_k$. У результаті отримуємо аналог так званого "шифру з динамічним ключем", коли один текст шифрується за допомогою другого тексту, взятого з певного місця деякої книги. Відомо, що такий шифр не є стійким, хоча він широко використовувався в епоху "донаукової" криптографії [31]. Статистичний аналіз, який ґрунтується на надлишковості текстів, дає можливість у більшості випадків досить точно відновити обидва повідомлення.

Розшифрування повідомлень для цього генератора блокового шифру може виконуватися тільки з початку, оскільки неможливо отримати довільний елемент послідовності $z$, не обчисливши попередні. Перевагою OFB є те, що послідовність $z$ може бути сформована завчасно для того, щоб швидко шифрувати або розшифрувати повідомлення на підставі формул (2.73) в момент їхнього надходження. Це надзвичайно актуально для систем реального часу.

**Контрольні запитання**

1. Навести означення псевдовипадкового числа, послідовності.
2. Які основні властивості повинен мати генератор випадкових чисел?

3. Сформулювати основні властивості генераторів ЛКГ, LFSR, OFB.

**Задачі і вправи**

1. Нехай а) $a = 7, b = 15, m = 29, X_0 = 3$; б) $a = 9, b = 11, m = 19, X_0 = 7$;
в) $a = 13, b = 22, m = 31, X_0 = 5$; г) $a = 5, b = 17, m = 17, X_0 = 9$.
Знайти 15 елементів послідовності, яка згенерується ЛКГ.

2. Користуючись генератором другого порядку та LFSR, знайти 10 членів послідовності, якщо

а) $a = 7, b = 15, m = 29, c = 3, X_0 = 3$; б) $a = 9, b = 11, m = 19, c = 7, X_0 = 5$.

# Розділ 3

# КРИПТОГРАФІЧНІ СИСТЕМИ

Розглянемо детальніше об'єкти, які беруть участь у обміні криптографічними повідомленнями. Під час шифрування ВТ перетворюється таким чином, щоб його зміст не був зрозумілим для тих осіб, які не знають секретного ключа. Нагадаємо, що текстом називається слово в деякому скінченному алфавіті $X = \{x_1, x_2, \ldots, x_m\}$. Як зазначалося, елементами алфавіту можуть бути літери, літери та цифри, цифри та знаки пунктуації і взагалі скінченний набір будь-яких елементів.

Відкритий текст шифрується абонентом $A$ і каналами зв'язку передається абоненту $B$, який за допомогою ключа розшифровує шифрограму, отриману від абонента $A$ (рис. 3.1.1).


Рис. 3.1.1. Схема шифрування ВТ та розшифрування ШТ

Шифрування ВТ і розшифрування ШТ ґрунтується на функціях шифрування $f$ і розшифрування $g = f^{-1}$ та алгоритмах їхньої реалізації з використанням відповідних ключів. Ці функції повинні

задовольняти умови:

$$(\forall p \in F(X)(\forall k \in K)(\exists k_1 \in K_1) f_k(p) = q, \quad g_{k_1}(q) = p, \tag{3.1}$$

де $K, K_1$ – простори ключів шифрування і розшифрування відповідно. Об'єднуючи обидві рівності, дістаємо

$$g_{k_1}(f_k(p)) = p.$$

Зауважимо, що ключі $k$ і $k_1$ можуть збігатися. Як випливає з попередніх розділів, функція $f$ повинна бути односторонньою. Нехай $E_k$ означає алгоритм реалізації функції $f$, а $D_{k_1}$ – алгоритм реалізації функції $g = f^{-1}$. Тоді

$$(\forall p \in F(X)(\forall k \in K)(\exists k_1 \in K_1) E_k(p) = q \text{ і } D_{k_1}(q) = p.$$

Обмін ключами між абонентами може відбуватися різними шляхами. За наявності закритого каналу зв'язку (рис. 3.1.2) обмін, як правило, виконується за допомогою такого каналу. Якщо ж закритого каналу немає, то використовують загальнодоступні канали зв'язку за допомогою шифрування (рис. 3.1.3).

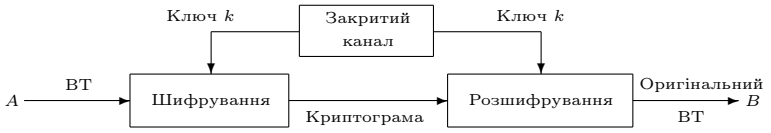



Рис. 3.1.2. Шифрування і розшифрування ВТ з одним ключем

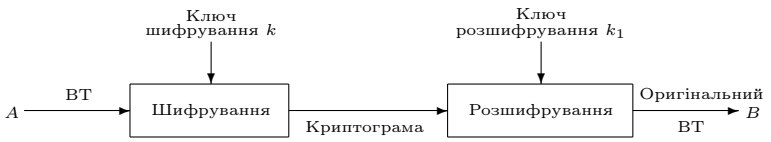



Рис. 3.1.3. Шифрування і розшифрування ВТ з двома різними ключами

Безпека алгоритмів, які використовують в останній схемі, опирається на ключі і не опирається на деталі алгоритму. Це означає, що алгоритм може бути опублікований і аналізований, а, отже, такий алгоритм може вживатися масово. Небажаний користувач може знати алгоритм, але якщо він не знає таємного ключа, то не зможе прочитати відкритий текст.

Таким чином ми приходимо до такого означення криптографічної системи.

**Означення 74.** *Криптографічна система $\mathcal{S}$ складається з таких компонентів.*

• *Множина початкових повідомлень $M \subset F(X)$: множина слів у деякому алфавіті $X$.*

• *Множина зашифрованих текстів $C$: множина криптограм, тобто множина слів або в тому ж алфавіті $X$, або в іншому алфавіті.*

• *Простір ключів $K$ – множина можливих значень ключів шифрування і розшифрування (простір ключів розшифрування може не збігатися із $K$).*

• *Ефективний алгоритм генерації ключів $G : N \to K \times K_1$.*

• *Ефективний алгоритм шифрування $E_k : M \times K \to C$.*

• *Ефективний алгоритм розшифрування $D_{k_1} : C \times K_1 \to M$.*

*Для цілого числа $n$ результатом алгоритму $G(n)$ є пара ключів $(k, k_1) \in K \times K_1$, які мають довжину $l$.*

*Для ключа $k \in K$ і повідомлення $m \in M$ криптограма будується алгоритмом $E_k$, тобто $c = E_k(m)$, а ВТ – алгоритмом $D_{k_1}$, тобто $m = D_{k_1}(c)$.*

*Необхідно, щоб для всіх повідомлень $m \in M$ і всіх ключів $k \in K$ існував ключ $k_1 \in K_1$ такий, що*

$$D_{k_1}(E_k(m)) = m. \tag{3.2}$$

Отже, **шифром** називається відображення $f : M \times K \to C$ таке, що $(\forall k \in K)(\forall m \in M)(\exists k_1 \in K_1)(D_{k_1}(E_k(m)) = m$, де $E_k$ – алгоритм реалізації $f$, а $D_{k_1}$ – алгоритм реалізації оберненого відображення $f^{-1}$. Отже, $f$ – бієкція і це дає можливість зашифрувати і розшифрувати будь-який ВТ за допомогою ключа $k$.

## 3.1. Різновиди криптосистем із ключами

**Симетричні криптосистеми** мають ключ до шифрування і розшифрування один і той самий. Такі системи називають теж *ши-*

*фрами з ключем таємним* або *шифрами з єдиним ключем*. Вони вимагають узгодження ключа між абонентами і лише після цього вони можуть пересилати повідомлення. Ключ такий повинен утримуватися в таємниці і безпека такого шифру залежить від безпеки ключа. Втрата ключа означає, що в такій системі кожний зможе шифрувати і розшифрувати повідомлення.

Існує декілька різновидів симетричних систем:

*1) системи підстановки;* *2) системи перестановки;*
*3) системи потокові;* *4) системи блокові.*

Для алгоритмів потокових (потокових шифрів) одиницею перетворення виступає один біт (інколи один байт).

Для алгоритмів блокових (блокових шифрів) одиницею перетворення виступає група бітів, яка називається **блоком**.

Алгоритми, реалізовані на комп'ютерах, найчастіше оперують на блоках у 64 біти, що є достатнім для того, щоб ускладнити криптоаналіз, і недостатнім для того, щоб бути ефективними. До ери комп'ютерів алгоритми шифрування працювали на відкритих текстах, виконуючи операції знак по знаку і тому про такі алгоритми можна говорити як про алгоритми потокові.

**Асиметричні криптосистеми** або **криптосистеми з ключем відкритим**. Такі системи називаються **асиметричними системами**. Вони проєктуються так, щоб ключ до шифрування був один, а для розшифрування інший. Більше того, ключ до розшифрування не можна визначити з ключа до шифрування (принаймні в деякому розумному проміжку часу). Такі шифри називаються **шифрами з ключем відкритим**, оскільки ключ до шифрування можна опублікувати: довільна особа може скористатися цим ключем для шифрування повідомлення, але розшифрувати таке повідомлення може тільки та особа, яка має ключ до розшифрування. У такій системі ключ до шифрування називається **ключем відкритим**, а ключ до розшифрування – **ключем таємним** (інколи цей ключ називають **приватним**).

### 3.1.1. Характеристика криптосистем

Розглянемо коротко переваги і недоліки обох типів криптосистем.

**а) Переваги симетричних криптосистем**.

1. Система може бути спроєктована так, що її пропускна здатність для даних буде високою. Деякі апаратні реалізації можуть досягати криптографічної потужності сотні мегабайт у секунду, в той час як програмні реалізації досягають лише мегабайтної потужності в секунду.

2. Ключі в таких системах відносно короткі.

3. Цифрові ключі можуть використовуватися для конструювання різноманітних криптографічних механізмів, таких як генератори псевдовипадкових чисел, хеш-функцій тощо.

4. Системи можуть утворювати композиції простих систем із метою побудови складніших систем.

5. Системи цього типу мають велику історію, а це відповідні знання для побудови стійких комп'ютерних систем.

**б) Недоліки симетричних криптосистем**.

1. У двосторонній комунікації ключ повинен триматися в секреті обома сторонами.

2. У великих робочих системах існує велика кількість пар ключів, а це ускладнює управління.

3. У двосторонній комунікації існуюча криптографічна практика вимагає частої зміни ключів шифрування та розшифрування, можливо, при кожній сесії зв'язку.

**в) Переваги асиметричних криптосистем**.

1. Лише один ключ є секретним (хоч автентифікація відкритих ключів має бути гарантована).

2. Адміністрування ключами в робочій мережі потребує лише функціональної довіри до протоколів трансмісії, а не безумовної довіри. Крім цього, адміністрування може реалізовуватися в режимі "$off\text{-}line$", а не в реальному часі.

3. Залежно від способу використання приватні й відкриті ключі можуть не змінюватися тривалий час.

4. Багато систем мають ефективну реалізацію.

5. У багатьох системах використовують ключі меншої довжини, ніж у симетричних криптосистемах.

**г) Недоліки асиметричних криптосистем**.

1. Швидкість комунікації у більшості популярних криптосистем повільніша, ніж швидкість комунікації у симетричних криптосистем.

2. Розміри ключів, як правило, набагато більші, ніж у симетричних криптосистемах.

3. Немає жодної системи, для якої доведена її повна секретність. Більшість криптографічних систем (як симетричних, так і несиметричних) ґрунтуються на важких обчислювальних проблемах, множина яких невелика.

4. Асиметрична криптографія має невелику історію, яка має початок у 70-х рр. минулого століття.

### 3.1.2. Небезпека для криптосистем

Основною небезпекою інформаційних систем є особи, яких називають **зловмисниками** або **хакерами**. Але крім них існують і інші, не менш небезпечні причини повного або часткового знищення таких систем. Ось деякі з них:

**Причини людські**. У кожній організації існує група працівників, які відповідають за експлуатацію інформаційних систем. Зокрема, до них належать адміністратори і розробники системи. Ці особи володіють усією інформацією відносно системи: інформацією про побудову, управління й експлуатацію. Якщо організація роботи в такому закладі не передбачує документування ключової інформації, то може статися так, що в момент втрати співробітника система може стати некерованою. Якщо адміністратор системи не документує паролів доступу до виділених рахунків, то за його відсутності система не може бути модифікована. Відомі також випадки, коли деякі співробітники свідомо нищили інформаційну систему своєї фірми. Отже, існують загрози, які походять зі сторони власних співробітників. Згідно зі статистикою понад 50 % ламань інформаційних систем виконується власними співробітниками фірми.

**Неправильна експлуатація приладдя**. Згідно із статистикою, найчастішою причиною аварії комп'ютерних приладів є перепади напруги в електромережах. Кількість аварій такого типу

доходить до 25 % всіх аварій систем. Причинами аварії можуть стати атмосферні явища, вихід із ладу індуктивних електромашин, неправильна експлуатація електрообладнання тощо. Для запобігання цим аваріям необхідно забезпечувати системи засобами згладжування стрибків напруги та їхніх перепадів. Знищення системи може бути також результатом повені, не врахування кліматичних умов тощо. До цієї категорії належать також кражі комп'ютерів і їх окремих деталей.

**Причини, залежні від виробників**. Інформаційна система деякої організації побудована з великої кількості елементів, до яких, зокрема, належать засоби та програмні продукти. Жоден із виробників як програмного, так і технічного забезпечення не може дати гарантії повної справності програмних і технічних засобів. Програмні засоби з помилками можуть спричиняти втрату даних, а в окремих випадках, навіть, аварію всієї системи в цілому. Пошкодженням може підлягати також техніка незалежно від експлуатації, якщо не виконуються умови виробника. Окремим випадком небезпеки, яка походить від виробника, є його банкрутство. У таких випадках користувач не матиме доступу до запасних частин, первинних кодів програм тощо.

Зі всього сказаного випливає, що в інформаційній системі **найважливішим є охорона множини даних**. У більшості випадків безпека даних забезпечується за допомогою засобів організаційно-технічних.

## 3.2. Симетричні криптографічні системи

Розглянемо основні криптографічні алгоритми симетричних систем шифрування. Існує два основних типи симетричних криптосистем: **блокові** і **потокові**.

Блокові шифри працюють із блоками ВТ і ШТ, як правило, довжиною 64 біти або більшою довжиною. Потокові шифри працюють із бітовими або байтовими потоками ВТ і ШТ, як правило з потоками 32-бітових слів. Блоковий шифр, використовуючи один і той самий ключ, під час шифрування перетворює один і той самий блок

ВТ в один і той самий блок ШТ. Потоковий шифр перетворює один і той самий біт чи байт на різні біти чи байти ШТ.

У криптографічних застосуваннях блоковий шифр повинен задовольняти певні умови, які залежать від ситуації, в якій цей шифр використовується. У більшості випадків досить вимагати, щоб шифр був стійким до атаки на основі вибраного тексту. Неформально це можна сформулювати так: *блоковий шифр вважається стійким (в разі атаки за вибраним текстом), якщо для нього невідомі жодні алгоритми злому, суттєво ефективніші, ніж прямий перебір ключів.*

До класу блокових шифрів належать шифри *підстановки* і *перестановки*. Ці шифри можуть шифрувати ВТ у одному (моноалфавітні шифри), або в декількох алфавітах (мультиалфавітні шифри).

### 3.2.1. Шифри підстановки і перестановки

У такого типу алгоритмах шифрування $E_k(m)$ є функцією підстановки, яка замінює кожне повідомлення $m \in M$ відповідною криптограмою $c \in C$. Алгоритм розшифрування $D_k(c)$ є оберненою підстановкою. Як відомо множина підстановок утворює групу і для кожної підстановки існує обернена підстановка. Отже, підстановка задається як відображення $\pi : M \to C$, а обернена підстановка як відображення $\pi^{-1} : C \to M$.

Нехай $M = C = X_{26}$, а літери англійського алфавіту інтерпретуються таким чином: $A = 0, B = 1, \ldots, Z = 25$. Алгоритм шифрування $E_k(m)$ визначаємо як підстановку на групі $G_{26}$.

$$\begin{pmatrix} a & b & c & d & e & f & g & h & i & j & k & l & m & n & o & p & q & r & s & t & u & v & w & x & y & z \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 \\ 21 & 12 & 25 & 17 & 24 & 23 & 19 & 15 & 22 & 13 & 18 & 3 & 9 & 5 & 10 & 2 & 8 & 16 & 11 & 14 & 7 & 1 & 4 & 20 & 0 & 6 \end{pmatrix}.$$

Тоді відповідний алгоритм розшифрування $D_k(c)$ набуває вигляду

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 \\ 24 & 21 & 15 & 11 & 22 & 13 & 25 & 20 & 16 & 12 & 14 & 18 & 1 & 9 & 19 & 7 & 17 & 3 & 10 & 6 & 23 & 0 & 8 & 5 & 4 & 2 \end{pmatrix}.$$

Метод підстановки стає практичним, якщо існує великий простір для вибору ключів. Наприклад, американський стандартний

алгоритм DES використовує 56-бітовий ключ, що дає простір вибору $2^{56}$, або більше $7 \cdot 10^{16}$ можливих ключів.

Істотним є знання мови відкритого тексту, оскільки коли не відома мова відкритого тексту, то можна не розпізнати результати розшифрування. Більше того, шифрограма може бути яким-небудь способом скорочена або спресована, а це додаткові перешкоди на шляху розшифрування. Якщо спресуємо файл, а потім його зашифруємо простим шифром, то текст явний може бути не розпізнаний цим методом.

**Шифри моноалфавітні**. До криптографічних систем, які ґрунтуються на техніці підстановок, як було сказано, належить шифр Цезаря. Цей шифр із його 25 ключами не можна вважати безпечним. Для ускладнення криптоаналізу шифрів підстановки можна застосувати інші арифметичні операції, зокрема операцію множення. Прикладом такого шифру є моноалфавітний шифр, в якому підстановка визначається за допомогою такого виразу:

$$f(a) = a \cdot k_1 \pmod{n}, \tag{3.3}$$

де $a$ – порядковий номер літери в алфавіті, а $k_1$ – коефіцієнт зсуву. Приклад 3.2.1 ілюструє використання цього шифру. У цьому методі шифрування потужність алфавіту (число) $n$ і ключ $k_1$ повинні бути взаємно простими числами. Інакше символи відкритого тексту і в деяких позиціях символи зашифрованого тексту можуть бути однаковими.

**Приклад 3.2.1.** Скористаємося вищенаведеною підстановкою на групі $G_{26}$ для моноалфавітного кодування з ключем (коефіцієнтом зсуву) $k_1 = 9$ і модулем $n = 26$ та зашифруємо за її допомогою слово **"student"**.

У цій підстановці літері $s$ відповідає літера $G$ оскільки $9 \cdot 18 \pmod{26} = 6$, а 6 – це номер літери $G$. Виконуючи такі обчислення, дістаємо ШТ:

ВТ: **student**

ШТ: **GPYBKNP**. ♠

Подальше ускладнення процедури заміни досягається шляхом перетворення на основі виразу (так званий **афінний шифр**):

$$f(a) = (a \cdot k_1 + k) \pmod{n}. \tag{3.4}$$

У цьому виразі теж повинна виконуватися умова НСД$(k_1, n) = 1$. Наведений вираз може також ускладнюватися і застосовуватися до побудови так званих поліноміальних алгоритмів, в яких перетворення символів реалізується на підставі використання поліномів, степінь яких більший одиниці. Розглянуті шифри Цезаря й афінний мають відповідно нульовий степінь і перший степінь.

Шифрування за формулами (3.3) і (3.4) можна узагальнити, якщо в цих формулах замінити коефіцієнт $a$ матрицею

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

розмірності $2 \times 2$, а елемент відкритого тексту подати у вигляді вектор-стовпця $(x, y)^T$. Зауважимо, що матриця $A$ повинна мати детермінант $D = ad - bc$, який взаємно простий із модулем $n$, тобто що НСД$(D, n)$ =1. Тоді на підставі твердження 4 для матриці $A$ існує обернена матриця

$$A^{-1} = \begin{pmatrix} D^{-1}d & -D^{-1}b \\ -D^{-1}c & D^{-1}a \end{pmatrix}$$

і процес шифрування/розшифрування відбувається за формулами

$$Y = A \cdot X \text{ і } X = A^{-1}Y,$$

де $X = (x, y)^T$, $Y = (x', y')^T$.

Такий метод шифрування називається **біграмним** [12].

**Приклад 3.2.2.** В алфавіті англійської мови ($n$=26) зашифрувати слово $''no''$ за допомогою матриці

$$A = \begin{pmatrix} 2 & 3 \\ 7 & 8 \end{pmatrix}.$$

*Розв'язання.* Літери $n$ і $o$ мають порядкові номери 13 і 14 відповідно. Отже,

$$Y = A \cdot X = \begin{pmatrix} 2 & 3 \\ 7 & 8 \end{pmatrix} \cdot \begin{pmatrix} 13 \\ 14 \end{pmatrix} = \begin{pmatrix} 68 \\ 203 \end{pmatrix} = \begin{pmatrix} 16 \\ 21 \end{pmatrix} \pmod{26}.$$

Таким чином, $Y =$ "$qv$". ♠

Цей метод можна узагальнити на $X = X_1X_2\cdots X_k$ $k$ біграм відкритого тексту. Для шифрування будується матриця $P$ розмірності $2 \times k$ із $k$ вектор-стовпців і, помноживши матрицю $A$ на матрицю $P$, дістаємо зашифрований текст.

**Приклад 3.2.3.** а) Зашифрувати текст "noanswer" за допомогою матриці $A$ з попереднього прикладу.

*Розв'язання.* Числові відповідники для літер слова "noanswer" утворюють такі вектор-стовпці: (13,14), (0,13), (18,22), (4,17). Тоді зашифрований текст набуває вигляду:

$$Y = A \cdot P = \begin{pmatrix} 2 & 3 \\ 7 & 8 \end{pmatrix} \cdot \begin{pmatrix} 13 & 0 & 18 & 4 \\ 14 & 13 & 22 & 17 \end{pmatrix} = \begin{pmatrix} 16 & 13 & 24 & 7 \\ 21 & 0 & 16 & 8 \end{pmatrix}.$$

Таким чином, зашифрований текст буде таким: "qvnayqhi".

б) Розшифрувати текст "fwmdiq" за допомогою матриці, оберненої до матриці $A$, з попереднього пункту а).

*Розв'язання.* Маємо

$$P = A^{-1}Y = \begin{pmatrix} 14 & 11 \\ 17 & 10 \end{pmatrix} \cdot \begin{pmatrix} 5 & 12 & 8 \\ 22 & 3 & 16 \end{pmatrix} = \begin{pmatrix} 0 & 19 & 2 \\ 19 & 0 & 10 \end{pmatrix} = \text{“}attack''. \spadesuit$$

Розглянемо коротко прості класичні шифри, які є шифрами підстановки.

**Двознакові шифри**. За використання цього шифру літери відкритого тексту кодують двознаками за допомогою таблиці літер алфавіту відкритого тексту. Пояснимо це прикладом.

**Приклад 3.2.4.** Зашифрувати повідомлення: "goal is S" за допомогою таблиці:

| | Z | Y | X | W | V |
|---|---|---|---|---|---|
| A | a | b | c | d | e |
| B | f | g | h | i/j | k |
| C | l | m | n | o | p |
| D | q | r | s | t | u |
| E | w | v | x | y | z |

**Відкритий текст:** goal is S.

**Криптограма:** BYCWAZCZBWDXDX. ♠

Підстановку можна робити за допомогою цифр. Під час використання цього шифру теж маємо в розпорядженні таблицю цифр, які підставляються замість літер відкритого тексту. Цей спосіб ілюструє розглянутий нижче приклад.

**Приклад 3.2.5.** Зашифрувати текст "attack in eleven" на підставі таблиці:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 2 | u | v | w | d | a | k | m | r | q | p |
| 4 | s | c | t | n | . | , | i | l | ; | o |
| 8 | x | y | z | ? | / | b | e | ; | f | j |

**Відкритий текст:** attack in eleven.

**Криптограма:** 24424224412546438647862186434 4. ♠

**Шифр із двома ключами**. Вищенаведений шифр можна узагальнити й ускладнити за допомогою використання двох ключів.

**Приклад 3.2.6.** Зашифрувати повідомлення "11 rakiet" на основі таблиці:

| | K | A | M | I | L | E |
|---|---|---|---|---|---|---|
| O | a | b | c | d | e | f |
| R | t | u | v | w | x | g |
| T | s | 6 | 7 | 8 | y | h |
| E | r | 5 | 0 | 9 | z | i |
| G | q | 4 | 3 | 2 | 1 | j |
| A | p | o | n | m | l | k |

**Відкритий текст:** 11 rakiet.
**Криптограма:** GLGLEKOKAEEEOLRK ♠

**Приклад 3.2.7.** Зашифрувати текст "olivares" на основі таблиці:

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 12 | m | u | r | p | h | y | s | l | a | w |
| 24 | b | c | d | e | f | g | i | j | k | n |
| 36 | o | q | t | v | x | y | . | , | ? | / |

**Відкритий текст:** olivares.

**Криптограма:** 360 127 246 363 128 122 243 126. ♠

Аналогічним чином можна будувати тризнакові шифри. Шифрування тризнаками відбувається так само як і шифрування двозначками.

**Монадичний шифр**. У цьому способі шифрування використовується таблиця, в якій рядки і стовпчики перенумеровані числами, але один стовпчик (або рядок) залишається без номера. Розглянемо приклад.

**Приклад 3.2.8.** Зашифрувати текст "worog atakuje." на основі таблиці:

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| - | k | e | x | a | - | - | d | e | i | m |
| 7 | l | b | f | g | j | k | n | o | p | q |
| 9 | r | s | t | u | v | w | y | z | . | , |

**Відкритий текст:** worog atakuje.

**Криптограма:** 96 78 91 78 74 4 93 4 76 94 75 2 99. ♠

**Шифри мультиалфавітні**. Черговим покращенням шифрів моноалфавітних є застосування мультиалфавітних шифрів. Усі методи цього типу мають такі спільні риси:

1) використовується множина пов'язаних між собою правил моноалфавітних підстановок;

2) ключ визначає, яке правило буде використано для перетворення тексту.

***Шифр Віженера*** – найбільш відомий шифр такого типу є мультиталфавітною підстановкою з періодичною ключовою послідовністю $\pi_1, \pi_2, \ldots, \pi_r$ з періодом $r$.

Нехай період послідовності дорівнює $r$, а кожна з підстановок $\pi_i$ є шифром Цезаря. Зазвичай ключ є деяким відрізком змістового тексту – словом або фразою. Його підписують під ВТ, повторюючи стільки разів, скільки потрібно. Додаючи номери літери ВТ і літери ключа, що стоїть під ним, за модулем $m$, одержують номер літери ШТ. Наприклад:

| ВТ | п | о | л | і | а | л | ф | а | в | і | т | н | а | п | і | д | с | т | а | н | о | в |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ключ | ц | е | з | а | р | ц | е | з | а | р | ц | е | з | а | р | ц | е | з | а | р | ц | е |
| ШТ | ї | ф | ф | і | р | ж | ь | з | в | ю | л | у | з | п | ю | ю | ч | ю | а | ґ | і | ж |

В українській мові 33 літери, занумеруємо їх таким чином:

| а | б | в | г | ґ | д | е | є | ж | з | и | і | ї | й | к | л | м | н |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |

| о | п | р | с | т | у | ф | х | ц | ч | ш | щ | ь | ю | я |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 |

Номер літери “п” – 19, номер літери “ц” – 26: отже, 19+26 (mod 33)=12 – номер літери “ї” і так далі.

Така поліалфавітна підстановка називається шифром Віженера.

Основною властивістю цього шифру є те, що на одну літеру відкритого тексту припадає багато літер тексту зашифрованого, одна на кожну літеру ключового слова. У такий спосіб інформація про частоту появи літер приховується. Але не вся інформація про структуру відкритого тексту підлягає приховуванню.

Розглянемо один із способів ламання такого типу шифрів [9]. У таких шифрах поліалфавітна підстановка визначається ключем – нескінченною послідовністю підстановок $\{\pi_1, \pi_2, \ldots\}$ на $m$-елементному алфавіті $X_m = \{x_1, x_2, \ldots, x_m\}$. Розглянемо три моделі вибору підстановок у цій послідовності. В усіх трьох вважається, що підстановки вибираються незалежно одна від одної.

**Модель 1**. $\pi_i,\ i = 1, 2 \ldots$ – довільні підстановки на $X_m$. $P(\pi_i = = \pi) = \frac{1}{m!}$, тобто всі підстановки рівноймовірні.

**Модель 2**. Кожна з підстановок $\pi_i,\ i = 1, 2 \ldots$ є шифром Цезаря (з циклічним зсувом алфавіту) із своїм ключем $k$. $P(\pi_i$ є циклічним зсувом на $k$ позицій$)=\frac{1}{m},\ 0 \le k \le m-1$, тобто всі циклічні зсуви рівноймовірні.

**Модель 3**. $\pi_i,\ i = 1, 2 \ldots$ – шифри Цезаря, $P(\pi_i$ є циклічним зсувом на $k$ позицій$) = p(k)$, де $p(k)$ – ймовірність літери з номером $k$ у мові. Ця модель відповідає випадку, коли ключова послідовність є змістовим текстом.

Криптоаналіз шифру Віженера досить простий, якщо відома довжина періоду $r$. У цьому випадку ШТ $Y = \{y_1, y_2, \ldots\}$ розбивають на $r$ фрагментів:

$$
\begin{gathered}
Y_1 = \{y_1, y_{r+1}, y_{2r+1}, \ldots\}, \\
Y_2 = \{y_2, y_{r+2}, y_{2r+2}, \ldots\}, \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
Y_r = \{y_r, y_{2r}, y_{3r}, \ldots\},
\end{gathered}
$$

тобто із ШТ вибирають літери, що лежать на відстані $r$, починаючи з першої, другої і т. д. Кожен із фрагментів зашифрований шифром Цезаря з певним ключем і тому легко розшифровується.

Таким чином одержуємо ключ $\{k_1, k_2, \ldots, k_r\}$.

При невідомому періоді $r$ спочатку шукають період, а потім – ключ, так, як вказано вище. Ідея знаходження періоду така. Припустимо, що ВТ описується моделлю джерела М1 (див. підрозділ 2.4), тобто літери ВТ незалежні і мають розподіл $p(x), x \in X_m$, відповідно до частот літер у мові. Тоді $p(x_0, x_1, x_2, \ldots) = p(x_0)p(x_1)p(x_2)\ldots$, не дивлячись на те, що це груба модель ВТ, але вона дозволяє одержати бажаний результат. Спочатку спробуємо розпізнати два випадки: або $r = 1$, або $r > 1$ – тобто дізнатися, зашифрований текст моноалфавітною чи поліалфавітною підстановкою. Нехай $N_t(Y),\ 0 \le t \le m-1$, – кількість входжень літер $t$ у ШТ $Y$. Для моноалфавітної підстановки (в даному випадку – шифру Цезаря) за законом великих чисел $\frac{N_t(Y)}{n} \to p(t-k),\ n \to \infty$, де $n$ – довжина тексту, $k$ – ключ. Тобто розподіл літер у ШТ такий самий, як і у ВТ, але циклічно зсунутий на $k$. Якщо ж період $r > 1$, то

$$\frac{N_t(Y)}{n} = \sum_{i=1}^{r} \frac{N_t(Y_i)}{n} * \frac{1}{r} \to \frac{1}{r} \sum_{i=1}^{r} p(t-k_i),\ n \to \infty, \tag{3.5}$$

бо кожен фрагмент $Y_i$ зашифрований шифром Цезаря з ключем $k_i$. Вираз у правій частині (3.5) задає "згладжений" розподіл на $X_m$, який ближче до рівномірного, ніж розподіл літер ВТ $\{p(t)\}$, оскільки ймовірність кожної літери тут є середнім арифметичним кількох імовірностей. За степенем "згладженості" розподілу літер у ШТ можна зробити висновок щодо значення $r$. Для порівняння розподілів використовують так званий $\psi$-тест.

Визначимо $\psi$-функцію від ШТ $Y$ як

$$\psi(Y) = \sum_{t=0}^{m-1} N_t(Y)(N_t(Y) - 1). \tag{3.6}$$

Оскільки права частина (3.6) не залежить від порядку доданків, то значення $\psi(Y)$ для будь-якої моноалфавітної підстановки при фіксованому ШТ $Y$ те саме. $\psi(Y)$ – випадкова величина, якщо ВТ – випадковий (нагадаємо, що він описується моделлю М1). Знайдемо

математичне сподівання та дисперсію $\psi(Y)$ у випадку моноалфавітної підстановки. $N_t(Y)$ має біноміальний розподіл із параметрами $n, p(n-k)$. Отже,

$$MN(_t(Y) = np(t-k),\ DN_t(Y) = np(t-k)[1-p(t-k)].$$

Звідки дістаємо

$$MN_t(Y)(N_t(Y)-1) = M(N_t(Y))^2 - MN_t(Y) = DN_t(Y) + (MN_t(Y))^2 - MN_t(Y) =$$
$$= np(t-k)(1-p(t-k)) + n^2p^2(t-k) - np(t-k) = n(n-1)p^2(t-k),$$

і

$$M\psi(Y) = \sum_{t\in X_m} MN_t(Y)(N_t(Y)-1) = n(n-1)\sum_{i=0}^{m-1} p^2(t). \qquad (3.7)$$

Отже, $M\psi(Y)$ залежить лише від довжини тексту та від мови ВТ.

Величину $I(Y) = \frac{\psi(Y)}{n(n-1)}$ називають індексом відповідності тексту $Y$. Для моноалфавітної підстановки математичне сподівання $MI(Y)$ – це величина стала і дорівнює $\sum\limits_{i=0}^{m-1} p^2(t)$. (Наприклад, в англійській мові $\sum\limits_{i=0}^{m-1} p^2(t) \approx 0,069$, у російській $\approx 0,059$).

Дещо складніше підраховується дисперсія $\psi(Y)$. Наведемо остаточний вираз:

$$D\psi(Y) = n(n-1)(n-2)(n-3)\sum_t p^4(t) + 4n(n-1)(n-2)\sum_t p^3(t) + 2n(n-1)\sum_t p^2(t).$$

У випадку поліалфавітної підстановки з періодом $r$ шифру Віженера $\psi(Y)$-функцію будемо позначати $\psi_r(Y)$, підкреслюючи залежність від довжини періоду. Тут уже $M\psi_r(Y)$ залежить від того, за яким законом вибирався ключ. Нехай ключ шифрування кожної літери $k$ вибирається з імовірністю $q(k)$ незалежно від інших ключів на періоді. Зокрема, може бути $q(k) = \frac{1}{m}$ або $q(k) = p(k)$, що відповідає моделям вибору ключа 2 і 3, розглянутим вище. Тоді

$$M\psi_r(Y) = \sum_{k_1,\dots,k_r\in X_m} q(k_1)\dots q(k_r)M(\sum_{t\in X_m}(\sum_{i=1}^{r} N_t(Y_i))(\sum_{j=1}^{r} N_t(Y)-1)/k_1\dots k_r).$$

Позначимо $p[t] = \sum\limits_{k=0}^{m-1} q(k)p(t-k)$. Оминаючи досить громіздкі перетворення, наведемо остаточну формулу для $M\psi_r(Y)$:

$$M\psi_r(Y) = n(\frac{n}{r} - 1)\sum_{t=0}^{m-1} p^2(t) + n^2(1 - \frac{1}{r})\sum_{t=0}^{m-1} p^2[t]. \qquad (3.8)$$

При $r = 1$ вираз (3.8) переходить у (3.7). З (3.8) також видно, що за великих $r$ різниця між $M\psi_r(Y)$ і $M\psi_{r+1}(Y)$ невелика. Найбільша різниця між $M\psi_1(Y)$ і $M\psi_2$ для англійської мови $\frac{M\psi_2(Y)}{n(n-1)} \approx 0,052$ при великих $n$. Тому безпосередньо за значенням індексу відповідності можна робити висновки щодо величини $r$ тільки для малих $r$ ($r = 1 \div 5$).

У загальному випадку за значенням $I(Y)$ розрізняють дві гіпотези: $r = 1$ і $r > 1$. (Для оцінки вірогідності висновку можна також вивести формулу для $D\psi_r(Y)$ при $r > 1$, але вона надто громіздка). При $r > 1$ перебирають значення $r$: для кожного можливого $r > 1$ ШТ розбивають на фрагменти і підраховують $\psi(Y_1), \ldots, \psi(Y_r)$. При $r$, кратних істинному періоду, всі величини $\psi(Y_i)$ будуть близькі до $M\psi_1$, при інших – істотно меншими. Ту ж саму ідею можна використати дещо інакше. У тексті зустрічаються однакові літери, біграми, триграми і т. д., які знаходяться на відстані, що кратна періоду. Тож вони шифруються теж однаково. Таким чином, якщо в ШТ підрахувати кількість літер, що знаходяться на відстані $j$ одна від одної і збігаються, то при величині $j$, що кратна періоду, кількість збігів різко зросте.

Щоб ускладнити криптоаналіз, застосовують багатоконтурну систему Віженера: ВТ шифрують спочатку шифром Віженера з періодом $r_1$, одержаний ШТ знову шифрують іншим шифром Віженера з періодом $r_2$, ... і так $n$ разів. Якщо періоди $r_1, r_2, \ldots, r_n$ взаємно прості, то результуючий шифр еквівалентний шифру Віженера з періодом $r_1 r_2 \ldots r_n$.

Ще одне ускладнення криптоаналізу запропонував Віженер: використання так званої *системи автоматичного ключа*, яка полягає в тому, що відкритий текст додається до ключового слова і створюється в такий спосіб ключ динамічний.

**Приклад 3.2.9.** Зашифрувати текст із ключем:
ключ: ц е з а р п о л і а л ф а
ВТ: п о л і а л ф а в і т н а
ШТ: ї ф ф і р б з л й і ґ ж а. ♠

Варто зауважити, що коли в криптосистемі Віженера використовується кільце порядку 33, то для символів ШТ можна використовувати не суму за модулем 33, а різницю або добуток за цим модулем. Тоді частоту входження літер у ШТ можна практично повністю приховати.

### 3.2.2. Варіація шифру Віженера

Розглянемо один із способів ускладнення криптоаналізу шифру Віженера. Цей спосіб полягає у використанні кілець, адитивні групи яких повноциклічні. Як було показано в пункті 2.9.5 для цього потрібно задати визначальний рядок додавання з одиницею кільця.

Визначальний рядок додавання з одиницею генерується таким алгоритмом.

**GEN-**$G(a, c, k)$
*Вхід:* Порядок $k$ і коефіцієнти виразу $f(i) = a \cdot i + c$, де НСД$(a, k)$=1.
*Вихід:* Рядок таблиці додавання – одновимірний масив $b = (b_1, b_2, \ldots, b_k)$.
*Метод:*
1) for $i = 0$ to $k - 1$ do $b_{i+1} := a \cdot i + c \ (mod\ k)$ od
2) for $i = 1$ to $k$ do
if $(b_i = 0 \wedge i \neq k)$ then change $b_i$ and $b_k$;
if $(b_i = 1 \wedge i \neq 0)$ then change $b_i$ and $b_1$;
od (* визначили ізоморфізм $g(i) = b_i$, де $i = 1, 2, \ldots, k$ *)
3) за масивом $b = (b_1, b_2, \ldots, b_k)$ будуємо масив $P[1 \times k]$
(* де зберігається визначальний рядок *)
$P[0] := b_1$;
for $i = 1$ to $k - 2$ do $P[b_i] := b_{i+1}$ od
$P[b_{k-1}] := 0$. (* побудували визначальний рядок $P$ *)

**Правильність алгоритму** випливає з того, що коли $i$ пробігає повну систему лишків, то $a \cdot i + c$ теж пробігає повну систему лишків за умови, що НСД$(a, k)$=1 [2].

**Часова складність** алгоритму GEN-$G$ – $O(k\log^2 k)$, оскільки множення цілих чисел ($< k$) має складність $O(\log^2 k)$, а всіх таких множень не більше ніж $k$.

Зазначимо, що алгоритм GEN-$G$ згенерує не більше ніж $(k-2)\varphi(k)$ визначальних рядків, де $\varphi$ – функція Ойлера. Але для першого сеансу цього досить, оскільки інші визначальні рядки можна генерувати вже за домовленістю з абонентом шляхом застосування алгоритмів генерації підстановок [18].

**Приклад 3.2.10.** Згенерувати алгоритмом GEN-$G$ визначальний рядок для кільця $G_6$, де $f(i) = 5 \cdot i + 4$.

Перший цикл алгоритму генерує таку послідовність значень:

1) $b_1 = 4;\ \ b_2 = 3;\ \ b_3 = 2;\ \ b_4 = 1;\ \ b_5 = 0;\ \ b_6 = 5.$

Другий цикл розставляє все на свої місця:

2) $b_1 = 1;\ \ b_2 = 3;\ \ b_3 = 2;\ \ b_4 = 4;\ \ b_5 = 5;\ \ b_6 = 0.$

Третій цикл генерує за масивом $b_1 = 1, b_2 = 3, b_3 = 2, b_4 = 4, b_5 = 5, b_6 = 0$ визначальний рядок $P = (1, 3, 4, 2, 5, 0)$. ♠

Як показано в пункті 2.9.5, кільце $G_k$, адитивна група якого повноциклічна, ізоморфне кільцю лишків $\mathcal{Z}_k$. Скористаємося цим ізоморфізмом у такому протоколі.

**Протокол обміну повідомленнями** передбачає виконання таких дій.

**Крок 0.** Попередньо Аліса і Боб секретним каналом обмінюються трійкою $(a, c, k)$, елементи якої є параметрами алгоритму GEN-$G$. За допомогою виразу $f(i) = a \cdot i + c$ згенерували початкову підстановку і потім за домовленістю зафіксували визначальний рядок $b = (b_1, b_2, \ldots, b_{k-1}, 0)$.

Після цього Аліса і Боб виконують такі кроки.

**Крок 1.**

а) Аліса будує систему виразів

$$l(x) = \begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1q}x_q, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2q}x_q, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_q + \ldots + a_{mq}x_q. \end{cases}$$

б) Перетворює $l(x)$ у кільці $G_k$ таким чином:

$$L(x) = B_r(B_{r-1}(\dots B_2(B_1(l(x)+a)+a_1)\dots+a_{r-1})+a_r)+a_{r+1},$$

де $B_i$ – невироджені матриці в кільці $G_k$ розмірності $m \times m$, $a, a_j$ – вектори розмірності $1 \times m$, $i = 1, 2, \dots, r, j = 1, 2, \dots, r+1$.

**Крок 2.**

а) Боб розв'язує систему $l(x) = v$, де $v$ – повідомлення, яке він хоче передати Алісі, обирає вектор $\bar{a}$ розмірності $1 \times q$.

б) Обчислює в кільці $G_k$ вектори значень $l(\bar{a}) = d$ і $L(\bar{x}+\bar{a}) = = d_1$ за модулем $k$.

в) Значення $v$ зберігає в таємниці, а значення $d$ і $d_1$ висилає Алісі відкритим каналом.

**Крок 3.**

а) Аліса обчислює обернені матриці до матриць $B_i$ в кільці $G_k$.

б) Знаходить значення $v$, оскільки всі дані для цього в неї є.

**Твердження 9.** *Обмін повідомленнями за наведеним протоколом виконується коректно.*

*Доведення* очевидним чином випливає з властивостей лінійних операторів і лінійних функцій у кільці $G_k$. Дійсно, оскільки

$$d_1 = L(\bar{x}+\bar{a}) = B_r(B_{r-1}(...B_2(B_1(l(\bar{x}+\bar{a})+a_0)+a_1)...)+a_{r-1})+a_r)+a_{r+1},$$

то

$$B_1^{-1}(B_2^{-1}(...(B_{r-1}^{-1}(B_r^{-1}(d_1)))...)-a_{r+1})-a_r)...)-a_1)-a_0 =$$
$$= B_1^{-1}(B_2^{-1}(...(B_{r-1}^{-1}(B_r^{-1}(B_r(B_{r-1}(...B_2(B_1(l(\bar{x}+\bar{a})+a_0)+a_1)...)+a_r)+a_{r+1})-$$
$$-a_{r+1})-a_r)...)-a_1)-a_0 = l(\bar{x}+\bar{a}).$$

Звідси дістаємо $l(\bar{x}) = l(\bar{x}+\bar{a})-d$. ■

**Криптоаналіз протоколу.** Очевидними кроками криптоаналітика є спроба розв'язати в кільці $G_k$ систему лінійних рівнянь

$$l(x) = d, \tag{3.9}$$

з метою знаходження $l(\bar{a})$. Зауважимо, що для цього потрібно знайти довільний розв'язок рівняння $l(x) = d$ в кільці $G_k$.

Потім за матрицями виразів $L(x)$ і $l(x)$ знайти матрицю $D^{-1}$ і значення $L(\bar{a})$. А для цього потрібно знайти розв'язок системи рівнянь

$$D^{-1}(B_r(B_{r-1}(\dots B_1(l(x)+a)\dots) = (a_{11}, a_{21}, \dots, a_{m1})^t. \qquad (3.10)$$

Обчислити різницю $L(\bar{x}+\bar{a}) - L(\bar{a})$ і розв'язати систему рівнянь

$$L(\bar{x}+\bar{a}) - L(\bar{a}) = d_1 \pmod{k}, \qquad (3.11)$$

звідки знаходиться значення $l(\bar{x})$ за допомогою матриці $D^{-1}$.

Описані дії криптоаналітика будуть успішними, якщо йому відомий ізоморфізм $g : G_k \to \mathcal{Z}_k$. Але цей ізоморфізм йому невідомий і тому він не може знайти розв'язки систем (3.9), (3.10) і (3.11). Отже, стійкість пропонованого протоколу цілком ґрунтується на ізоморфізмі $g$.

З розглянутої побудови скінченного кільця $G_k$ випливає, що ізоморфізмів між $G_k$ і $\mathcal{Z}_k$ існує $(k-2)!$. Звідси дістаємо, що коли у кожному сеансі обміну змінювати визначальний рядок і порядок кільця $k$, змінюючи тим самим ізоморфізм, то протокол буде мати необхідну стійкість.

**Приклад 3.2.11.** Нехай Аліса і Боб обмінялися трійкою $(7, 12, 25)$ і зупинилися на такому визначальному рядку кільця $G_{25}$:

$$b = (1, 6, 8, 10, 2, 4, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 12, 14, 16, 18, 20, 24, 22, 23, 0).$$

Другий цикл алгоритму GEN-$G_k$ знаходить ізоморфне відображення $g : \mathcal{Z}_{25} \to G_{25}$ (яке наводилося в пункті 2.9.5, а тут повторюється з метою зручності читання):

$$\begin{array}{lllll} g(0)=0, & g(5)=2, & g(10)=9, & g(15)=19, & g(20)=18, \\ g(1)=1, & g(6)=4, & g(11)=11, & g(16)=21, & g(21)=20, \\ g(2)=6, & g(7)=3, & g(12)=13, & g(17)=12, & g(22)=24, \\ g(3)=8, & g(8)=5, & g(13)=15, & g(18)=14, & g(23)=22, \\ g(4)=10, & g(9)=7, & g(14)=17, & g(19)=16, & g(24)=23. \end{array}$$

За цим ізоморфізмом третій цикл алгоритму GEN-$G_k$ будує масив $P[1 \times k]$ (для зручності читання він поданий нижнім рядком підстановки).

$$P = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \\ 1 & 6 & 4 & 5 & 3 & 7 & 8 & 9 & 10 & 11 & 2 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 24 & 12 & 23 & 0 & 22 \end{pmatrix}.$$

За масивом $P$ будуються таблиці операцій додавання і множення кільця $G_{25}$ (алгоритми побудови таблиць операцій кільця $G_k$ були наведені в пункті 2.9.5, а для кільця $G_{25}$ були подані і самі таблиці, згенеровані цими алгоритмами).

**Крок 1.** Нехай Аліса побудувала в кільці $G_{25}$ такі вирази (які подано не в дужках, у дужках наведено вирази, за якими відбувається шифрування і розшифрування):

$$l(x) = \left\{ \begin{array}{l} 2x_1 - 16x_2 + 7x_3 + 20x_4, \\ 0x_1 + 1x_2 - 17x_3 - 11x_4 \end{array} \right. \quad (= \left\{ \begin{array}{l} 2x_1 + 4x_2 + 7x_3 + 20x_4, \\ 0x_1 + 1x_2 + 11x_3 + 17x_4 \end{array} \right. )$$

і

$$L(x) = B_1(l(x) + (1,2)^t) =$$

$$= \left\{ \begin{array}{l} 9x_1 - 13x_2 + 10x_3 - 16x_4 + 3, \\ 18x_1 + 14x_2 - 18x_3 + 19x_4 + 16, \end{array} \right. \quad (== \left\{ \begin{array}{l} 9x_1 + 15x_2 + 10x_3 + 4x_4 + 3, \\ 18x_1 + 14x_2 + 2x_3 + 19x_4 + 16, \end{array} \right. )$$

де матриця $B_1$ має вигляд

$$B_1 = \left( \begin{array}{cc} 6 & 1 \\ 23 & 23 \end{array} \right).$$

(Спробуйте знайти обернену матрицю в кільці $G_{25}$, не знаючи ізоморфізму між $\mathcal{Z}_{25}$ і $G_{25}$).

**Крок 2.** Боб розв'язує систему лінійних рівнянь

$$l(x) = \left\{ \begin{array}{lcl} 2x_1 + 4x_2 + 7x_3 + 20x_4 & = & 5, \\ 0x_1 + 1x_2 + 11x_3 + 17x_4 & = & 3, \end{array} \right.$$

знаходить розв'язок $x = (0, 0, 13, 0)$ і вибирає вектор $\bar{a} = (0, 1, 0, 1)$.

Обчислює вектор $\bar{x} + \bar{a} = (0, 1, 13, 1)$ і значення

$$l(\bar{a}) = (6, 19) = d \text{ і } L(\bar{x} + \bar{a}) = (23, 13) = d_1.$$

Боб зберігає значення $v = (5, 3)$ в таємниці, а значення $d$ і $d_1$ висилає Алісі відкритим каналом.

**Крок 3.** Аліса виконує такі обчислення.

а) Знаходить обернену матрицю до $B_1^{-1}$ в кільці $G_{25}$:

$$B_1^{-1} = \left( \begin{array}{cc} 1 & 1 \\ 23 & 22 \end{array} \right).$$

б) Обчислює $B_1^{-1}(23, 13)^t$ і знаходить

$$B_1^{-1} = \left( \begin{array}{cc} 1 & 1 \\ 23 & 22 \end{array} \right) (23, 13)^t = (11, 6) = l(x + a) + (1, 2)^t.$$

в) Знаходить значення

$$(11, 6) - [(1, 2) + (6, 19)] = (11, 6) + (24, 2) = (5, 3) = v. \spadesuit$$

Із цього прикладу видно, що в обчислювальному сенсі найскладнішим етапом є побудова обернених матриць у кільці $G_k$. Щоб спростити ці обчислення, краще скористатися ізоморфізмом між кільцями $g : G_k \to \mathcal{Z}_k$ і вести обчислення в кільці лишків $\mathcal{Z}_k$. Коли обернені матриці будуть знайдені, то виконати обернені підстановки й отримати відповідні матриці в кільці $G_k$.

**Формування повідомлення.** Розглянемо питання: яким чином формуються параметри протоколу та повідомлення, яке потрібно передати. Зі сказаного вище випливає, що Аліса і Боб повинні мати алгоритми побудови визначального рядка кільця, який задає ізоморфізм. Для цього вони повинні обмінятися закритим каналом трійкою чисел $(a, c, k)$, де $a, c, k$ – параметри алгоритму GEN-$G_k$.

Далі з наведеного протоколу і прикладу випливає, що для передачі потрібного повідомлення $b = (b_1, b_2, \ldots, b_m)$ необхідно, щоб система рівнянь

$$l(x) = \begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1q}x_q = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2q}x_q = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_q + \ldots + a_{mq}x_q = b_m \end{cases} \tag{3.12}$$

мала розв'язок. Розв'язання такої системи зводиться до розв'язання системи однорідних рівнянь вигляду

$$l(x) = \begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1q}x_q - b_1x_0 = 0, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2q}x_q - b_2x_0 = 0, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_q + \ldots + a_{mq}x_q - b_mx_0 = 0, \end{cases} \tag{3.13}$$

де $x_0$ – додаткова невідома.

Серед розв'язків системи (3.13) потрібно знайти розв'язок, у якого остання координата, що відповідає додатковій невідомій, дорівнює одиниці (це буде окремий розв'язок системи (3.12)).

Отже, потрібно сформулювати умови, за яких така система буде сумісною. Ці умови дає

**Теорема 77.** *Система лінійних рівнянь $Ax = b$ сумісна над кільцем $G_k$ тоді, коли кожний розв'язок СЛОР $A^T y = 0$ за модулем $k$ ортогональний вектору вільних членів $b$ за модулем $k$.*

Із цієї теореми випливає, що гарантією сумісності системи $Ax = b$ є лінійна незалежність за модулем $k$ її рівнянь. Дійсно, у випадку лінійної незалежності рівнянь системи $Ax = b$ єдиним розв'язком СЛОР $A^T y = 0$ буде нульовий вектор. А нульовий вектор ортогональний довільному іншому вектору.

**Приклад 3.2.12.** Нехай літери алфавіту англійської мови перенумеровано природним чином (табл. 3.2.1).

Таблиця 3.2.1 (цифрових відповідників символів алфавіту)

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i/j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |

Крім того, Аліса і Боб домовилися працювати в кільці $G_{25}$, яке задане вищенаведеним визначальним рядком, і Аліса побудувала вирази, які були наведені у прикладі 3.2.11.

Припустимо, що Боб хоче передати Алісі повідомлення

*meet me in twelve.*

а) Боб розбиває повідомлення на блоки по два символи у блоці, цифровими відповідниками яких є такі пари чисел, взяті з таблиці 3.2.1:

| me | et | me | in | tw | el | ve |
|---|---|---|---|---|---|---|
| 11,4 | 4,18 | 11,4 | 8,12 | 18,21 | 4,10 | 20,4 |

.

б) Розв'язує систему рівнянь

$$l(x) = \begin{cases} 2x_1 + 4x_2 + 7x_3 + 20x_4 = 11, \\ 0x_1 + 1x_2 + 11x_3 + 17x_4 = 4. \end{cases}$$

Значення $v_1 = (11, 4)$ він тримає в секреті.

в) Обирає вектор $\bar{a}_1 = (0, 1, 0, 1)$ і обчислює значення $d = l(\bar{a}_1) = (6, 19)$ і до розв'язку даної системи $\bar{x} = (0, 4, 0, 0)$ додає вектор $\bar{a}_1 = (0, 1, 0, 1)$, і цю суму векторів $\bar{x} + \bar{a}_1 = (0, 3, 0, 1)$ підставляє в $L(x)$, знаходячи тим самим значення $d_1 = (10, 9)$. Значення $d$ і $d_1$ Боб передає Алісі.

Аліса, отримавши значення $d, d_1$, виконує такі обчислення:

а) Знаходить обернену матрицю до $B_1^{-1}$ в кільці $G_{25}$:

$$B_1^{-1} = \begin{pmatrix} 1 & 1 \\ 23 & 22 \end{pmatrix}.$$

б) Обчислює $B_1^{-1} d_1^t = B_1^{-1}(10, 9)^t$ і знаходить

$$B_1^{-1} = \begin{pmatrix} 1 & 1 \\ 23 & 22 \end{pmatrix} (10,9)^t = (17,1) = l(\bar{x} + \bar{a}_1) + (1,2)^t.$$

в) Обчислює значення

$$(17,1) - [(1,2) + l(\bar{a}_1)] = (17,1) - [(1,2) + (6,19)] = (17,1) + (24,2) = (11,4) = v_1.$$

а) Боб розв'язує систему рівнянь

$$l(x) = \begin{cases} 2x_1 + 4x_2 + 7x_3 + 20x_4 = 4, \\ 0x_1 + 1x_2 + 11x_3 + 17x_4 = 18. \end{cases}$$

Значення $v_2 = (4,18)$ він тримає в секреті.

б) Обчислює значення $d = l(\bar{a}_1) = (6,19)$ і до розв'язку даної системи $\bar{x} = = (0,22, 6,0)$ додає вектор $\bar{a}_1 = (0,1,0,1)$ і цю суму векторів $\bar{x} + \bar{a}_1 = (0,23,6,1)$ підставляє в $L(x)$, знаходячи тим самим значення $d_1 = (5,1)$.

в) Значення $d$ і $d_1$ Боб передає Алісі.

Аліса, отримавши значення $d, d_1$, виконує такі обчислення:

а) Знаходить обернену матрицю до $B_1^{-1}$ (у даному випадку в цьому немає потреби, оскільки матриця не змінювалася) в кільці $G_{25}$:

$$B_1^{-1} = \begin{pmatrix} 1 & 1 \\ 23 & 22 \end{pmatrix}.$$

б) Обчислює $B_1^{-1} d_1^t = B_1^{-1}(5,1)^t$ і знаходить

$$B_1^{-1} = \begin{pmatrix} 1 & 1 \\ 23 & 22 \end{pmatrix} (5,1)^t = (7,19) = l(\bar{x} + \bar{a}_1) + (1,2)^t.$$

в) Обчислює значення

$$(7,19) - [(1,2) + l(\bar{a}_1)] = (7,19) - [(1,2) + (6,19)] = (7,19) + (24,2) = (4,18) = v_2.$$

Оскільки наступний блок такий самий, як і перший, то $v_3 = (11,4)$. Це змушує Боба вибрати новий вектор $\bar{a}_2 = (0,0,1,1)$, за яким він обчислює значення

$$d = l(\bar{a}_2) = (2,0),\ \bar{x} + \bar{a}_2 = (0,4,1,1),\ d_1 = L(\bar{x} + \bar{a}_2) = (18,24).$$

Боб висилає Алісі $d = (2,0),\ d_1 = (18,24)$.

Аліса обчислює

$$B_1^{-1} d_1^t = B_1^{-1}(18,24)^t = (12,11) = l(\bar{x} + \bar{a}_2) + (1,2)^t.$$

Звідки знаходить

$$l(\bar{x} + \bar{a}_2) - [(1,2) + (2,0)] = (12,11) - (4,2) = (12,11) + (16,18) = (11,4) = v_3.$$

Цю процедуру Боб і Аліса повторюють стільки разів, скільки блоків у повідомленні (у даному випадку ще 4 рази). У такий спосіб Аліса отримує повідомлення

| 11,4 | 4,18 | 11,4 | 8,12 | 18,21 | 4,10 | 20,4 | . ♠ |
|---|---|---|---|---|---|---|---|
| me | et | me | in | tw | el | ve | |

У наведеному прикладі використовувалося одне і те саме кільце, але можна при кожному сеансі передачі або з певним періодом між передачами змінювати кільце. Можна також змінювати вектори $a$ і $\bar{a}$, які змінюють значення $l(\bar{a})$ і $L(\bar{x}+\bar{a})$.

**Обчислювальні особливості.** Враховуючи, що обчислення в кільці $G_k$ не є звичними при обчисленнях, то покращити ефективність шифрування і розшифрування можна, якщо знову скористатися ізоморфізмом між кільцями $G_k$ і $\mathcal{Z}_k$. Дійсно, пошук протилежного елемента до елемента $a$ в кільці $\mathcal{Z}_k$ зводиться до обчислення різниці $k-a$, а обчислення оберненого елемента до $a$ виконується шляхом застосування розширеного алгоритму Евкліда для розв'язання рівняння $ax+ky=1$ (розширений алгоритм Евкліда обчислює розклад $ax+by=d$, де $d$ =НСД$(a,b)$). Результатом виконання цього алгоритму є значення $x=a^{-1}$.

Використовуючи ізоморфізм $g$ з прикладу 3.2.11, система рівнянь із прикладу 3.2.12 в кільці $\mathcal{Z}_{25}$ набуває вигляду

$$\bar{l}(x)=\begin{cases} 5x_1+6x_2+9x_3+21x_4=11,\\ 0x_1+1x_2+11x_3+14x_4=6.\end{cases}$$

Розв'язком цієї системи є вектор $x=(0,6,0,0)$, якому відповідає розв'язок $\bar{x}=(0,4,0,0)$ системи

$$l(x)=\begin{cases} 2x_1+4x_2+7x_3+20x_4=11,\\ 0x_1+1x_2+11x_3+17x_4=4.\end{cases}$$

У кільці $\mathcal{Z}_{25}$ матриці

$$B_1=\begin{pmatrix} 6 & 1\\ 23 & 23\end{pmatrix} \quad \text{відповідає матриця} \quad \bar{B}_1=\begin{pmatrix} 2 & 1\\ 24 & 24\end{pmatrix}$$

або

$$\bar{B}_1=\begin{pmatrix} 2 & 1\\ -1 & -1\end{pmatrix}, \quad \text{де обернена до неї} \quad \bar{B}_1^{-1}=\begin{pmatrix} 1 & 1\\ -1 & -2\end{pmatrix}.$$

Оскільки вектору $d_1 = (10, 9)$ кільця $G_{25}$ відповідає вектор $\bar{L}(x + a) = (4, 10)$ кільця $\mathcal{Z}_{25}$, то $\bar{B}_1^{-1}(4, 10)^t = (14, 1)$. Далі вектору (1,2) кільця $G_{25}$ відповідає вектор (1,5) кільця $\mathcal{Z}_{25}$, а вектору (6,19) кільця $G_{25}$ відповідає вектор (2,15) кільця $\mathcal{Z}_{25}$. А звідси дістаємо вектор $(14, 1) - (1, 5) - (2, 15) = (11, 6)$ в кільці $\mathcal{Z}_{25}$, якому відповідає вектор $l(\bar{x}) = (11, 4)$ в кільці $G_{25}$.

***Шифр гамування***. Гамування є одним із найбільш зручних у реалізації шифрів. Цей метод шифрування близький до методу Віженера, про який говорилося вище. У цьому шифрі використовується алфавіт повідомлень і криптограм $X_m$, разом із множиною ключів $K$. Тоді для довільного відкритого тексту $p = \{a_1, a_2, \ldots, a_s, \ldots\}$ і довільного ключа $k \in K$

$$E(A, k) = b_1, b_2, \ldots, b_s, \ldots,$$

де $b_1 = a_1 + f_1(k) \pmod m, \ldots, b_i = a_i + f_i(k) \pmod m$, $i = 2, 3, \ldots$.

Таким чином, шифр гамування полягає в додаванні за модулем $m$ тексту відкритого і послідовності чисел із $X_m$. Ця послідовність чисел називається **гамою** і утворюється з початкового ключа та попередніх знаків відкритого тексту. Очевидно, що в такому формулюванні, розглянутий вище шифр Вернама, є шифром гамування.

Нині використовується множина ключів $K = X_m^n$, де $X_m^n = X_m \times \times X_m \times \cdots \times X_m$.

Гама може бути як скінченною, так і нескінченною (реально це означає, що її довжина дорівнює довжині шифрованого тексту і вона використовується для передачі тільки одного повідомлення) і змінюється за певним законом.

Розшифрування виконується шляхом накладання ключової гами на криптограму з використанням функції розшифрування.

Стійкість шифрів, отриманих методом гамування, визначається характеристиками гами – її довжиною і статистичними характеристиками. Якщо використовується нескінченна випадкова гама, то шифр стає стійким, тобто теоретично незламним.

Метод гамування стає безсильним, якщо зловмиснику стають відомі фрагмент початкового тексту і відповідна йому криптограма. Зловмисник може це зробити на основі догадок про зміст початко-

вого тексту. Наприклад, якщо текст починається зі слів "ЦІЛКОМ ТАЄМНО", то криптоаналіз усього тексту значно полегшується. Це необхідно враховувати у розробці реальних систем інформаційної безпеки.

## 3.3. Шифри потокові і блокові

Потокові шифри, як сказано вище, перетворюють ВТ на ШТ по одному біту або байту. Проста реалізація потокового шифру показана на рис. 3.3.1.

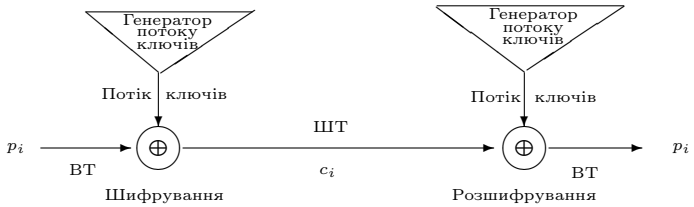



Рис. 3.3.1. Потоковий шифр

Під час роботи потокового шифру працює генератор потоку ключів, який генерує потік бітів $k_1, k_2, \ldots, k_i$. Цей потік ключів і потік відкритого тексту $p_1, p_2, \ldots, p_i$ перетворюється з допомогою операції $XOR$ – додавання за модулем 2 – і в результаті дістаємо потік бітів ШТ:

$$c_i = p_i \oplus k_i.$$

Розшифрування операції $XOR$ виконується над бітами ШТ і тим самим потоком ключів для відновлення ВТ:

$$p_i = c_i \oplus k_i = p_i \oplus k_i \oplus k_i.$$

Оскільки $p_i \oplus k_i \oplus k_i = p_i$, то розшифрування відбувається правильно.

*Безпека цієї системи* цілком залежить від властивостей генератора ключів. Якщо генератор потоку ключів видає нескінченну послідовність нулів, то операція втрачає сенс, оскільки на виході буде

ВТ. Якщо генератор видає послідовність випадкових, а не псевдовипадкових бітів, то така система буде ідеально безпечною.

Оскільки абсолютно випадкового потоку ключів не можна дістати, то ідеально безпечний шифр побудувати неможливо. Крім того, якщо генератор генерує кожний раз один і той самий потік ключів, то такий шифр неважко зламати. Дійсно, якщо зловмиснику дістався ШТ і відповідний ВТ, то він знаходить за допомогою операції $XOR$ потік ключів. Більше того, він зможе розшифрувати і прочитати всі попередні повідомлення, які він перехопив.

Отже, всі потокові шифри використовують ключі. Вихід генератора ключів є функцією ключа (наприклад, генератор із регістром зсуву) і тепер, коли зловмисник отримає пару ВТ/ШТ, то він зможе прочитати тільки ті повідомлення, які зашифровані тим же ключем. Як тільки ключ змінюється, зловмиснику потрібно починати все спочатку. У попередньому розділі розглядалися типи генераторів випадкових послідовностей і принципи їхньої побудови. Зауважимо тільки, що генератори ключів для абонентів повинні бути синхронними. У протилежному випадку як тільки якийсь біт буде неправильним, то результат розшифрування теж буде неправильним. Це означає, що генератор повинен видавати один і той самий потік ключів і для шифрування і для розшифрування. Але якість генератора ключів, як було вказано вище, залежить від періоду генератора. Якщо період менший розміру ВТ, то різні частини ВТ будуть зашифровані однаковим чином, а це сильно впливає на стійкість системи.

**Класичні блокові шифри.** Вищерозглянуті методи підстановки і перестановки хоча є простими методами шифрування, але вони склали основу для деяких шифрів, які нині використовують зрідка, але вони досить практичні.

***Роторові машини***. Роторові машини широко використовувалися під час Другої світової війни німецькою армією (машина ENIGMA) і японською армією (машина PURPLE). Роторові машини давали спосіб створення шифрограм, який був значно важчий для

зламування.

Основа будови роторової машини показана на рис. 3.3.2. Машина складається з декількох роторів, які є циліндрами і які можуть повертатися та передавати електричні імпульси. Кожний циліндр має 26 вхідних контактів і 26 вихідних, а також внутрішні провідники, які з'єднують ці контакти відповідним чином. Для спрощення на рисунку показано тільки три з'єднання в кожному циліндрі.

Якщо приписати кожному вхідному і вихідному контакту літеру алфавіту, то кожний окремий циліндр визначає моноалфавітну підстановку. Наприклад, якщо на цій схемі оператор натисне клавішу, яка відповідає літері $A$, то електричний сигнал піде до першого контакту першого циліндра, а потім піде через внутрішні з'єднання до 24 вихідного контакту.

Розглянемо роторову машину з одним циліндром. Після кожного натискання клавіші на вході циліндр переміщується на одну позицію, і внутрішні контакти також відповідним чином переміщуються. У результаті визначається новий шифр моноалфавітної підстановки. Після написання 26 літер вхідного текста циліндр повернеться до вихідної позиції. Дістаємо в такий спосіб алгоритм мультиалфавітної підстановки з околом 26 символів.

Система з одним циліндром є тривіальною і не виникає якихнебудь труднощів у її зламуванні. Але сила ротора полягає на можливості використання багатьох циліндрів, де вихідні контакти одного циліндра з'єднані з вхідними контактами наступного.

На рис. 3.3.2 показано трироторову систему. Ліва частина схеми представляє позицію, в якій дані, впроваджені оператором до першого контакту (літера A проходить через три циліндри до виходу, а потім до контакту другого циліндра (який відповідає літері B у криптограмі).

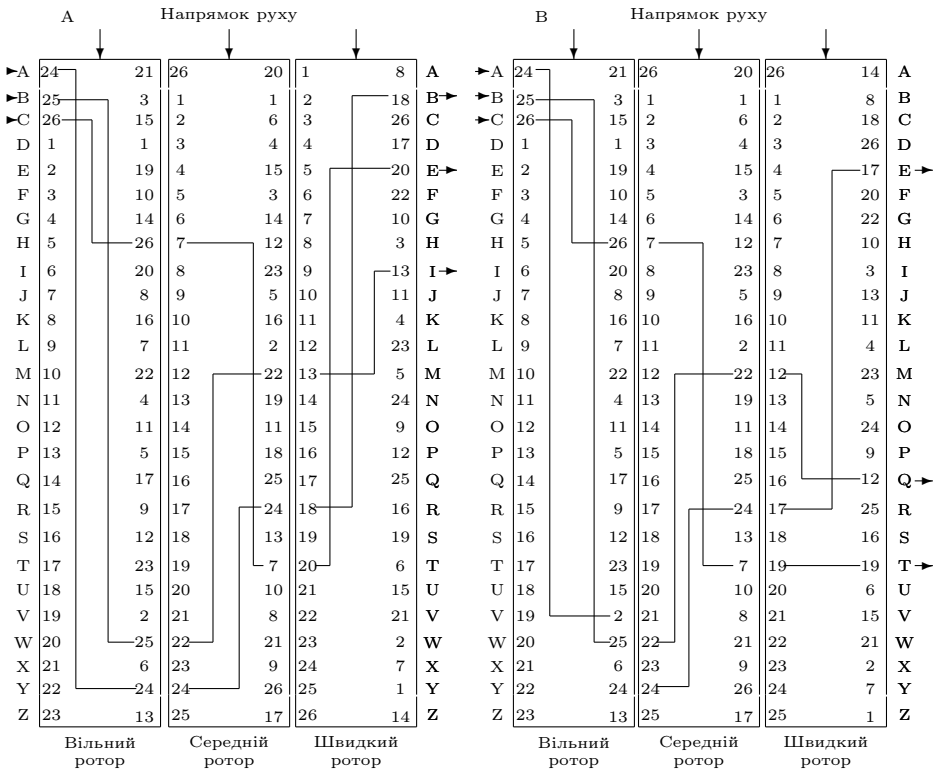



Рис. 3.3.2. Трироторова машина зі з'єднаннями у вигляді нумерованих контактів

У системі з багатьма циліндрами найдальший від оператора циліндр обертається на одну позицію при кожному натисканні клавіші оператором. При кожному повному обертанні зовнішнього циліндра, внутрішній циліндр обертається на одну позицію з'єднання. У результаті в системі використовується $26 \times 26 \times 26 = 17576$ різних алфавітів підстановки. Додавання четвертого і п'ятого циліндрів дає алфавітні околи відповідно 456976 літер і 11881376 літер. Така довжина околу виключає реальні можливості простого криптоаналізу на підставі частотного аналізу.

Значення ротора полягає в тому, що він визначив напрямок у криптографії, який привів до створення найсучасніших шифрів. Одним із таких шифрів є шифр DES (Data Encryption Standard),

який став американським стандартним шифром.

**Алгоритм шифрування DES**. Вищеописані алгоритми шифрування характеризуються низьким рівнем безпеки. Зокрема, ці методи наражаються на атаки частотним методом. Тому тепер вони мають лише академічне значення, а в комерційних застосуваннях не використовуються. З точки зору потреб охорони ідентичності і секретності інформації, яка міститься в повідомленні, необхідна розробка нових криптографічних технік. Рівень безпеки в таких системах досягається за допомогою багатократного, каскадного виконання перетворень у процесі шифрування.

Алгоритмом симетричного шифрування є вищезгаданий шифр DES. Цей шифр належить до блокових шифрів. У таких шифрах охороні підлягає менша порція інформації, яка в подальшому підлягає криптографічним перетворенням. Можна довести, що рівень безпеки криптограми зростає, якщо довжина ключа шифрування порівнювана з довжиною відкритого тексту. Оскільки документи, які охороняються, часто мають значний об'єм, то застосування блокового ключа такої самої довжини просто неможливе. Отже, блоковість шифра повинна підвищувати рівень безпеки криптографічного алгоритму.

Алгоритм DES був розроблений фірмою IBM на замовлення Національного Бюро Стандартів США у другій половині 70-х рр. минулого століття. Оскільки цей шифр був призначений для використання в урядових установах, то він мусив пройти відповідну акредитацію, яка давала б право на його використання. В 1977 р. алгоритм DES дістав перший п'ятирічний термін використання. Його змінили лише 2001 р. Основними властивостями, які привели до вибору алгоритму DES як офіційного стандарту криптографічної охорони, стали такі властивості:

• *Застосовані в алгоритмі методи шифрування опиралися на операціях підстановки і перестановки. Ці операції просто і швидко реалізуються в довільній комп'ютерній системі. Це було особливо важливим з точки зору дефіциту обчислювальних потужностей у доступних на той час комп'ютерах. Вважається, що алгоритм DES і деякі інші алгоритми більше ніж у 1000 разів швидші від*

*більшості алгоритмів асиметричних (RSA, Ель-Гамаля і т. п.).*

• *Операції підстановки і перестановки можуть реалізовуватися апаратно. Завдяки цьому, стає можливою велика швидкість шифрування, а це дає можливість його застосовувати в системах реального часу.*

• *Завдяки блоковості стало можливим використання відносно короткого ключа і забезпечення високого рівня безпеки. Застосування ключа великої довжини не є ефективним із точки зору погіршення часових характеристик шифрування.*

• *Алгоритм із самого початку був відкритим. Завдяки цьому користувач міг переконатися самостійно, що він не має жодних таємних входів, які б призводили до його ламання.*

Операції, які виконуються в алгоритмі DES можна поділити на три основні групи: операції підготовчі, операції ітераційного шифрування і операції закінчення. Підготовчі операції і операції закінчення виконують перестановки вхідних і вихідних даних, а також ключа. Перестановки виконуються на основі незмінної, описаної в стандарті, схеми. Як операція шифрування в алгоритмі DES використовується симетрична сума. Ця операція має аргументи однакової довжини. Оскільки довжина ключа і блока, який шифрується, в даному методі різні, то необхідне їхнє спеціальне перетворення. З цією метою шифрований блок ділиться на дві рівні частини довжини 32 біти. У кожній із 16 ітерацій шифруванню підлягає тільки права половина блока даних, причому при переході до наступної ітерації, права і ліва частини блока даних міняються місцями. Операція симетричної суми виконується на словах довжини 48-бітів. Із цією метою кожна з половин блока даних розширюється до заданої довжини за допомогою спеціальної операції. Це розширення полягає в багаторазовій вставці до 48-бітового вхідного слова вибраних бітів 32-бітової половини блока даних. Разом із розширенням виконується операція шифрування на основі перестановки.

Щоб досягти відповідного рівня безпеки, шифрування з використанням симетричної суми виконується 16 разів. Зазначимо, що кожна ітерація виконується з іншим ключем шифрування, який отримано з введеного користувачем базового ключа. Для створення

нового ключа шифрування ключ базовий ділиться на дві рівні половини по 28 бітів кожна, які після цього циклічно пересуваються на вказане число бітів. Після виконання операції зсуву половини знову з'єднуються в 56-бітовий ключ, з якого вибираються певні біти, і в такий спосіб утворюється 48-бітовий ключ ітерації.

Схема блокового алгоритму показана на рис. 3.3.3.

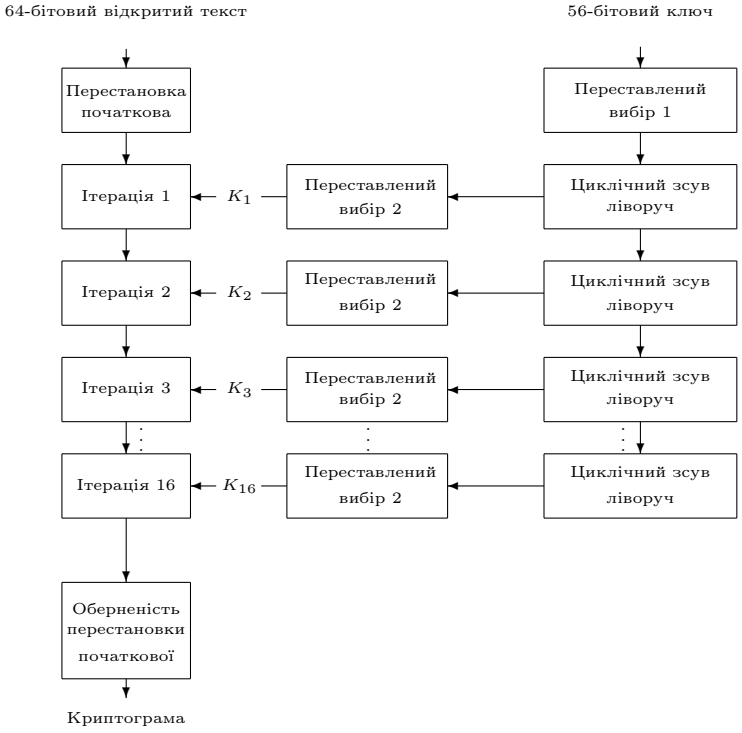



Рис. 3.3.3. Схема блокового алгоритму DES

Створені в такий спосіб ключі застосовують до шифрування блоку інформації. Цей блок сумується симетрично, після чого дістаємо часткову криптограму, яка далі підлягає перетворенню з використанням так званого 5-блокового методу. Із цією метою вона ділиться на вісім шестибітових частин. До кожної із цих частин приписується один оригінальний 5-блок. 5-блок є таблицею, яка має 16 стовпчиків і 4 рядки. Серед 6 бітів, перший і останній адресуються рядкам таблиці, у той час як 4 середні біти адресуються її стовпчи-

кам. Оскільки 5-блоки включають 4-бітові числа, то 48-бітове слово скорочується до 32 бітів.

Останнім кроком шифрування в рамках ітерації є перетворення 32-бітового вхідного слова згідно з так званою $P$-функцією (рис. 3.3.4).

У 2001 р. алгоритм DES був замінений на алгоритм AES, але DES і сьогодні трактується як метод, що гарантує достатній рівень безпеки для більшості застосувань. Детальний опис алгоритму DES можна знайти в [40].

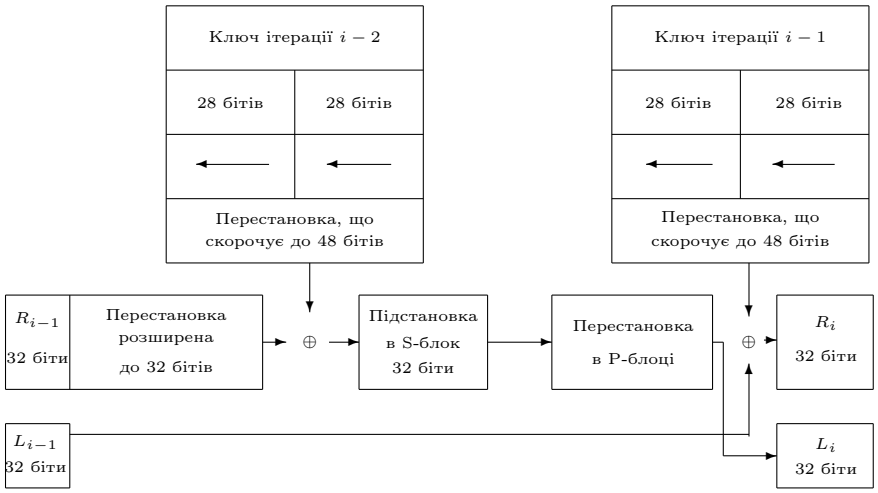



Рис. 3.3.4. Схема однієї ітерації алгоритму DES

### 3.3.1. Користь класичних алгоритмів шифрування

Класичні алгоритми шифрування, які були розглянуті вище, складають основу симетричних криптографічних систем і мають перевагу над іншими такими системами в тому, що вони прості в реалізації. Крім того, мультиалфавітні шифри і шифри перестановки стійкіші до ламань порівняно з простими шифрами підстановки. Але коли ключ короткий, а повідомлення довге, то методи криптоаналізу ламають такі шифри.

Класичні шифри і, навіть шифри підстановки, можуть виявитися *досить стійкими*, якщо утворювати їхні композиції з криптографічними ключами, які задовольняють певним умовам. Це є причиною того, що прості шифри підстановки широко використовують і нині у криптографічних системах і протоколах.

Розглянемо приклад протоколу, який використовує найпростіший шифр підстановки і який приводить до двох важливих умов, які забезпечують стійкість протоколів.

Нехай над адитивною групою $G_n$ (група лишків за модулем $n$ відносно операції додавання) задана функція$f(x)$, що має такі властивості:

а) **Односторонність**. Для заданого елемента $x \in G_n$ функція $f(x)$ ефективно обчислюється, в той час як майже для всіх елементів $y \in G_n$ і довільного ефективного алгоритму $A$ ймовірність $P$ знаходження значення $x$ за значенням $y$ досить мала.

б) **Гомоморфізм**. Для всіх $x_1, x_2 \in G_n$ виконується рівність $f(x_1 + x_2) = f(x_1) \cdot f(x_2)$.

Існує багато функцій, які задовольняють цим умовам. Використовуючи їх, можна побудувати протокол, який дозволяє (наприклад, Алісі) довести (наприклад, Бобу), що Аліса знає прообраз елемента $f(z)$, де $z < n$, не повідомляючи самого прообразу. Це досягається за допомогою простого протоколу, який ґрунтується на шифрі зсуву.

**Протокол 2**.

*Спільні вхідні дані:*

Функція $f$, яка задовольняє умови а) і б);

$X = f(z)$ для деякого $z \in G_n$.

*Початкові дані Аліси:*

$z < n$. (* закриті вхідні дані Аліси *)

*Висновки Боба:*

Аліса знає елемент $z \in G_n$, такий, що $X = f(z)$.

Наступні кроки виконуються $m$ разів:

1. Аліса вибирає число $k \in G_n$, обчислює число $Commit = f(k)$, де $k$ – число, яке випадково вибирається з однаковою ймовірністю із $G_n$.

2. Боб вибирає з однаковою ймовірністю випадкове число $Challenge$ з множини $\{0, 1\}$ і висилає його Алісі.

3. Аліса обчислює число

$$Response = \begin{cases} k, & \text{якщо} \quad Challenge = 0, \\ k + z \pmod{n}, & \text{якщо} \quad Challenge = 1, \end{cases}$$

і висилає його Бобу (Якщо $Challenge$=1, то $Response$ є результатом шифрування на основі шифра зсуву й одноразового ключа $k$).

4. Боб перевіряє умову

$$f(Response) = \begin{cases} Commit, & \text{якщо} \quad Challenge = 0, \\ Commit \cdot X, & \text{якщо} \quad Challenge = 1. \end{cases}$$

Якщо на якомусь кроці виникає помилка, то Боб відкидає доведення і перериває виконання протоколу, у протилежному випадку Боб приймає доведення тотожності особи Аліси.

У цьому протоколі величина $X = f(z)$ виступає в ролі криптографічного посвідчення особи Аліси, яке дає змогу її ідентифікувати. Це посвідчення може використовувати тільки Аліса, оскільки тільки вона знає, як із ним поводитися, знаючи прообраз $z$. Цей протокол демонструє, як Аліса показує своє посвідчення без передачі Бобу яких-небудь відомостей про прообраз $z$.

**Стійкість класичних шифрів**. Аналізуючи протокол 2, маємо питання: який рівень секретності він забезпечує і що дозволяє Алісі тримати свою інформацію в секреті? Стверджується, що цей протокол забезпечує *вищий* рівень безпеки, оскільки після запуску цього протоколу Боб *не отримує* абсолютно ніякої інформації про елемент $z \in G_n$, крім тієї, яку він отримує в результаті аналізу величини $f(z)$ (спільні вхідні дані дають тільки апріорну інформацію).

Зауважимо, що шифрування на основі зсуву

$$Response = z + k \pmod{n}$$

утворює перестановку елементів групи $G_n$. Якщо $k$ вибирається рівномірно із $G_n$ і $G_n = K = M$, то перестановка переводить це число в елемент $Response \in G_n$, оскільки вона відображає рівномірний розподіл у рівномірний. Це означає, що заданий зашифрований текст $Response$ може бути утворений за допомогою довільного ключа з групи $G_n$ (ймовірнісний простір складається з простору ключів і простору повідомлень). Іншими словами, у повідомленні $Response$

з однаковою ймовірністю може бути зашифрований довільний елемент $x \in G_n$. Отже, початковий текст $z$ не залежить від зашифрованого тексту *Response*, тобто зашифрований текст не несе жодної інформації про початковий текст $z$.

**Контрольні запитання**

1. Яка різниця між симетричними й асиметричними криптосистемами?
2. Дайте означення шифрів підстановки, перестановки і блокових.
3. У чому полягає метод одноразового блокнота?
4. Дайте означення шифрів Віженера та Вернама.
5. Дайте означення моноалфавітних, мультиалфавітних, двознакових, тризнакових, монадичних шифрів і шифрів з одним і двома ключами.

**Задачі і вправи**

1. Чи є шифр Вернама шифром підстановки? Чи є цей шифр моноалфавітним, чи мультиалфавітним?
2. У чому полягає різниця між шифром Вернама і одноразовим блокнотом? Чому шифр одноразового блокнота абсолютно захищений від злому.
3. Чому шифри підстановки і перестановки, не дивлячись на їхню нестійкість до злому, широко використовуються в сучасних криптосистемах і протоколах?
4. Зашифрувати шифрами, які розглянуто в цьому розділі, повідомлення
   а) "зустріч об одинадцятій"; б) "10 ракет".
5. Побудувати шифр, який є суперпозицією двох шифрів перестановки і підстановки.
6. Вкажіть місця в алгоритмі DES, в яких застосовуються шифри підстановки, перестановки та шифр Вернама.
7. Застосовуючи метод частотного аналізу, виконати криптоаналіз тексту:

"I love Mary and my children. My children are Bob and Alice. Now they study at the California university. This university is very high level university. Bob and Alice will be study two years at this university."

8. Зашифрувати повідомлення "I love Mary and my children" за допомогою протоколу і даних прикладу 3.2.11.

## 3.4. Асиметричні криптографічні системи

До сучасних систем шифрування належать асиметричні системи, які називають системами з відкритим ключем.

### 3.4.1. Криптологія відкритого ключа

Електромеханічний ротор спричинив швидкий розвиток досить складних систем шифрування. А з появою комп'ютерів виникли ще складніші криптографічні системи. Найбільш знаною серед цих систем була Lucifer фірми IBM, яка стала основою стандарту шифрування DES. Як ротор, так і DES, разом з їхніми вдосконаленнями, були побудовані на основі операцій підстановки і перестановки.

Розвиток методів шифрування з відкритим ключем, як зазначалося, привів до революції у криптографії, що викликало радикальні зміни порівняно з минулими методами. Алгоритм із відкритим ключем ґрунтується на математичних функціях, а не на перестановках і підстановках. Найважливішим тут є те, що шифрування з відкритим ключем є асиметричним, використовує два ключі на відміну від симетричного шифрування, де застосовується лише один ключ. Використання двох ключів має глибокий сенс для ідентифікації особи та розподілу ключів доступу. Але ейфорії тут немає, тому що існує декілька частих непорозумінь, пов'язаних із шифруванням за допомогою відкритого ключа.

1) Існує переконання, що шифрування з відкритим ключем більш стійке до ламання ніж шифрування з ключем секретним. Насправді безпека кожної системи шифрування залежить від довжини ключа і затрат часу на обчислення, які необхідні для ламання ключа. Немає жодних фактів, які б свідчили про те, що один із цих двох типів шифрування є кращим від другого з точки зору стійкості до зломів.

2) Друга помилкова думка зводиться до того, що шифрування з відкритим ключем є методом загального застосування, який виконує шифрування старим способом. Отже, з точки зору додаткових витрат на обчислення в сучасних системах шифрування з ключем відкритим не видно, що шифрування симетричне перестане бути

актуальним.

3) Нарешті, існує переконання, що розподіл ключів у випадку застосування шифрування з відкритим ключем є тривіальним порівняно з обтяжливими обставинами, що пов'язані з необхідністю звертатися до центральних систем розподілу ключів. Насправді необхідна якась форма протоколу, яка включає роботу із центральною системою розподілу ключів і потребує наявності відповідних процедур реалізації, а це, як правило, не є простим і ефективним.

Розглянемо коротко основи методів шифрування з відкритим ключем.

Як сказано у вступі, у симетричних системах шифрування використовується один і той самий ключ як до шифрування, так і до розшифрування. Але можна побудувати криптографічний алгоритм, який використовує один ключ до шифрування, а другий ключ, пов'язаний із першим, до розшифрування. Такого типу алгоритми мають такі важливі риси:

– неможливо під час виконання обчислень знайти ключ розшифрування, знаючи тільки криптографічний алгоритм і ключ шифрування;

– кожний із цих двох ключів можна вживати як до шифрування, так і до розшифрування. Такого типу алгоритмом шифрування є алгоритм шифрування RSA.

Будемо припускати, що обидві риси алгоритмів шифрування з відкритим ключем наявні, а потім розглянемо алгоритми шифрування, у яких відсутня друга риса.

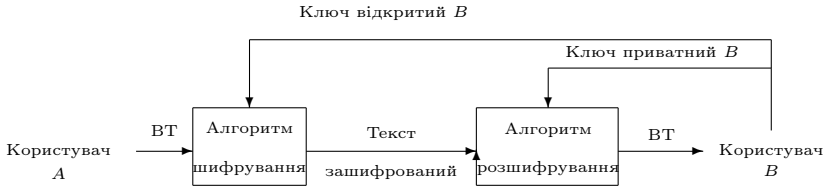



Рис. 3.4.1. Спрощена модель шифрування з відкритим ключем

На рис. 3.4.1 зображено процес шифрування з відкритим ключем, де основними етапами є такі:

1. Кожна прикінцева система в мережі генерує пару ключів до шифрування і розшифрування.

2. Кожна система публікує свій відкритий ключ шляхом його реєстрації в публічному реєстрі системи. Другий ключ є приватним і зберігається в таємниці.

3. Якщо абонент $A$ хоче вислати повідомлення до $B$, то він його шифрує за допомогою відкритого ключа $B$.

4. Якщо $B$ отримав повідомлення, то він його розшифровує за допомогою свого приватного ключа. Жоден інший абонент не в змозі прочитати повідомлення, оскільки тільки $B$ знає свій приватний ключ.

Застосовуючи такий метод, усі користувачі мають доступ до публічних ключів, а приватні ключі генеруються кожним користувачем і не повинні пересилатися лініями зв'язку.

Доки система контролює свої приватні ключі, доти вона безпечна для провідомлень, які до неї надходять. У довільний момент система може змінити свій приватний ключ і опублікувати новий відкритий ключ, який відповідає приватному.

Аби відрізнити ключ, який застосовується до симетричного шифрування, будемо його називати **таємним ключем**. Ключі, які застосовують до шифрування в асиметричних системах, будемо називати **відкритим ключем** і **ключем приватним** відповідно.

Розглянемо детальніше найважливіші елементи системи шифрування з відкритим ключем. Ці елементи показано на рис. 3.4.2, де маємо джерело повідомлення $A$, яке генерує повідомлення у вигляді відкритого тексту $X = [X_1, X_2, \ldots, X_M]$.

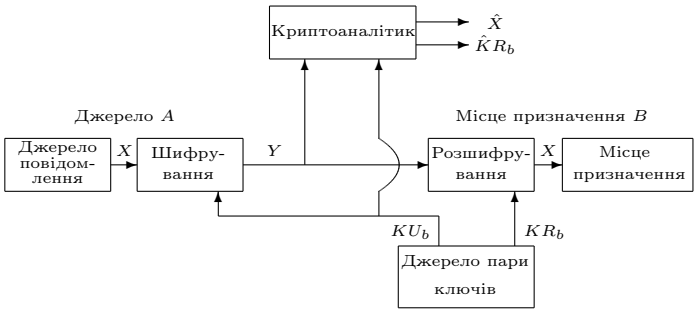



Рис. 3.4.2. Система з відкритим і таємним ключами

Елементи $X_i$ є літерами деякого скінченного алфавіту. Повідомлення, призначене для отримувача $B$, генерується разом із парою ключів: ключ відкритий $KU_b$ і ключ приватний $KR_b$. $KR_b$ є ключем таємним і його знає тільки $B$, а ключ $KU_b$ доступний кожному, отже, доступний і для $A$.

Маючи в розпорядженні повідомлення $X$ і ключ для шифрування $KU_b$, $A$ створює зашифрований текст $Y = [Y_1, Y_2, \ldots, Y_N]$: $Y = E_{KU_b}(X)$.

Отримувач, маючи в розпорядженні відкритий ключ, який відповідає приватному ключу, може знайти обернене відображення $X = D_{KR_b}(Y)$.

Зловмисник, який слідкує за $Y$ і має доступ до $KU_b$, але не має доступу до $KR_b$ або $X$, повинен намагатися отримати $X$ і/або $KR_b$. Припустимо, що супротивник знає алгоритм шифрування ($E$) і алгоритм розшифрування ($D$). Якщо його цікавить тільки одне конкретне повідомлення, то він сконцентрується на отриманні $X$ за допомогою генерації гіпотетичного відкритого тексту $X$.

Часто зловмисник на підставі попередніх повідомлень намагається отримати ключ $KR_b$, генеруючи гіпотетичний ключ $K\overline{R}_b$.

Як було сказано, коли який-небудь із двох пов'язаних ключів застосовується до шифрування, то другий можна застосувати до розшифрування. Система, зображена на рис 3.4.2, забезпечує секретність, а система на рис. 3.4.3 забезпечує шифрування з ключем відкритим та перевірку ідентичності уповноважень

$$Y = E_{KR_a}(X) \qquad X = D_{KU_a}(Y).$$

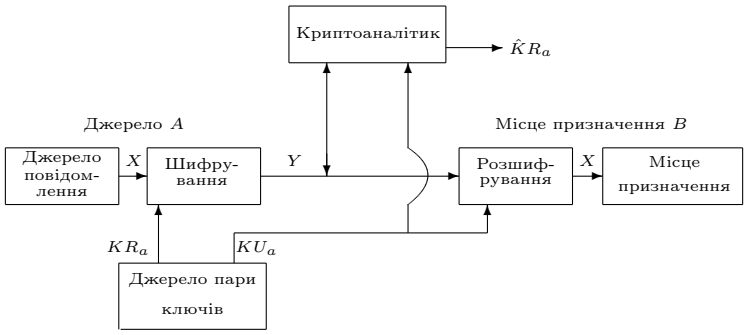



Рис. 3.4.3. Система з відкритим ключем: уповноваження

У цьому випадку $A$ виготовляє повідомлення для $B$ і висилає його за допомогою свого приватного ключа. $B$ може розшифрувати повідомлення за допомогою відкритого ключа $A$. Оскільки повідомлення зашифроване за допомогою приватного ключа $A$, то тільки $A$ може його приготувати. Таким способом реалізується функція *цифрової сигнатури (цифрового підпису)*. Більше того, не можна поміняти повідомлення, не маючи доступу до приватного ключа $A$, а отже, повідомлення є дійсним як з боку сторони, що надає, так і з боку непорушності даних.

У цій системі повідомлення зашифроване і підтверджує особу його надавача, але вимагає зберігання великого роміру даних. Кожний документ повинен зберігатися в явному вигляді для простоти практичного застосування. Крім того, потрібно також зберігати і копії у зашифрованому вигляді, щоб можна було зверифікувати їхнє походження і зміст у випадку існування сумніву. Ефективнішим способом такої перевірки є шифрування відносно невеликого блоку бітів, який є певною функцією документа. Такі блоки називаються *значенням уповноваження (автентифікатором)* і повинні мати властивість, яка не змінюється під час внесення змін у документ. Якщо вартість уповноваження зашифрована за допомогою приватного ключа надавача, то вона виконує роль сигнатури, яка підтверджує походження, зміст і самого абонента.

Підкреслимо, що описаний процес шифрування не забезпечує секретності. Повідомлення, яке пересилається, зебезпечене перед змінами, але не перед підслуховуванням. Це стає очевидним у випадку

сигнатури, яка ґрунтується на фрагменті повідомлення, оскільки решта повідомлення пересилається у явному вигляді. Навіть у випадку повного шифрування, що показано на рис. 3.4.3, немає збереження секретності, оскільки кожний може розшифрувати повідомлення, користуючись відкритим ключем надавача.

Існує можливість забезпечення уповноваження і секретності на основі подвійного застосування системи шифрування з відкритим ключем (рис. 3.4.4):

$$Z = E_{KU_b}[E_{KR_a}(X)] \quad X = D_{KU_a}[E_{KR_b}(Z)].$$

У цьому випадку, як і раніше, почнемо з шифрування повідомлення за допомогою приватного ключа його надавача. Отримуємо в такий спосіб цифрову сигнатуру. Далі шифруємо знову за допомогою ключа отримувача повідомлення. Остаточний зашифрований текст може розшифрувати тільки дійсний отримувач, який єдиний, хто має відповідний приватний ключ. Таким способом забезпечується секретність. Цей метод має такий недолік: алгоритм шифрування з відкритим ключем, необхідно використовувати чотири рази, а не два рази на кожний акт пересилання.

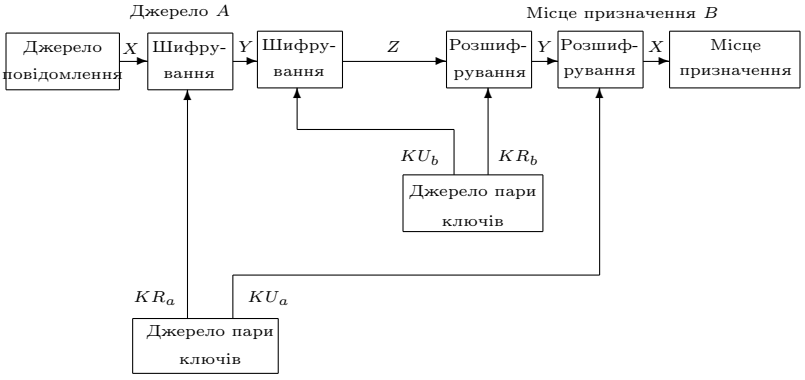



Рис. 3.4.4. Система з відкритим ключем: секретність і уповноваження

### 3.4.2. Вимоги до асиметричних систем

Системи на рис. 3.4.1 – 3.4.4 ґрунтуються на криптографічному алгоритмі, підставою якого є два, пов'язані між собою ключі. Умо-

ви, які повинні виконуватися для вказаних алгоритмів, є такими:

1. Сторона $B$ може легко обчислити пару ключів (відкритий ключ KU і приватний ключ KR).

2. Надавач $A$, знаючи відкритий ключ і повідомлення для шифрування, може легко його зашифрувати: $C = E_{KU_b}(M)$.

3. Отримувач B може легко розшифрувати отримане повідомлення за допомогою приватного ключа й отримати оригінальний текст: $M = D_{KR_b}(C) = D_{KR_b}[E_{KU_b}(M)]$.

4. Для зловмисника, який знає відкритий ключ $KU_b$, отримання ключа приватного $KR_b$ мусить бути невиконуваним.

5. Для зловмисника, який знає відкритий ключ $KU_b$ і зашифрований текст $C$, отримання оригінального тексту $M$ є невиконуваною задачею.

Можна також додати до цього шосту умову. Ця умова є життєвою, але не мусить бути необхідною до виконання для всіх застосувань шифрування з відкритим ключем.

6. Функції шифрування і розшифрування можуть бути застосовані в довільному порядку: $M = E_{KU_b}[D_{KR_b}(M)]$.

Ці умови складні для виконання, тому що вони зводяться до потреби використання односторонньої функції з додатковою інформацією:

$$\begin{array}{ll} q = f_k(p) & \text{легко обчислюється,} \\ p = f_k^{-1}(q) & \text{невиконуване.} \end{array}$$

На практиці односторонні функції з додатковою інформацією повинні задовольняти таким умовам: якщо функція $f_k$ – одностороння з додатковою інформацією, то знаходження значення

$$\begin{array}{ll} q = f_k(p) & \text{легке при відомих } k \text{ і } p, \\ p = f_k^{-1}(q) & \text{легке при відомих } k \text{ і } q, \\ p = f_k^{-1}(q) & \text{невиконуване, коли відоме } q \text{ і не відоме } k. \end{array}$$

Отже, побудова практичної системи шифрування з відкритим ключем зводиться до побудови відповідної односторонньої функції із секретом.

### 3.4.3. Рюкзачний алгоритм

Для систем шифрування з відкритим ключем запропоновано багато алгоритмів. Деякі із цих алгоритмів, які на початку виглядали абсолютно стійкими, були зламані. Розглянемо один повчальний приклад такого алгоритму.

Цим прикладом є рюкзачний алгоритм, який розробив Ральф Меркль [32]. Проблема упакування рюкзака полягає у знаходженні предметів, які повинні міститися в рюкзаку, якщо відома вага упакованого рюкзака.

Для ілюстрації проблеми розглянемо простий приклад. Нехай предмети мають таку вагу: 171 грам, 459 грамів, 197 грамів, 28 грамів, 341 грам, 2410 грамів, 1191 грам, 56 грамів, 82 грами, 2410 грамів.

Це завдання досить просте, але стає невиконуваним у обчисленнях, якщо маємо, наприклад, 100 заданих предметів, а не 10, як у даному прикладі. Меркль показав:

(1) як на основі проблеми рюкзака збудувати систему шифрування і розшифрування, а також

(2) як убудувати до неї таємну інформацію, яка дає можливість швидкого розв'язання цієї проблеми.

Для початку опишемо метод шифрування/розшифрування, який ґрунтується на проблемі упакування рюкзака. Нехай ми хочемо вислати повідомлення блоками з $n$ бітів. Визначимо:

| | | |
|---|---|---|
| вектор вмісту | $a = (a_1, a_2, \ldots, a_n)$ | $a_i$ ціле число |
| блок відкритого тексту | $x = (x_1, x_2, \ldots, x_n)$ | $x_i$ бінарне число |
| текст зашифрований | $S = a \cdot x = \sum_{i=1}^{n} (a_i \cdot x_i)$ | |

Нехай вектор вмісту $a$ буде списком елементів, які потенціально можуть міститися в рюкзаку і вага кожного елемента дорівнює вазі відповідного елемента вектора вмісту. Нехай $x$ – деякий вибір елементів вектора вмісту, де $x_i = 1$ для кожного елемента вмісту $a_i$, який вибирається для вкладення до рюкзака. Тоді добуток векторів $S$ є сумою вибраних об'єктів, які містяться в рюкзаку.

Під час шифрування використовуємо вектор $a$ як відкритий ключ. Сторона, яка хоче вислати повідомлення $x$, обчислює $S = a \cdot x$

і висилає $S$. Сторона, яка отримала повідомлення, повинна знати $x$ для заданих $S$ і $a$.

Перша умова є такою: для кожного значення $S$ існує тільки одне значення $a$, яке складає вагу рюкзака. Наприклад, вектор

$$a = (1, 3, 2, 5) \text{ і } S = 3$$

не задовльняє цій умові, оскільки задача має два розв'язки: $x = 1010$ і $x = 0100$. Елементи $a$ повинні бути вибраними так, щоб кожна комбінація елементів давала одне значення.

Друга умова: розшифрування досить складне, але стає легким за наявності спеціальної інформації. Очевидно, що для великих значень $n$, проблема рюкзака є важкою, але за деяких обставин ця проблема стає легкою для розв'язання. Припустимо, що виконується така умова: кожний елемент $a$ повинен бути більшим ніж сума попередніх вибраних елементів:

$$a_i > \sum_{j=1}^{i-1} a_j, \quad 1 < j \leq n.$$

Такий вектор швидко зростає і в такому випадку розв'язання є легким. Наприклад, вектор

$$a' = (171, 197, 459, 1191, 2410)$$

задовольняє цю умову. Припустимо, що маємо $S' = a' \cdot x' = 3798$. Оскільки $3798 > 2410$, то потрібно включити $a_5$ ($x_5 = 1$), тому що без $a_5$ інших елементів не вистачає до 3798.

Знаходимо, що $3798 - 2410 = 1388$ і оскільки ця величина більша ніж 1191, то потрібно вкласти до рюкзака $a_4$ ($x_4 = 1$). Крокуючи таким чином далі, знаходимо $x_3 = 0,\ x_2 = 1,\ x_1 = 0$.

Припустимо, що випадково вибрано рюкзачний вектор $a'$ з $n$ елементами. Виберемо два цілих взаємно простих числа $m$ і $w$, де $m$ більше від суми елементів $a'$. Тобто,

$$m > \sum_{i=1}^{n} a_i \text{ і НСД}(w, m) = 1.$$

Побудуємо тепер важкий рюкзачний вектор $a$ за допомогою множення легкого вектора $a'$ на $w$ за модулем $m$:

$$a = w \cdot a' (\mathrm{mod}\ m).$$

Вектор $a$ не буде зростати швидко і його можна використати до конструювання важких рюкзачних проблем. Знання $w$ і $m$ дає можливість конверсії важкої рюкзачної проблеми до простого вигляду. Аби в цьому переконатися, зауважимо, що оскільки $w$ і $m$ взаємно прості, то існує тільки одне $w^{-1}$ таке: що $ww^{-1} = 1$ за модулем $m$. Отже,

$$w^{-1}a = a' \ (\mathrm{mod}\ m).$$

Тепер можемо збудувати рюкзачну систему. Її складовими є такі елементи:

| | | |
|---|---|---|
| $a'$ | – швидко зростаючий вектор | (приватний, вибирається); |
| $m$ | – ціле число, більше суми $\sum_{i=1}^{n} a_i$ | (приватне, вибирається); |
| $w$ | – ціле число взаємно просте з $m$ | (приватне, вибирається); |
| $w^{-1}$ | – обернене до $w$ за модулем $m$ | (приватне, обчислюється); |
| $a$ | – яке дорівнює $w \cdot a'$ (mod $m$) | (явне, обчислюється). |

Приватний ключ складається з трійки $\{w^{-1}, m, a'\}$, а відкритим ключем є значення $a$. Припустимо, що користувач $A$ опублікував свій відкритий ключ $a$ і що користувач $B$ хоче вислати повідомлення $x$ до $A$. Подальші кроки ілюструємо прикладом.

**Приклад роботи рюкзачного алгоритму**.

**Генерація ключа**

Легка рюкзачна проблема, $a'$

| 1 | 3 | 7 | 13 | 26 | 65 | 119 | 267 |
|---|---|---|---|---|---|---|---|

Множник $w = 467$. Модуль $m = 523$. $w^{-1} = 28$ (mod 523).

Важка рюкзачна проблема, $a$

| 467 | 355 | 131 | 318 | 113 | 21 | 135 | 215 |
|---|---|---|---|---|---|---|---|

Відкритий ключ $KU = a$. Секретний ключ $KR = \{w^{-1}, m, a'\}$.

**Шифрування**

ВТ = 01001011
ШТ $= 0 \cdot 467 + 1 \cdot 355 + 0 \cdot 131 + 0 \cdot 318 + 1 \cdot 113 + 0 \cdot 21 + 1 \cdot 135 + 1 \cdot 215 = 818$

**Розшифрування**

| | | |
|---|---|---|
| $818 \cdot w^{-1}$ | $=$ | $818 \cdot 28 = 415 \pmod{523}$ |
| $415 > 267$ | $\to$ | $x_8 = 1$ |
| $415 - 267 = 148 > 119$ | $\to$ | $x_7 = 1$ |
| $148 - 119 = 29 < 65$ | $\to$ | $x_6 = 0$ |
| $29 > 26$ | $\to$ | $x_5 = 1$ |
| $29 - 26 = 3 < 13$ | $\to$ | $x_4 = 0$ |
| $3 < 7$ | $\to$ | $x_3 = 0$ |
| $3 \geq 3$ | $\to$ | $x_2 = 1$ |
| $3 - 3 = 0 < 1$ | $\to$ | $x_1 = 0$ |
| Явний текст | $=$ | $x_1x_2x_3x_4x_5x_6x_7x_8 = 01001011$. ♠ |

Знаходження $x$ для заданих $S$ і $a$ – важка задача, отже, передача повідомлення безпечна. Користувач $A$ може легко розшифрувати повідомлення. Нехай $S' = w^{-1}S \pmod{m}$. Маємо

$$S = a \cdot x = wa' \cdot x,$$
$$S' = w^{-1}S \pmod{m} = w^{-1}wa' \cdot x \pmod{m} = a' \cdot x.$$

Таким чином редукується важка проблема знаходження $x$ для заданих $S$ і $a$ до легкої проблеми знаходження $x$ для даних $S'$ і $a'$.

Рюкзачний алгоритм було оголошено як криптографічну систему, яку зламати неможливо. Упевнений у собі Меркль обіцяв нагороду 100 $ тому, хто зламає його алгоритм. Через чотири роки Аді Шамір, один із винахідників алгоритму RSA, зламав систему Меркля і отримав 100 $ нагороди.

Зазначимо, що важку рюкзачну проблему можна зробити складнішою, якщо виконати багаторазове перетворення $(w_1, m_1), (w_2, m_1)$ і т. д. Отримане таким способом перетворення не відповідає жодному однократному перетворенню пари $(w, m)$. Знаючи цей факт, Меркль підняв нагороду до 1000 $ для того, хто зламає його багатоітераційну систему. Чекати довелося два роки на виплату нагороди. Відтоді рюкзачна проблема не бралася за основну у побудові криптосистем із відкритим ключем.

## 3.5. Цифровий або електронний підпис

Розглянута вище система RSA стійка за великих значень чисел $p$ і $q$, але вона має один недолік. Якщо абонент $A$ передає пові-

домлення до $B$, користуючись відкритою лінією зв'язку абонента $B$, то зловмисник не може прочитати повідомлення, але він може вислати до $B$ зовсім інше повідомлення від імені $A$. Аби цьому запобігти, використовують складніші протоколи обміну інформацією, наприклад, такий протокол.

Нехай $A$ хоче передати $B$ повідомлення $m$. Для цього $A$ мусить мати секретний ключ $c_A$ і відкритий ключ $(d_A, n_A)$, а $B$ теж повинен мати секретний ключ $c_B$ і відкритий ключ $(d_B, n_B)$. Тоді $A$ обчислює число $e = m^{c_A} \pmod{n_A}$. Зловмисник не може цього зробити, оскільки $c_A$ секретне число. Далі $A$ обчислює число $f = e^{c_B} \pmod{n_B}$ і висилає $f$ до $B$. $B$ отримує $f$ і обчислює послідовно числа $u = f^{c_B} \pmod{n_B}$ і $w = u^{d_A} \pmod{n_A}$.

У результаті абонент $B$ отримує повідомлення $w = m$. Як і раніше, у попередній системі RSA, зловмисник не може прочитати повідомлення, але тут він не в змозі також послати повідомлення від імені $A$, оскільки не знає секретного ключа $c_A$.

В описаному протоколі виникає нова ситуація. $B$ знає, що повідомлення прийшло від $A$, тобто $A$ начебто "підписав" своє повідомлення своїм приватним ключем $c_A$. Ця ситуація є прикладом так званого **цифрового** або **електронного підпису**. Цей підпис є одним із найуживаніших винаходів сучасної криптографії.

Спочатку сформулюємо три основні умови, яким в ідеалі повинен задовольняти довільний підпис, зокрема і звичайний рукописний підпис.

1. Підписати документ може лише "законний" володар підпису і цей підпис ніхто не може підробити.

2. Автор підпису не може від нього відмовитися.

3. У випадку суперечки можлива участь третьої сторони (наприклад, суд, адвокат) для встановлення істинності підпису.

Очевидно, що цифровий підпис теж повинен задовольняти всі ці умови, тому що люди, які підписують документи і які перевіряють їх істинність, можуть знаходитися за тисячі кілометрів один від одного і можуть взаємодіяти лише за допомогою комп'ютерної мережі.

### 3.5.1. Цифровий підпис RSA

Цей підпис ґрунтується на проблемі складності розкладу цілого числа на прості множники, як і сама система RSA.

Якщо Аліса планує підписати документ, то вона повинна спочатку вибрати параметри точно так, як це вимагається в системі RSA. Для цього Аліса вибирає два великих простих числа $p$ і $q$, обчислює $n = pq$ і $\varphi(n) = (p-1)(q-1)$. Далі вибирає число $d$, взаємно просте з $\varphi(n)$, і обчислює $c = d^{-1} \pmod{\varphi(n)}$. Нарешті, вона публікує числа $n$ і $d$, наприклад, розміщуючи їх під своїм іменем на своєму сайті, і зберігає в секреті число $c$ (про числа $p, q, \varphi(n)$ можна забути, оскільки вони стають непотрібними). Тепер Аліса може ставити свій підпис на документах і повідомленнях.

Нехай Аліса хоче підписати повідомлення $m = m_1, m_2, \ldots, m_n$. Тоді вона обчислює хеш-функцію $y = h(m_1, \ldots, m_n)$, яка ставить у відповідність повідомленню $m$ число $y$. Вважається, що алгоритм обчислення хеш-функції всім відомий (про хеш-функції йтиметься пізніше). Основна властивість цієї функції полягає в тому, що практично неможливо щось змінити в основному тексті $m_1, m_2, \ldots, m_n$, не змінивши $y$. Тому на наступному кроці Алісі достатньо забезпечити підписом тільки число $y$ і цей підпис буде стосуватися до всього повідомлення $m$.

Аліса обчислює число

$$s = y^c \pmod{n}, \tag{3.14}$$

тобто вона підносить число $y$ до свого секретного степеня. Число $s$ і є цифровим підписом, який просто додається до повідомлення $m$, цим самим Аліса сформувала підписане повідомлення

$$(m, s). \tag{3.15}$$

Тепер кожний, хто знає відкриті ключі Аліси, які асоціюються з її іменем (тобто числа $n$ і $d$), можуть перевірити істинність її підпису. Для цього необхідно взяти підписане повідомлення, обчислити значення хеш-функції $h(m)$, знайти число

$$w = s^d \pmod{n} \tag{3.16}$$

і перевірити виконання умови $w = h(m)$.

**Твердження 10.** *Якщо підпис істинний, то* $w = h(m)$.

*Доведення.* Із (3.16) і (3.14) і властивостей системи RSA випливає

$$w = s^d \pmod n = y^{cd} \pmod n = y = h(m). \blacksquare$$

**Твердження 11.** *Описаний цифровий підпис задовольняє всі умови, яким повинен задовольняти підпис.*

*Доведення.* Перевіримо першу умову. Ніхто не може розкласти число $n$ на прості множники (при великому $n$, порядок якого 1024 біти, на 2022 рік задача практично не розв'язувана). Отже, знаючи $n$ і $d$, неможливо знайти $c$. Дійсно, щоб обчислити $c = d^{-1} \pmod{\varphi(n)}$, потрібно знайти $\varphi(n) = (p-1)(q-1)$, а для цього потрібно знайти прості множники $p$ і $q$. Таким чином, перша умова виконана – ніхто, крім Аліси, не може знати число $c$ і тому не в змозі підписати повідомлення.

Друга умова виконується на підставі виконання першої. Автор підпису не може від нього відмовитися, тому що ніхто інший не в змозі "сфабрикувати" підпис від його імені.

Третя умова теж очевидно виконується – у випадку конфлікту зацікавлена сторона може надати суду всі обчислення для їхньої перевірки і встановлення істини. ■

**Приклад 3.5.1.** Нехай $p = 5, q = 11$, тоді $n = 5 \cdot 11 = 55$, $\varphi(n) = 4 \cdot 10 = 40$. Нехай $d = 3$ і цей вибір правильний, оскільки НСД(3,40) = 1. Параметр $c = 3^{-1} \pmod{40}$ знаходимо за допомогою узагальненого алгоритму Евкліда, $c = 27$.

Нехай Аліса хоче підписати повідомлення $m = abbbaa$, для якого значення хеш-функції $h$ дорівнює 13, тобто

$$y = h(abbbaa) = 13.$$

У цьому випадку Аліса обчислює за формулою (3.14) число

$$s = 13^{27} \pmod{55} = 7$$

і формує підписане повідомлення $(abbbaa, 7)$. Тепер той, хто знає явні ключі Аліси $n = 55$ і $d = 3$, може перевірити істинність підпису. Отримавши підписане повідомлення, він обчислює значення хеш-функції

$$h(abbbaa) = 13$$

(якщо ніяких змін у повідомленні не було, то значення хеш-функції збігається з тим, яке обчислювала Аліса) і знаходить за допомогою формули (3.16)

$$w = 7^3 \pmod{55} = 13.$$

Значення $w$ і хеш-функції однакові, а це означає істинність підпису. ♠

### 3.5.2. Цифровий підпис Ель-Гамаля

Розглянемо варіант цифрового підпису, який ґрунтується на задачі дискретного логарифма.

Нехай, як і у схемі RSA, Аліса збирається підписати документ. Вона вибирає велике просте число $p$ і число $g$, таке, що різні степені числа $g$ є різними числами за модулем $p$. Ці числа передаються або зберігаються у відкритому вигляді і доступні всім користувачам.

Аліса вибирає випадкове число $x$, $1 < x < (p-1)$, яке стає її приватним ключем. Потім вона обчислює число

$$y = g^x \pmod{p}. \tag{3.17}$$

Це число Аліса публікує у вигляді свого відкритого ключа. Нам відомо, що при великому $p$, знаючи $y$, неможливо знайти $x$ у задачі дискретного логарифма.

Тепер Аліса може підписувати документи. Нехай вона хоче підписати повідомлення $m = m_1, m_2, \ldots, m_n$, тоді послідовність її дій є такою.

Спочатку Аліса обчислює значення хеш-функції $h(m)$, яке повинно задовольняти нерівність $1 < h < p$. Потім вона вибирає випадкове число $k$ $(1 < k < (p-1))$, взаємно просте з $p-1$, і обчислює число

$$r = g^k \pmod{p}. \tag{3.18}$$

Далі Аліса обчислює числа

$$u = (h - xr) \pmod{(p-1)}, \tag{3.19}$$

$$s = k^{-1}u \pmod{(p-1)}. \tag{3.20}$$

Число $k^{-1}$ є таким, яке задовольняє конгруенції

$$k^{-1}k \equiv 1 \pmod{(p-1)}. \tag{3.21}$$

Таке число існує на підставі взаємної простоти чисел $k$ та $p-1$ і може бути знайдене узагальненим алгоритмом Евкліда. Нарешті Аліса формує підпис повідомлення

$$(m; s, r). \tag{3.22}$$

Боб отримує підписане повідомлення (3.22) і обчислює хеш-функцію $h = h(m)$, а потім перевіряє підпис за допомогою конгруенції

$$y^r r^s \equiv g^h \pmod{p}. \tag{3.23}$$

**Твердження 12.** *Якщо підпис істинний, то умова* (3.23) *виконується.*

*Доведення.* Дійсно,

$$y^r r^s = (g^x)^r (g^k)^s = g^{xr} g^{k(k^{-1}(h-xr))} = g^{xr} g^h g^{-xr} \equiv g^h \pmod{p}.$$

Перша рівність випливає з (3.17) і (3.18), а друга з (3.20). ■

**Твердження 13.** *Описаний цифровий підпис задовольняє всі умови, яким повинен задовольнятися підпис.*

*Доведення.* Перевіримо першу умову, що ніхто, крім Аліси, не може підробити її підпис. Дійсно із (3.19) ми бачимо, що при формуванні підпису використовується секретне число $x$. Більше того, співмножник $xr$, який бере участь у формуванні підпису в (3.19), змінюється від повідомлення до повідомлення (тому що, коли $k$ випадкове, то і $r$ випадкове).

З тих самих причин, що і в системі RSA, Аліса не може відмовитися від свого підпису, оскільки тільки вона знає число $x$. Отже, виконується і друга умова підпису.

Зрозуміло, що в разі виникнення конфлікту між Алісою і Бобом, вони можуть звернутися до третьої сторони для встановлення

істини. А суддя може перевірити всі обчислення, якщо йому дадуть числа $x, m$ і $r$. ■

**Приклад 3.5.2.** Нехай спільні параметри для деякої множини користувачів вибрані такими: $p = 23,\ g = 5$. Аліса вибирає свій приватний ключ $x = 7$ і обчислює відкритий ключ $y$ за формулою (3.17):

$$y = 5^7 \pmod{23} = 17.$$

Нехай Аліса хоче вислати документ *baaaab* з підписом.

Аліса переходить до обчислення підпису за алгоритмом. Перш за все вона обчислює хеш-функцію, нехай її значення $h(m) = 3$. Після цього Аліса генерує випадкове число $k$, наприклад $k = 5$. Обчислення за формулами (3.18), (3.19) дають

$$r = 5^5 \pmod{23} = 20,$$
$$u = (3 - 7 \cdot 20) \pmod{22} = 17.$$

Далі Аліса знаходить $k^{-1}$ за модулем 22:

$$k^{-1} \pmod{22} = 5^{-1} \pmod{22} = 9.$$

Обчислення за формулою (3.20) дають значення

$$s = 9 \cdot 17 \pmod{22} = 21.$$

Нарешті Аліса формує підписане повідомлення у вигляді (3.22), тобто $(baaaab, 20, 21)$.

Підписане повідомлення передається Бобу, Боб його отримує і перевіряє істинність підпису. Спочатку він обчислює значення хеш-функції $h(baaaab) = 3$, а потім обчислює ліву і праву частину за формулою (3.23), тобто

$$17^{20}20^{21} \pmod{23} = 16 \cdot 15 \pmod{23} = 10 \quad \text{і} \quad 5^3 \pmod{23} = 10.$$

На підставі цього Боб робить висновок, що підпис істинний. ♠

Розглянутий метод електронного підпису складніший, ніж RSA, але його стійкість ґрунтується на іншій односторонній функції. Ця обставина досить важлива у криптографії, оскільки в разі компрометації одного методу, його можна замінити другим. Крім того, на основі алгоритму Ель-Гамаля можна побудувати ефективніший алгоритм, в якому час обчислень значно скорочується за рахунок "коротких" показників степеня.

Зауважимо, що електронний підпис займає настільки важливе місце у криптографії, що для нього розроблено декілька стандартів як у США, так і в інших країнах.

### 3.5.3. Криптографічні хеш-функції

У розглянутих системах цифрового підпису важливу роль відіграють хеш-функції, за значеннями яких установлюється оригінальність підпису та самого повідомлення. Сформулюємо точніше поняття хеш-функції, основні криптографічні вимоги до таких функцій та способи їхнього обчислення.

**Означення 75.** *Хеш-функцією називається довільна функція* $y = h(x_1x_2\cdots x_n)$*, яка слову* $x_1x_2\cdots x_n$ *довільної довжини* $n$ *ставить у відповідність ціле число фіксованої довжини.*

Прикладом хеш-функції може служити контрольна сума повідомлення, яка має вигляд

$$h(x_1x_2\cdots x_n) = (x_1 + x_2 + \cdots + x_n) \pmod{2^w},$$

де $w$ – довжина машинного слова (32 або 64 біти). Для цієї функції довжина її значення складає $w$ бітів незалежно від довжини повідомлення. Контрольні суми часто використовують для виявлення ненавмисних помилок у повідомленні (зміна одного символу в повідомленні веде до зміни значення контрольної суми). Але таку хеш-функцію можна легко "обманути", вводячи навмисно помилку в повідомлення і зберігши значення контрольної суми. Якби така хеш-функція використовувалася для формування цифрового підпису, то неважко було б змінити зміст підписаного повідомлення. Тому розглянута функція не може застосовуватися у криптографічних системах.

Основні умови, які повинні задовольняти криптографічні хеш-функції, зводяться до таких: нехай $p$ – деяке слово (повідомлення) в алфавіті $X$, тоді

1) для довільного заданого $p$ обчислення $y = h(p)$ повинно виконуватися швидко;

2) якщо відоме $y$, то обчислення, пов'язані зі знаходженням $p$, для якого $y = h(p)$, у розумному проміжку часу виконати неможливо;

3) якщо відоме $p$, то знайти інше значення $p \neq p'$, таке, що $h(p') = h(p)$, важка задача;

4) важко знайти пару різних повідомлень $p$ і $p'$, для яких $h(p) = h(p')$.

Зазначимо, що перша умова повинна виконуватися завжди, інакше хеш-функція втрачає будь-яке практичне значення. Решта умов важливі залежно від застосувань. Наприклад, якщо паролі входу в систему зберігаються у вигляді значень відповідних їм хеш-функцій, то ці хеш-функції повинні задовольняти другу вимогу. У схемі електронного підпису важливе виконання третьої умови, а четверта умова актуальна у криптографічних протоколах. Очевидно, що коли виконується четверта умова, то третя умова виконується автоматично.

Побудова хеш-функцій, які задовольняють усі чотири умови, досить складна задача. Нині розроблено і практично використовуються хеш-функції, які вважають такими, що відповідають вищенаведеним умовам (хоча строго доведення цього факту немає).

Розглянемо універсальний спосіб побудови хеш-функцій на основі блокових шифрів, який має практичний інтерес, не дивлячись на те, що побудовані хеш-функції не є такими, які швидко обчислюються.

Нехай дано блоковий шифр $E$, який для заданого блока $X$ і ключа $K$ формує шифротекст $Y$, тобто $Y = E_K(X)$.

Розглянемо два алгоритми, для яких довжина слова, отриманого у вигляді значення хеш-функції, дорівнює розміру блока в шифрі. Зауважимо, що відомі конструкції, які дозволяють отримувати хеш-функції з довжинами слів, що кратні розміру блока.

У першому алгоритмі повідомлення спочатку подається у вигляді послідовності блоків $X_1, X_2, \ldots, X_n$. Останній блок, за потреби, доповнюється нулями, а деколи в останній блок приписують довжину повідомлення у вигляді двійкового числа. Значення хеш-функції $h$ дістаємо в результаті виконання такого ітеративного процесу:

$h := 0;$
for $i = 1, 2, \ldots, n$ do
    $h := E_h(X_i) \oplus X_i.$

За початкове значення $h$ можна вибирати не тільки нуль, а якенебудь інше число, але це не має принципового значення. У даному алгоритмі значення $h$, отримане на попередній ітерації, використовується як ключ шифру в наступній ітерації. Тому неявно вважається, що довжина ключа в шифрі дорівнює довжині блока. Але довжина ключа може значно перевищувати розміри блока. У таких випадках ефективнішим є другий алгоритм.

У цьому алгоритмі повідомлення спочатку подається у вигляді послідовності $X_1, X_2, \ldots, X_n$, в якій розмір кожного елемента дорівнює довжині ключа в шифрі. Останній елемент заповнюється так само, як і в першому алгоритмі. Значення хеш-функції $h$ обчислюється таким чином:

$h := 0;$
for $i = 1, 2, \ldots, n$ do
$\quad h := E_{X_i}(h) \oplus h.$

Тут елементи повідомлення вже виконують роль ключів у шифрі.

Описані алгоритми обчислення хеш-функцій задовольняють усі чотири умови, які ставляться до криптографічних хеш-функцій, за умови стійкості використовуваних блокових шифрів [31].

## 3.6. Криптографічні протоколи

Розглянуті в попередніх розділах криптографічні методи часто використовують як інструменти для розв'язання практичних задач. Вони дають можливість розв'язувати проблеми, які раніше вважалися нерозв'язуваними, і тепер вони використовуються в реальних комп'ютерних системах. Прикладами їхнього використання є оформлення комерційних угод у режимі віддаленої взаємодії учасників, грошові розрахунки, проведення виборів тощо. Методи розв'язання подібних задач, як правило, описують у вигляді *криптографічних протоколів*, приклади яких були розглянуті у вступі

(див. протоколи 1 і 2).

Загальноприйнятою криптографічною технікою в задачах такого типу є шифрування кожного індивідуального повідомлення окремим ключем. З умов, які повинна задовольняти цілком таємна криптосистема, такий ключ повинен бути одноразовим. Це також пов'язано з тим, що час існування одноразового ключа визначається часом сеансу зв'язку.

З попередніх підрозділів випливає, що обмін секретною інформацією можна виконувати шляхом використання симетричних і асиметричних криптосистем. Розглянемо коротко протоколи обміну ключами за допомогою однієї і другої систем.

**Обмін ключами за допомогою симетричної криптосистеми**. Протокол такого обміну припускає, що Аліса і Боб отримали секретний ключ від Джона. Перед початком виконання протоколу ці ключі повинні бути в обох користувачів і Єва (зловмисник) не має про них жодної інформації. У цьому протоколі ігнорується дуже важлива проблема доставки секретних ключів.

*Протокол обміну:*

1) Аліса звертається до Джона з вимогою згенерувати їй одноразовий ключ для зв'язку з Бобом.

2) Джон генерує випадковий одноразовий ключ; зашифровує дві копії цього ключа – одну для Аліси, а другу для Боба і висилає обидві копії Алісі.

3) Аліса розшифровує свою копію одноразового ключа.

4) Аліса висилає Бобу його копію одноразового ключа.

5) Боб роозшифровує свою копію одноразового ключа.

6) Аліса і Боб мають спільний одноразовий ключ для обміну інформацією.

Цей протокол повністю ґрунтується на надійності Джона, для ролі якого більше підходить комп'ютерна програма, що заслуговує на довіру, ніж людина, яка заслуговує на довіру. Зрозуміло, що коли Єва отримає доступ до Джона, то скомпрометованою виявиться вся мережа. Оскільки в її розпорядженні будуть усі секретні ключі, що були згенеровані Джоном користувачам. Їй залишиться тільки підключитися до ліній зв'язку і перехоплювати потік зашифрованих повідомлень.

Крім того, Джон у такій мережі є потенційно вузьким місцем, тому що коли з ним щось трапиться, то це розвалить всю систему в цілому.

**Обмін ключами за допомогою асиметричної криптосистеми**. У такій системі Аліса і Боб використовують криптографію з відкритим ключем для узгодження одноразового ключа, а потім обмінюються інформацією за допомогою цього ключа. Якщо ключі Аліси і Боба доступні в деякій базі розподілу ключів, то це значно полегшує роботу протоколу.

*Протокол обміну:*

1) Аліса одержує відкритий ключ Боба з бази розподілу ключів.

2) Аліса генерує випадковий одноразовий ключ, зашифровує його відкритим ключем Боба і висилає його Бобу.

3) Боб розшифровує повідомлення Аліси за допомогою свого таємного ключа.

4) Аліса і Боб шифрують свої повідомлення цим одноразовим ключем.

Такий протокол значно кращий попереднього, оскільки Аліса може безпечно посилати Бобу повідомлення, навіть якщо Боб ніколи не чув про Алісу.

Далі будуть розглянуті деякі уточнення цих і подібних до них протоколів та проблеми, які в них виникають.

### 3.6.1. Електронні гроші

Нині все більша і більша кількість людей у різних країнах світу роблять закупівлі в магазинах за допомогою мережі інтернет, замовляють квитки, виконують розрахунки за допомогою електронних карток. Оскільки інформація про закупи залишається в магазинах і банках, то анонімність процесу закупу зникає. Ця ситуація не є приємною для покупця і він бажав би залишатися особою невідомою в такому процесі. Тому виникає проблема побудови таких схем платежів, які зберігали б анонімність покупця так само, як і при розрахунках готівкою. Такого типу схеми дістали назву *електронних протоколів* або *цифрових грошей*, які призвані забезпечувати такий ступінь анонімності, як і звичайні гроші.

**Постановка задачі**. У протоколі задіяно три учасники: *банк, покупець* і *магазин*. Покупець і магазин мають відповідні рахунки в банку. Покупець хоче придбати деякий товар у магазині. Покупка реалізується у вигляді триступеневого процесу:

1) покупець знімає потрібну суму зі свого рахунку в банку;

2) покупець перераховує гроші в магазин;

3) магазин повідомляє про це в банк, відповідна сума грошей перераховується на рахунок магазину, а покупець отримує товар або цей товар йому доставляють.

**Необхідно розробити** таку схему, щоб

а) вона була надійна;

б) банк не знав, хто купив товар, тобто анонімність має бути збережена.

*Перша схема* ґрунтується на системі RSA і не є найкращою (що буде встановлено далі).

Банк має таку інформацію: секретні числа $p, q, c$ і відкриті числа

$$n = p \cdot q, \quad c = d^{-1} \pmod{\varphi(n)}. \tag{3.24}$$

Нехай покупець вирішив використати певну завчасно замовлену в банку суму (наприклад 100\$). Розглянемо спочатку випадок, коли може використовуватися лише одна банкнота вартістю 100\$. Покупець висилає в банк число $m$, яке буде номером банкноти (це повинно бути число з інтервалу $[2, n-1]$.

Банк обчислює число

$$s = m^c \pmod{n} \tag{3.25}$$

і формує банкноту $(m, s)$, яку повертає покупцеві, зменшивши перед цим його рахунок на 100\$. Параметр $s$ у банкноті – це електронний підпис банку. Його ніхто не може підробити, оскільки число $c$ секретне.

Покупець показує банкноту $(m, s)$ у магазині, для того щоб купити товар. Магазин відправляє банкноту в банк для перевірки. Спочатку банк перевіряє правильність підпису (цю перевірку міг би виконати й магазин, користуючись відкритими ключами банку). Але, крім цього, банк зберігає в списку всі номери повернених йому банкнот і перевіряє, чи немає числа $m$ у цьому списку. Якщо $m$ у списку є, то платіж не приймається (хтось намагається використати банкноту повторно) і банк повідомляє про це магазин. Якщо ж усі перевірки пройшли успішно, то банк перераховує на рахунок магазину 100\$, а магазин відпускає товар покупцеві.

Недолік цієї схеми полягає в тому, що відсутня анонімність. Банк, а також всі, хто має доступ до відкритих ліній зв'язку, можуть запам'ятати, якому покупцеві відповідає число $m$, і тим самим зрозуміти, хто купив товар.

Розглянемо *другу схему*, яка вже забезпечує анонімність. Ця схема ґрунтується на так званому "сліпому підписі".

Знову покупець хоче купити товар. Він генерує число $m$, яке тепер не висилається в банк. Далі він генерує випадкове число $r$, взаємно просте з $m$, і обчислює число

$$\hat{m} = (m \cdot r^d) \pmod{n}. \tag{3.26}$$

Число $\hat{m}$ покупець відправляє в банк.

Банк обчислює число

$$\hat{s} = \hat{m}^c \pmod{n} \tag{3.27}$$

і відправляє $\hat{s}$ назад покупцеві (знявши перед цим 100\$ з його рахунку).

Покупець знаходить число $r^{-1} \pmod{n}$ і обчислює

$$s = (\hat{s} \cdot r^{-1}) \pmod{n}. \tag{3.28}$$

Беручи до уваги співвідношення (3.28), (3.27) і (3.25), дістаємо

$$s = \hat{m}^c \cdot r^{-1} = (m \cdot r^d)^c \cdot r^{-1} = m^c r^{dc} \cdot r^{-1} = m^c r r^{-1} = m^c \pmod{n},$$

тобто покупець отримав підпис банку до $m$, але самого числа $m$ ні банк, ні хто-небудь інший не бачив. Обчислення (3.28) називається "сліпим підписом", оскільки реальне повідомлення $m$ підписант не бачив і взнати не може.

Таким чином, покупець має число $m$, яке нікому не відоме і ніколи не передавалося каналами зв'язку, і підпис банку $s$, який збігається з числом, знайденим за формулою (3.26). Покупець формує банкноту $(m, s)$ і діє так само, як у попередній схемі. Але тепер ніхто не знає, кому належить ця банкнота, тобто вона стала анонімною, як звичайна паперова банкнота.

Дії магазину і банку після того, як покупець пред'явив банкноту $(m, s)$, нічим не відрізняються від дій, описаних у першій схемі.

Описана схема має недолік, який полягає в тому, що можна сфабрикувати фальшиву банкноту, якщо відомі принаймні дві не фальшиві банкноти. Робиться це так. Нехай зловмисник (чи то покупець, чи магазин) має дві не фальшиві банкноти $(m_1, s_1)$ і $(m_2, s_2)$. Тоді він може виготовити фальшиву банкноту $(m_3, s_3)$, обчисливши числа

$$m_3 = m_1 m_2 \pmod{n}, \quad s_3 = s_1 s_2 \pmod{n}.$$

Дійсно,

$$m_3^{\mathrm{c}} = (m_1 m_2)^{\mathrm{c}} = m_1^c m_2^c = s_1 s_2 = s_3 \pmod{n}, \tag{3.29}$$

тобто $s_3$ є правильним підписом для $m_3$, і банк не має жодних підстав, щоб не приймати цю фальшиву банкноту (він її не може відрізнити від не фальшивої). Це так звана "мультиплікативна властивість" системи RSA.

Розглянемо тепер *хорошу схему*, яка не має недоліків попередніх схем. В одному варіанті такої схеми використовується деяка одностороння функція

$$f : \{1, \ldots, n\} \to \{1, \ldots, n\}.$$

Функція $f$ не секретна і відома всім (покупцеві, банку і магазину).

Банкнота тепер визначається як пара чисел $(m, s_f)$, де

$$s_f = (f(m))^c \pmod{n},$$

тобто підписується не $m$, а значення $f(m)$.

Покупець генерує число $m$ (і тримає його в секреті), обчислює $f(m)$, підписує в банку за допомогою сліпого підпису число $f(m)$ і формує банкноту $(m, s_f)$. Ця банкнота має всі хороші властивості, як і в другій схемі, але в той час підробити таку банкноту неможливо, оскільки неможливо обчислити $f^{-1}$. Для перевірки правдивості підпису (тобто оригінальності банкноти) потрібно обчислити $f(m)$ і переконатися, що $s_f^d = f(m) \pmod{n}$.

Як зазначалося в попередньому підрозділі, під час вибору односторонньої функції необхідно вибирати ту, яка не має мультиплікативної властивості. На практиці такими функціями є криптографічні хеш-функції.

Решта дій магазину і банку залишаються такими самими, як і в попередніх схемах.

Ще *одним із простих способів боротьби з мультиплікативністю* функції в системі RSA є внесення надлишковості до повідомлення. Нехай довжина модуля $n$ складає 1024 біти. Такою може бути і довжина числа $m$. Будемо записувати випадково вибраний номер банкноти тільки в молодші 512 бітів числа $m$, а у старші 512 бітів числа $m$ запишемо деяке фіксоване число. Це число может нести корисну інформацію, таку, як номінал банкноти і назва банку (за допомогою 512 бітів можна подати рядок із 64 символів коду ASCII). Тепер банк при пред'явленні йому банкноти буде обов'язково перевіряти наявність фіксованого заголовка в параметрі $m$ і не приймати банкноту за його відсутності. Імовірність того, що при множенні двох чисел за модулем $n$ їхній добуток збігатиметься з ними, в 512 бітах дуже мала. Тому отримати фальшиву банкноту за формулами (3.29) практично неможливо.

**Приклад 3.6.1.** Нехай секретними параметрами банк вибрав числа $p = 17, q = 7$, $c = 77$. Відповідні їм відкриті параметри будуть $n = 119, d = 5$. Для виключення можливості підробки банкнот їх допустимими номерами вважатимуться тільки числа, які складаються з двох однакових десяткових цифр, наприклад 11, 22, 33, 44, 55, 66, 77, 88, 99.

Коли покупець хоче отримати банкноту, він спочатку випадковим чином вибирає її номер із числа допустимих номерів. Нехай він вибрав $m = 33$. Потім він знаходить випадкове число $r$, взаємно просте зі 119. Нехай це буде число $r = 67$, НСД(67,119) = =1. Далі покупець обчислює число

$$\hat{m} = (33 \cdot 67^5) \pmod{119} = (33 \cdot 16) \pmod{119} = 52.$$

Число 52 посилається в банк.

Банк знімає з рахунку покупця 100$ і відправляє йому число

$$\hat{s} = 52^{77} \pmod{119} = 103.$$

Покупець обчислює число $r^{-1} = 67^{-1} \pmod{119} = 16$ і $s = 103 \cdot 16 \pmod{119} = 101$ і отримує банкноту $(m, s) = (33, 101)$. Цю банкноту він приносить або висилає в магазин, щоб купити товар.

Магазин пред'являє банкноту в банк. Банк перевіряє чи

1) номер банкноти $n = 33$ складється з двох однакових десяткових цифр, тобто має необхідну надлишковість;

2) раніше банкнота з таким номером не пред'являлась;

3) підпис банка правильний, тобто $33^5 = 101 (\mathrm{mod}\ 119)$.

Якщо всі перевірки успішно пройдено, то банк зараховує 100$ на рахунок магазина, про що його інформує. Магазин відпускає товар покупцеві. ♠

Розглянемо ще дві проблеми, які виникають у зв'язку з розглянутою схемою електронних грошей.

У цій схемі незалежно діють покупці і, навіть, один покупець, який не пам'ятає номерів раніше використаних ним банкнот, може випадково згенерувати дві або більше банкнот з однаковими номерами. За умовами протоколу, банк прийме тільки одну із банкнот, яка буде пред'явлена першою. Але, беручи до уваги розміри чисел, які фігурують у протоколі, і те, що ці числа генеруються випадковим чином, то ймовірність отримання двох однакових номерів надзвичайно мала.

Друга проблема полягає в тому, що в розглянутій схемі використовують банкноти лише одного фіксованого номіналу. А це, звичайно, незручно покупцеві при розрахунках. Розв'язання проблеми використання банкнот різних номіналів може бути таким. Банк заводить декілька пар чисел $(c_i, d_i)$, які мають властивість (3.24), і повідомляє, що $d_1$ відповідає, наприклад, 100 гривням, $d_2$ – 50 гривням і т. д. Коли покупець запитує сліпий підпис у банку, він додатково повідомляє, якого номіналу банкноту він хоче отримати. Банк знімає з його рахунку вказану покупцем суму і формує підпис, за допомогою секретного числа $c_i$. Коли банк отримує підписану банкноту, він використовує для перевірки підпису по черзі числа $d_1, d_2$ і т. д. Якщо підпис виявився правильним для якогось $d_i$, то приймається банкнота $i$-го номіналу. Якщо ж параметр $m$ банкноти включає заголовок з указанням номіналу, то задача перевірки підпису спрощується і банк зразу використовує потрібний ключ $d_i$.

### 3.6.2. Взаємна автентифікація

**Автентифікація** – це процедура, яка дозволяє одному об'єкту перевірити задані властивості іншого об'єкта. Наприклад, автен-

тифікація дає можливість першому об'єкту перевірити законність другого об'єкта, а другому об'єкту – довести свою законність. Отже, у процесі автентифікації взаємодіють два об'єкти, а процедура такої взаємодії називається *протоколом автентифікації*.

Автентифікація поділяється на три види:

- автентифікація джерела даних;
- автентифікація об'єкта;
- автентифікація ключів.

Перший вид автентифікації означає перевірку заданих властивостей повідомлення, другий зосереджується на перевірці достовірності відомостей про відправника повідомлення, а третій призначається для організації захищеного каналу зв'язку для обміну секретними повідомленнями.

Нижче аналізується приклад криптографічного протоколу автентифікації Діффі-Хеллмана, у результаті реалізації якого два об'єкти $A$ і $B$ взаємно ідентифікують один одного і формують спільний секретний ключ, який далі використовується для шифрування повідомлень. Насправді об'єктами $A$ і $B$ можуть виступати користувач і комп'ютерна система або дві різні комп'ютерні системи, але сенс наведеного нижче протоколу від цього не змінюється.

У сучасних мережах трансмісії даних, наприклад в інтернеті, інформація від одного користувача до іншого проходить через велику кількість проміжних вузлів (маршрутизатори, фільтри, шлюзи, поштові сервери тощо), які не контролюються користувачем. Отже, зловмисник може поселитися на одному з таких вузлів і не тільки прослуховувати іформацію, а й додавати, змінювати або вилучати повідомлення.

Розглянемо типову атаку на систему Діффі-Хеллмана в мережі зв'язку з активним зловмисником. Аліса вибирає своє секретне число $X_A$ і посилає Бобу $g^{X_A}$. Боб вибирає своє секретне число $X_B$ і посилає Алісі $g^{X_B}$. Але зловмисник Єва перехоплює ці числа і висилає замість них Алісі і Бобу $g^{X_E}$, де $X_E$ – її число. Усі ці числа виглядають зовсім як випадкові, тому ні Аліса, ні Боб нічого не підозрюють. У результаті Аліса формує ключ $K_A = g^{X_E X_A}$, а Боб – ключ $K_B = g^{X_E X_B}$. Обидва ключі можуть бути легко обчислені

Євою. Тепер, коли Аліса посилає Бобу повідомлення, зашифроване ключем $K_A$, Єва розшифровує повідомлення, заново шифрує його ключем $K_B$ і відправляє Бобу. Аналогічно Єва діє і при передачі повідомлень у зворотному напрямку. Боб і Аліса взаємодіють, як їм здається, у захищеному режимі, але насправді Єва читає всі їхні повідомлення.

Подібного типу атака стає неможливою, якщо Аліса і Боб не передають відкриті ключі (у системі Діффі-Хеллмана це числа $Y_A = g^{X_A}$ і $Y_B = g^{X_B}$) каналом зв'язку, а вибирають їх із деякої таблиці чи довідника, який був отриманий ними завчасно з "надійного" джерела.

Взагалі, більшість криптосистем з відкритими ключами вимагають наявності деякої організаційної структури, що займається сертифікацією ключів. Така структура може, наприклад, виглядати таким чином. У мережі, до якої належать Аліса і Боб, існує "чесний" користувач Джон (об'єкт $D$), який зацікавлений лише в тому, щоб мережа надійно функціонувала (наприклад, це може бути не людина, а комп'ютер, який надійно охороняється і працює за чітко зафіксованою програмою). Джон має в розпорядженні яку-небудь надійну криптосистему (наприклад RSA) з відповідними відкритими ключами і виконує дві операції:

– додає у свою базу даних інформацію про відкритий ключ користувача, яка присилається йому у вигляді повідомлення і яка зашифрована за допомогою відкритого ключа Джона;

– повідомляє про чий-небудь відкритий ключ, підписаний його підписом.

Відкриті ключі Джона доводяться до відома всіх користувачів яким-небудь способом, що не дає можливості втрутитися Єві. Наприклад, вони публікуються у вигляді рекламного повідомлення в газеті. Тепер Аліса, обчисливши свій відкритий ключ, формує повідомлення від свого імені і цього ключа, шифрує його за допомогою відкритого ключа Джона і висилає Джону (ніхто, крім Джона, не в змозі розшифрувати це повідомлення). Боб, коли йому потрібний відкритий ключ Аліси, висилає запит Джону, і Джон присилає йому підписаний ключ Аліси (ніхто не може підробити підпис Джона).

Боб перевіряє підпис Джона, користуючись його відкритим ключем, і приймає ключ Аліси як достовірний. Таким чином, кожний користувач мережі дістає достовірну інформацію про відкриті ключі інших користувачів, і Єва не може втрутитися в цей процес.

Отже, якщо Аліса і Боб користуються достовірними відкритими ключами, то схема Діффі-Хеллмана розв'язує задачу автентифікації секретного ключа. Але вона безпосередньо не забезпечує автентифікацію користувачів. Дійсно, якщо замість Аліси виступала б Єва, яка не знає секретного ключа Аліси, то у них із Бобом будуть сформовані різні секретні ключі. Така ситуація може стати зрозумілою пізніше, на стадії обміну даними, коли Боб не зможе розшифрувати передане йому повідомлення або виявить, що Аліса не розуміє того, що він їй посилає. Часто необхідно забезпечити явну автентифікацію, щоб після завершення протоколу сторони точно знали, хто є хто.

У схемі Діффі–Хеллмана є і другий недолік: секретний ключ, який формує Аліса і Боб, буде завжди одним і тим доки вони не поміняють свої відкриті ключі. Але заміна відкритих ключів є відносно довгим процесом, оскільки для цього потрібно повідомити всіх користувачів мережі про зміну деякого відкритого ключа, щоб вони змогли скорегувати інформацію у своїх довідниках. Бажано було б мати такий протокол, який забезпечував би оперативне утворення кожного разу випадково вибраних секретних ключів.

Розв'язання полягає у використанні якого-небудь шифра із відкритим ключем для передачі секретних ключів. Нехай $P_A(x)$ шифр з відкритим ключем користувача $A$, яким зашифровано повідомлення $x$. Наприклад, $P_A(x)$ може бути шифром RSA або шифром Ель-Гамаля. У випадку шифру RSA $P_A(x) = x^{d_A} \pmod{n_A}$, де $d_A$ і $n_A$ – відкритий ключ користувача $A$. Всі, хто знає відкритий ключ користувача $A$, можуть обчислити $P_A(x)$ для повідомлення $x$. Але тільки $A$, який знає відповідний секретний ключ, може отримати $x$ із $y = P_A(x)$. Нехай $P_B(x)$ означає шифр, побудований за допомогою відкритого ключа користувача $B$. Нехай символ $||$ означає конкатенацію чисел. Опишемо протокол поетапно.

Для розв'язання задачі, в якій Аліса і Боб хочуть взаємно іден-

тифікувати один одного і встановити спільний секретний ключ, розглянемо такий не зовсім хороший протокол.

**Крок 1**. Аліса вигадує секретний ключ $k_1$, шифрує його, використовуючи відкритий ключ Боба, і посилає Бобу

$$A \to B: \ P_B(k_1). \tag{3.30}$$

**Крок 2**. Боб розшифровує $k_1$, знову шифрує його, використовуючи відкритий ключ Аліси, і посилає Алісі:

$$B \to A: \ P_A(k_1). \tag{3.31}$$

**Крок 3**. Аліса розшифровує $k_1$ і порівнює його з вибраним на кроці 1.

Що ж дістаємо в результаті реалізації цього протоколу?

По-перше, Аліса і Боб отримали спільний секретний ключ $k_1$, невідомий Єві (вона не може розшифрувати ні $P_B(k_1)$, ні $P_A(k_1)$).

По-друге, Аліса отримала криптографічно стійку автентифікацію Боба, оскільки ніхто крім нього не зміг би розшифрувати $k_1$. Але в цьому протоколі Боб не отримує жодної автентифікації Аліси (повідомлення (3.30) міг би послати хто завгодно). Він міг би навести симетричний протокол зі свого боку:

$$B \to A: \ P_A(k_2), \quad A \to B: \ P_B(k_2)$$

і отримати таку автентифікацію. Але тут проблема полягає в тому, що немає гарантії, що обидва протоколи виконуються одними і тими ж учасниками.

Але є ще й інша проблема. Аліса може використати описаний протокол для злому криптосистеми Боба. Дійсно, припустимо, що Аліса перехопила якесь повідомлення, яке призначалося Бобу, тобто $y = P_B(x)$. Вона прикидається, що хоче увійти в систему Боба, і запускає протокол (3.30), (3.31). Але замість $P_B(k_1)$ вона передає Бобу повідомлення $y$. Оскільки число $k_1$ вибирається довільним чином, то Боб нічого не запідозрить. Він чесно виконує свій крок у протоколі і розшифровує для Аліси $x$.

Звідси випливає, що ніколи ні для кого не варто розшифровувати випадкові числа!!! Це може зашкодити вашій безпеці. Засобом боротьби з такою небезпекою є внесення певної надлишковості

в повідомлення, наприклад, уведення якого-небудь елемента, який відомий отримувачу повідомлення і який він очікує. Зокрема, Аліса могла б побудувати повідомлення, відводячи 512 бітів для випадкового числа $k_1$ і 512 бітів для свого імені, адреси, фрагмента відкритого ключа й іншу інформацію, яку легко перевірити. Якщо $\hat{A}$ означає таку інформацію, то вона посилає Бобу $P_A(k_1||\hat{A})$. У цьому випадку Боб не став би посилати Алісі повідомлення $x$, оскільки відповідні 512 бітів напевно не включали б $\hat{A}$.

Усі перераховані проблеми розв'язує протокол Нідхама-Шредера [31].

**Крок 1**. Аліса вибирає випадкове число $k_1$, об'єднує його зі своєю відкритою інформацією $\hat{A}$ і посилає Бобу

$$A \to B: \ P_B(k_1||\hat{A}). \tag{3.32}$$

**Крок 2**. Боб розшифровує (3.32) і переконується в тому, що повідомлення включає відкриту інформацію Аліси $\hat{A}$. Він вибирає випадкове число $k_2$, об'єднує його з $k_1$ і посилає Алісі:

$$B \to A: \ P_A(k_1||k_2). \tag{3.33}$$

**Крок 3**. Аліса розшифровує (3.33) і переконується в тому, що отримане повідомлення включає $k_1$. Це для неї є надійною ознакою автентифікації Боба, адже ніхто інший не зміг би знайти $k_1$ із (3.32). Аліса посилає Бобу

$$A \to B: \ P_B(k_2). \tag{3.34}$$

**Крок 4**. Боб розшифровує (3.34) і переконується в тому, що він отримав $k_2$. Це для нього є надійною ознакою автентифікації Аліси, оскільки ніхто інший не зміг би знайти $k_2$ із (3.33).

Тепер Аліса і Боб можуть сформувати спільний ключ із $k_1$ і $k_2$, наприклад, покласти $k = k_1 \oplus k_2$, де $\oplus$ операція додавання бітів за модулем 2.

**Приклад 3.6.2.** Нехай у мережі використовується шифр Ель-Гамаля з відкритими параметрами $p = 107$, $g = 2$. Користувачі $A$ і $B$ мають відкриті ключі $d_A = 58$, $d_B = 28$, яким відповідають секретні ключі $c_A = 33$, $c_b = 45$. Далі, нехай додаткова інформація має вигляд $\hat{A} = 1$, $\hat{B} = 2$ і секретний ключ будемо будувати десятковими цифрами.

На першому кроці протоколу $A$ вибирає секретний ключ, наприклад $k_1 = 3$, і формує повідомлення $m = k_1||\hat{A} = 31$. Це повідомлення шифрується шифром Ель-Гамаля з відкритим ключем користувача $B$ (з параметром шифру $k$):

$$k = 15,\ r = g^k \pmod p = 2^{15} \pmod{107} = 26,$$
$$e = m \cdot d_B^k \pmod p = 31 \cdot 28^{15} \pmod{107} = 47.$$

Пара чисел (26, 47) утворює ту шифрограму, яку необхідно вислати $B$. У позначеннях протоколу, які були введені вище, $P_B(k_1||\hat{A}) = (26{,}47)$.

На другому кроці протоколу $B$ розшифровує (26,47), використовуючи свій секретний ключ: $m' = e \cdot r^{p-1-c_B} \pmod p = 47 \cdot 26^{106-45} \pmod{107} = 31$.

$B$ переконується, що молодша цифра включає автентифікаційний номер користувача $A$ і дістає $k_1 = 3$. Потім він вибирає своє секретне число $k_2 = 7$, формує повідомлення $m = k_1||k_2 = 37$ і шифрує його за допомогою відкритого ключа $A$:

$$k = 77,\ r = g^k \pmod p = 2^{77} \pmod{107} = 63,$$
$$e = m \cdot d_A^k \pmod p = 37 \cdot 58^{77} \pmod{107} = 18.$$

Пара чисел (63,18) – це те, що потрібно вислати $A$, тобто $P_A(k_1||k_2) = (63, 18)$ і

$$B \to A:\ (63, 18).$$

На третьому кроці $A$ розшифровує (63, 18):

$$m' = e \cdot r^{p-1-c_A} \pmod p = 18 \cdot 63^{106-33} \pmod{107} = 37.$$

$A$ переконується в тому, що старша цифра включає $k_1 = 3$, і дістає $k_2 = 7$. Тепер $A$ шифрує $k_2$ для $B$:

$$k = 41,\ r = g^k \pmod p = 2^{41} \pmod{107} = 82,$$
$$e = m \cdot d_B^k \pmod p = 7 \cdot 28^{41} \pmod{107} = 49,$$

і висилає до $B$ пару чисел (82,49).

На четвертому кроці $B$ розшифровує (82,49):

$$m' = e \cdot r^{p-1-c_B} \pmod p = 49 \cdot 82^{106-45} \pmod{107} = 7.$$

$B$ переконується в тому, що він отримав число $k_2 = 7$.

Тепер $A$ і $B$ можуть сформувати спільний ключ за завчасно оговореною схемою, наприклад, $k = k_1 \oplus k_2 \pmod 2 = 3 \oplus 7 = (011) \oplus (111) = (100) = 4$. ♠

**Задачі і вправи**

У всіх нижченаведених задачах вважається, що хеш-функція $h(m) = m$ для всіх значень $m$.

1) Побудувати електронний підпис RSA для повідомлення $m$ з такими параметрами:
а) $p = 5,\ q = 11,\ c = 27,\ m = 7$; б) $p = 5,\ q = 13,\ c = 29,\ m = 10$;
в) $p = 7,\ q = 11,\ c = 43,\ m = 5$; г) $p = 7,\ q = 13,\ c = 29,\ m = 15$;
д) $p = 3,\ q = 11,\ c = 7,\ m = 24$; е) $p = 5,\ q = 13,\ c = 31,\ m = 15$.

2) Для заданих відкритих ключів користувача RSA перевірити легальність підписаних повідомлень:
а) $n = 55, d = 3 : (7, 28),\ (22, 15),\ (16, 36)$; б) $n = 65, d = 5 : (6, 42),\ (10, 30),\ (6, 41)$;
в) $n = 77, d = 7 : (13, 41),\ (11, 28),\ (5, 26)$; г) $n = 91, d = 5 : (15, 71), (11, 46), (16, 74)$;
д) $n = 33, d = 3 : (10, 14),\ (24, 18),\ (17, 8)$; є) $n = 65, d = 5 : (12, 31),\ (9, 26),\ (16, 44)$.

3) Для заданих відкритих ключів користувача системи Ель-Гамаля перевірити легальність підписаних повідомлень:

а) $y = 22:\ (15, 20, 3),\ (15, 10, 5),\ (15, 19, 3)$; б) $y = 9:\ (5, 19, 17),\ (7, 17, 8),\ (6, 17, 8)$;

в) $y = 10:\ (3, 17, 12),\ (2, 17, 12),\ (8, 21, 11)$; г) $y = 6:\ (5, 17, 1),\ (5, 11, 3),\ (5, 17, 10)$.

4) Для заданих секретних параметрів користувачів системи цифрового підпису Ель-Гамаля знайти відкритий ключ $y$ і побудувати підпис повідомлення $m$ при $p = 3,\ g = 5$:

а) $x = 11,\ k = 3,\ m = h = 15$; б) $x = 10,\ k = 15,\ m = h = 5$;

в) $x = 3,\ k = 13,\ m = h = 8$; г) $x = 18,\ k = 7,\ m = h = 5$;

д) $x = 9,\ k = 19,\ m = h = 15$; є) $x = 9,\ k = 3,\ m = h = 21$.

5) У системі електронних грошей вибрано секретні параметри банку: $p = 17, q = 7, c = 77$, а відповідні їм відкриті параметри: $n = 119, d = 5$. Сформувати електронні банкноти з такими номерами:

а) $m = 11$ при $r = 5$; б) $m = 99$ при $r = 6$ в) $m = 55$ при $r = 10$;

г) $m = 44$ при $r = 15$; д) $m = 77$ при $r = 30$; є) $m = 33$ при $r = 7$.

**Лабораторні роботи**

1. Розробити програмну реалізацію системи генерації і перевірки правильності підписів у системі RSA. Параметри користувачів рекомендується вибрати самостійно. Для тестування системи перевірки підписів пропонується використати підписане повідомлення (500,46514) для відкритого ключа користувача $n = 52891, d = 3$, де $h(m) = m$.

2. Розробити програмну реалізацію системи генерації і перевірки правильності підписів у системі Ель-Гамаля. Параметри користувачів рекомендується взяти такими: $p = 31259,\ g = 2$. Решту параметрів пропонується вибрати самостійно. Для тестування системи перевірки підписів пропонується використати підписане повідомлення (500,27665,25022) для відкритого ключа користувача $y = 16196$, де $h(m) = m$.

3. Виконати комп'ютерну реалізацію протоколу:

а) "електронні гроші";

б) Нідхама-Шредера, для якого всі необхідні параметри підібрати самостійно.

## 3.7. Загальний огляд потенційних кіберзагроз

Загальна картина, що характеризує порушників і зловмисників в області комп'ютерної безпеки змінюється із часом, але загальні категорії загроз у сфері небезпек залишаються незмінними. Щоб зірвати плани атакуючих зловмисників, проводять спеціальні дослідження, тому завжди важливо правильно класифікувати різні типи реально існуючих атак.

Наведемо короткий огляд основних типів кіберзагроз [25]:

• шкідливе програмне забезпечення (ПЗ) (malware) або вірус (virus) і програмне забезпечення, спеціально призначене для нанесення збитків або отримання несанкціонованого доступу до комп'ютерних систем (malware - malicious software);

• черв'як (worm) – автономна шкідлива програма, здатна розмножуватися і копіювати себе на інші комп'ютерні системи;

• троянська програма (trojan) – шкідлива програма, що видає себе за одну із звичайних програм, щоб уникнути виявлення;

• програма шпигун (spyware) – шкідлива програма, встановлена на комп'ютерній системі без дозволу і, навіть без відома оператора/користувача, для шпигування та збору інформації. До цієї категорії також належать кейлогери;

• рекламне ПЗ (adware) – шкідлива програма, яка вводить для перегляду рекламні матеріали (наприклад, спливаючі вікна, баннери, відеокліпи) в підсистему інтерфейсу користувача, найчастіше вони з'являються під час перегляду користувачем вебконтенту;

• програма-шантажист (ransomware) – шкідлива програма, спеціально призначена для обмеження функціональних можливостей комп'ютерних систем доки не буде виплачено певну грошову суму (викуп);

• руткіт (rootkit) – комплект ПЗ низького рівня (найчастіше), спеціально призначеного для отримання доступу або повного захоплення управління комп'ютерною системою (root позначає найвищий рівень доступу та управління системою);

• бекдор або чорний вхід (backdoor) – навмисно створена лазівка (дірка), розміщена на периметрі захисту системи, яка дозволяє

в майбутньому отримати доступ в обхід підсистеми зовнішнього захисту;

• бот (bot) – варіант шкідливої програми, що дозволяє атакуючому у віддаленому режимі перехопити управління комп'ютерними системами, перетворюючи їх на зомбі;

• ботнет, мережа ботів (botnet) – велика мережа ботів;

• експлойт (exploit) – фрагмент коду або програма, що використовує конкретні вразливості в інших прикладних програмах або програмних середовищах;

• сканування (scanning): під час атак цього типу на комп'ютерні системи надсилаються різноманітні запити, часто в режимі простого перебору (грубої сили), з метою виявлення слабких місць і вразливостей, а також для збирання інформації;

• перехоплення й аналіз мережного трафіка (sniffing) – непомітне спостереження і фіксація мережного трафіка та внутрішнього трафіка на сервері без відома мережних операторів;

• кейлогер (keylogger) – деталь апаратури або фрагмент ПЗ (найчастіше приховані від користувача), які фіксують усі натискання клавіш на клавіатурі або дії на іншому пристрої введення;

• спам (spam) – незапитувані повідомлення, що розсилаються у великих масштабах, найчастіше в рекламних цілях. Зазвичай використовується електронна пошта, але спам також може поширюватися в повідомленнях СМС або через провайдера системи обміну повідомленнями (наприклад, WhatsApp);

• атака під час процедури реєстрації (login attack) – численні, зазвичай автоматизовані спроби підібрати облікові дані для систем автентифікації, реалізовані у формі простого перебору (грубої сили) або які використовують викрадені або незаконно придбані облікові дані;

• захоплення облікового запису (account takeover – ATO) – отримання доступу до чужого облікового запису, як правило, з метою перешкодити комерційній діяльності, крадіжки особистих даних, викрадення коштів тощо. Зазвичай перехоплення облікового запису з метою атаки відбувається під час процедури реєстрації, але також може мати менший масштаб і більш високу спрямованість

(наприклад, шпигунське ПЗ, соціальна інженерія);

• фішинг (phishing), або маскарадинг (masquerading) – установлення зв'язку від імені людини або організації, що заслуговують на довіру. Мета: переконати об'єкт фішингу надати особисту інформацію або передати права володіння матеріальними цінностями;

• спрямований, або цільовий, фішинг (spear phishing) – фішинг, метою якого є конкретний користувач, із використанням інформації про цього користувача, зібраної з різних зовнішніх джерел;

• соціальна інженерія (social engineering) – отримання інформації від людей із застосуванням нетехнічних методів, таких як помилкова інформація, обман, підкуп, шантаж тощо;

• провокуючий обіг (incendiary speech) – принижувальне, дискредитуюче й інше подібне вороже звернення, адресоване окремо одній особі чи групі осіб;

• атака типу відмова в обслуговуванні, або DoS-атака, та розподілена DoS-атака – атаки, спрямовані на зниження доступності систем і виконання за допомогою численних некоректних запитів та/або запитів, що містять великі обсяги даних. Найчастіше атаки також порушують цілісність і надійність систем;

• цільова кібератака (і розвинена стійка загроза) (advanced persistent threat – APT) – цілеспрямована атака на мережу або на хост, за якої порушник, що ховається, залишається не виявленим протягом довгого часу і постійно викрадає і відстежує дані, що передаються;

• вразливість нульового дня (zeroday vulnerability) – вразливість чи помилка у ПЗ або в комп'ютерній системі, яка невідома виробнику (постачальнику), що дозволяє скористатися нею (атака інульового дня), перш ніж у виробника (постачальника) з'явиться можливість усунути цю проблему.

З яких причин роблять кібератаки? Злочини в інтернеті стають все більш комерціалізованими порівняно з початковим етапом поширення цієї технології. Перехід від кібератак, мотивацією яких було заробляння певної репутації (дешева популярність, особливо у молодіжному середовищі, популярність і навіть просто можливість здійснювати подібні витівки), до кібератак із метою отримання грошей (прямі розкрадання грошей, реклама, продаж особистої

секретної інформації) став дуже принадним процесом, особливо з точки зору зловмисників. У наш час основна спонукальна причина кібератак – це отримання великих грошових сум.

Атаки на фінансові організації або канали (онлайнові платіжні платформи, облікові записи, що містять дані про кредитні та дебетові картки, гаманці біткоїнів і т. п.) можуть відкрити атакуючим зловмисникам прямий доступ до коштів. Але зі збільшенням грошового обсягу, залученого в онлайновий обіг, фінансові організації все частіше застосовують удосконалені механізми захисту, що ускладнюють життя зловмисників. Через спокусу знайти більш короткий і легкий шлях у сферу фінансової діяльності, "ринок" пропонує використання вразливостей у системах захисту фінансових організацій і каналів платежів, що також являє собою багаточисельне та жваве співтовариство. Зловмисники постійно шукають об'єкти зі слабким захистом, неправильно експлуатовані системи з вразливістю через помилки при проєктуванні, а також звертаються до більш витончених методик, які зрештою дозволяють отримати у своє розпорядження деяку грошову суму.

### 3.7.1. Ринок послуг зломщиків

Усім відомо про існування ринків darknet та не цілком законних форумів хакерів і зломщиків. До появи організованих підпільних спільнот, які займаються незаконною діяльністю, допускаються тільки найуміліші хакери, що здатні взяти участь в організації кібератак та злому облікових записів і комп'ютерних систем. Але через зростанням комерціалізації хакерської діяльності і масове застосування комп'ютерів у всіх сферах життя навіть малодосвідчені хакери змогли впровадитися в екосистему кібератак, отримучи у своє розпорядження (придбавши) інформацію про вразливості та зручні для будь-кого користувача хакерські скрипти, програми й інструментальні засоби для здійснення власних кібератак.

На ринку вразливостей нульового дня існують як практично законні, так і абсолютно незаконні варіанти. Торгівля інформацією про вразливості та методиками їхнього використання може стати реальним джерелом доходу і для дослідників у галузі захисту та за-

безпечення безпеки, і для хакерів. Але більшість елітних хакерів не схильні користуватися вразливістю нульового дня та брати участь в організації масових атак. Занадто великий ризик, а крім того, процес переведення в готівку занадто довготривалий і невизначений. Створення ПЗ, яке дає можливість будь-якому недосвідченому скрипткіді (scriptkiddy) здійснити спробу реального злому, продаж інформації про вразливості на вільних ринках, а в деяких випадках навіть невеликі компанії, що надають консультації та послуги зі злому, то все це дозволяє повірити у те, що існує швидкий та легкий шлях до фінансового благополуччя. Як у часи знаменитої золотої лихоманки в Каліфорнії наприкінці 1840-х рр., продавці, що виявляють надзвичайну поштивість і люб'язність до зростаючої армії мисливців за багатством, набагато частіше одержували несподівані прибутки, ніж самі мисливці.

### 3.7.2. Непряма монетизація

Процес монетизації (або переведення в готівку коштів), який передбачається зловмисниками, пов'язаний із різними типами комп'ютерних атак і настільки різноманітний, що заслуговує на детальне вивчення. Тут ми не будемо заглиблюватися в дослідження такого роду, але розглянемо кілька прикладів, що демонструють, як може здійснюватися непряма монетизація. Поширення шкідливого ПЗ було комерціалізовано способом, схожим на розвиток хмарних обчислень (cloud computing) та розгортання провайдерів інфраструктури як сервісу (IaaS – Infrastructure-as-a-Service).

Ринкові відносини типу плати за встановленняі (payperinstall – PPI) під час розповсюдження шкідливого ПЗ є цілком сформованою екосистемою, що надає потужні канали поширення, доступні й авторам, і споживачам. Оренда ботнету заснована на тому ж принципі, що і хмарна інфраструктура, що надається за запитом, із погодинною оплатою ресурсів, що виділяються за прийнятну ціну. Розгортання шкідливого ПЗ на віддалених серверах також може бути прибутковим з фінансової точки зору, з різноманітними специфічними засобами монетизації. Цільові атаки на конкретні об'єкти іноді пов'язані з отриманням будь-якої фінансової вигоди, а поширe-

ння програм-шантажистів може бути достатньо ефективним способом вимагання коштів у великої групи жертв.

Шпигунське ПЗ може сприяти викраденню особистої секретної інформації, яку потім можна вигідно продати оптом на тих же онлайнових ринках для шпигунства. Рекламне ПЗ та засоби поширення спама можна використовувати як дешевий спосіб рекламування не цілком легальних фармацевтичних товарів і фінансових інструментів. Онлайнові облікові записи часто перехоплюються з метою викрадення цінностей, що зберігаються в особливій формі, наприклад, як подарункові (призові) картки, бонусні бали за лояльність, відкриті кредити в магазинах або преміальне повернення грошей під час покупок.

Викрадені номери кредитних карток, номери соціального страхування, облікові записи електронної пошти, номери телефонів, адрес та інша конкретна інформація може бути продана в режимі онлайн злочинцям, які мають наміри зайнятися крадіжками, підробкою, шахрайством та іншими подібними діями. Але процес монетизації (переведення в готівку), особливо якщо зловмисник має номер кредитної картки жертви, може стати довгим і складним. Через можливість легкого викрадення такої інформації компанії, що надають кредитні картки, а також компанії, які обслуговують облікові записи для спеціального зберігання цінностей, часто застосовують хитромудрі технічні методики для запобігання монетизації, яку намагаються виконати зловмисники. Наприклад, якщо виникає підозра, що облікові записи скомпрометовані, то вони оголошуються некоректними та непрацюючими, а для карток із преміальним поверненням грошей потрібні додаткові процедури автентифікації.

Мотиви кіберзлочинців складні, а способи монетизації звивисті. Тим не менш, фінансові вигоди від атак в інтернеті можуть стати потужним стимулом для технічно підготовлених людей, особливо з не дуже багатих країн і спільнот. Поки комп'ютерні атаки здатні створювати обширну кримінальну сферу діяльності для зловмисників, такі атаки продовжуватимуться.

Найбільш реальною небезпекою для комп'ютерних систем є програми, які використовують найслабкіші місця в системі. Далі бу-

дуть розглядатися як програми користувачів, так і програмне забезпечення самої комп'ютерної системи. Розглянемо спочатку небезпеки з боку програмного забезпечення, а потім охарактеризуємо злостиві програми та методи боротьби з ними.

## 3.8. Злостиві програми

На рис. 3.8.1 показано загальну класифікацію небезпек із боку програмного забезпечення, тобто з боку злостивих програм, а в табл. 3.8.1 подано їхні детальніші означення і дії.

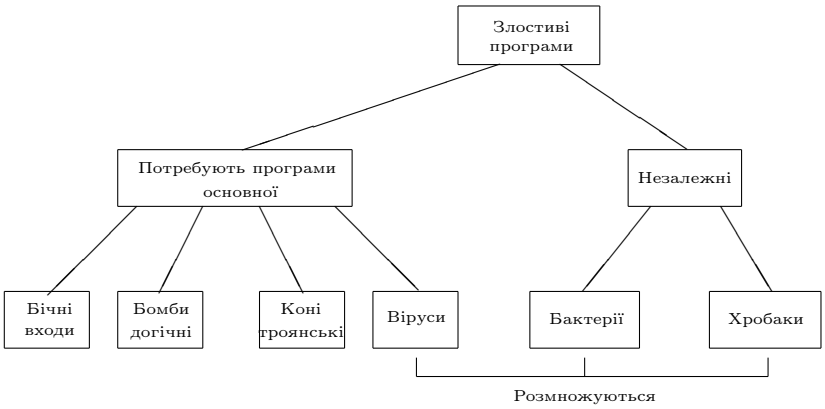



Рис. 3.8.1. Поділ злостивих програм

Ці небезпеки можна поділити на дві категорії: програми залежні і програми системні. Програми першого типу є фрагментами програм, які не можуть існувати незалежно від якої-небудь програми користувача або системної програми. Програми другого типу – це ті, які можуть викликатися і виконуватися за допомогою операційної системи.

Серед злостивих програм виокремлюють клас програм, які не розмножуються, і клас програм, які розмножуються. Програми першого класу є фрагментами програм, які активізуються під час виклику головної програми з метою виконання конкретної операції. До другого класу відносять програми або тільки їхні фрагменти (*віруси*), або незалежні програми (*бактерії, хробаки*), які в момент

їхнього старту можуть продукувати одну або більше копій самих себе і ці копії активізуються на тому самому комп'ютері або на іншій комп'ютерній системі.

Систематизація, подана на рис. 3.8.1 не є повною і вона необхідна лише для організації інформації про небезпечні програми. Наприклад, бомби логічні і коні троянські можуть стати вірусами або хробаками.

**Бічні входи**. Як визначалося вище, бічний вхід є таємним входом у програму, який дає доступ, в обхід існуючих процедур контролю, до інформації. Такі входи програмісти застосовували для тестування і виправлення помилок у програмах. Робиться це, як правило, тоді, коли програміст створює програму для процедури перевірки повноважень або для іншого процесу перевірки умов, виконання яких вимагаються в системі. Щоб виправляти помилки у програмі, програміст може вимагати спеціальних привілеїв або ігнорувати всі правила повноважень. Програміст може забажати переконатися в тому, що існує метод активізації програми у випадку, коли щось було б не в порядку з вбудованою до програми процедурою перевірки повноважень користувача.

**Таблиця 3.8.1.** Небезпеки з боку програм

| |
|---|
| **Бактерії** |
| Програми, які мають засоби розмноження. |
| **Бомби логічні** |
| Фрагмент комп'ютерної програми, який перевіряє виконання певної сукупності умов. Якщо такі умови виконуються, то починає виконувати недозволені дії. |
| **Бічні входи** |
| Таємний, не описаний у документації "вхід" до програми, який дає доступ без дозволу користувача комп'ютерної системи до процедур, до яких доступ обмежений. |
| **Троянські коні** |
| Таємна, не описана у документації процедура, яка міститься у використовуваній програмі. Виконання програми призводить до активізації цієї таємної процедури. |
| **Віруси** |
| Код, який міститься у програмі і який продукує свої копії для однієї або для кількох інших програм і, розповсюджуючись, такий вірус починає виконувати небажані дії. |
| **Хробаки** |
| Програма, яка може розмножуватися і висилати свої копії іншим комп'ютерам за допомогою мережних зв'язків. У момент попадання на комп'ютер хробак може активізуватися, розмножуватися та далі розповсюджуватися і, розповсюджуючись, хробак виконує певні набажані дії. |

Бічний вхід є фрагментом коду, який розпізнає спеціальну послідовність вхідних даних або який активізується під час активізації програми користувачем, або через яку-небудь іншу правдоподібну послідовність подій.

Бічний вхід стає небезпекою тоді, коли він використовується позбавленими доступу програмістами, які намагаються отримати такий доступ. Одна з основних тактик таких зловмисників полягає в тому, що вони висилають нову фальшиву версію операційної системи, в якій задіяний троянський кінь, який і активізується за допомогою бічного входу.

Важко забезпечити цілісність операційної системи перед такими бічними входами. Засоби безпеки мусять вмикатися на етапі створення програм і дій, пов'язаних з активізацією програмного забезпечення.

**Логічна бомба** є одним з найстарших типів небезпеки з боку програмного забезпечення, який передував появі вірусів і хробаків. Бомба логічна є кодом, який легально знаходиться у програмі і який повинен "вибухнути" за виконання певних умов. Прикладами умов, які діють як "детонатори" бомб логічних, є такі умови:

– наявність або відсутність певних файлів;

– конкретний день тижня або конкретна дата;

– конкретний користувач, який використовує програму.

У деяких випадках логічна бомба шукає номер ідентифікатора одного з користувачів (автора бомби), а потім вибухає, якщо ідентифікатор не появився, наприклад, у відомості заробітної плати. "Запалена" бомба може змінити або скасувати дані і цілі файли, призвести до затримки в роботі комп'ютера або заподіяти іншу шкоду.

Відомий приклад застосування логічної бомби до зруйнування бібліотечної системи в окрузі Монтгомері штату Меріленд. Програміст, який розробив сервісні програми для бібліотечної системи, додав до однієї з них логічну бомбу, яка мала вимкнути систему в заданий день, якщо йому не буде виплачена зарплата. Коли бібліотека затримала остаточну виплату, оскільки система мала великий час реакції на запити, то творець системи повідомив про існуван-

ня бомби і пригрозив вибухом, якщо бібліотека йому не заплатить. Бібліотека мусила заплатити йому за роботу.

**Троянські коні** є уживаним засобом, який є таємним кодом, який, у випадку виклику програми, виконує небажані дії. Наприклад, щоб дістати доступ до файлів іншого користувача в розподіленій системі, зловмисник може створити троянського коня, який під час старту програми іншого користувача, змінить умови доступу до його файлів у такий спосіб, що всі користувачі отримають доступ до цих файлів. Зловмисник може на наступному кроці змусити користувачів почати старт своїх програм, розмістивши їх у спільному каталозі і надавши їм пріоритетів легальних програм. Прикладом може служити список файлів користувача в бажаному форматі. Після того, як користувач почне старт програми зловмисник може дістати доступ до інформації, яка знаходиться у файлах цього користувача. Прикладом троянського коня, якого важко викрити, може бути компілятор, який був модифікований так, що він вставляє додатковий код до деяких програм під час їхньої компіляції, наприклад до програми, яка дозволяє початок роботи із системою. Такий код створює в цій програмі бічний вхід, який дозволяє зловмиснику розпочати працю із системою з використанням спеціального пароля. Такого коня не можна виявити за допомогою читання первинного коду програми.

Другою функцією коней троянських, яка часто зустрічається, є функція знищення даних. Програма виконує нормально своє завдання (наприклад, калькулятор), але в той же час тихо знищує файли користувача. Наприклад, шеф британського бюро розвідки був атакований конем троянським, який знищив усю інформацію, яка розміщувалася в пам'яті його комп'ютера. Троянський кінь був включений до сервісної процедури графічного інтерфейсу, якою користувався шеф розвідки.

**Віруси**. Вірус – це програма, яка "заражує" інші програми шляхом їхньої модифікації; модифікація полягає в копіюванні вірусної програми, яка може заражати інші програми. Біологічні віруси являють собою маленькі фрагменти генетичного коду – ДНК або РНК – які можуть наслідувати механізм звичайної клітини і змушува-

ти його до продукування тисяч точних своїх копій. Відповідний комп'ютерний вірус має у своєму коді команду на продукування точних копій самого себе. Типовий вірус, якщо він потрапляє до комп'ютера, бере тимчасовий контроль над операційною системою. Кожний раз, коли заражений комп'ютер зустрічається з незараженим програмним забезпеченням, продукується свіжа копія вірусу і переноситься на незаражені програми. Інфекція така може переноситися з одного комп'ютера на інші за допомогою користувача, який нічого не підозрюючи, використовує флешки чи дискетки або висилає програми через мережу. У мережному середовищі можливість заразитися самому або заразити інших є великою, оскільки умови для розповсюдження віруса хороші. Пізніше будуть розглянуті спеціальні типи вірусив.

**Бактерії** – це програми, які в принципі не знищують інші файли, а єдиним їхнім завданням є розмноження. Типова бактерія може нічого не робити, крім одночасного початку "виготовлення" двох і більше копій самої себе або створення двох нових копій, кожна з яких є копією оригінальної бактерії. Далі обидві програми двічі копіюються і так далі. Бактерії розмножуються екпоненціально, призводячи в кінці до блокування пам'яті, або зменшують швидкість процесора, забиваючи місце на диску і блокуючи доступ до необхідних засобів.

**Хробаки** – це фрагменти коду, які використовують мережні лінії зв'язку для роповсюдження із системи на систему. Як тільки вони стартують в якійсь системі, то можуть себе вести так само, як віруси або бактерії, а також можуть призвести до старту троянського коня і виконувати довільні знищувальні дії в системі.

### 3.8.1. Віруси

Вірус може робити те саме, що і довільна програма. Єдина різниця полягає в тому, що він може приєднуватися до іншої програми і виконуватися у кожному старті основної програми. У табл. 3.8.2 описано простий спосіб створення вірусу, основним завданням якого є розповсюдження. У цьому прикладі описано тільки спосіб утаємничення і механізм розповсюдження вірусу. Якби діяльність

вірусів обмежувалася тільки такими діями, то не було б особливих проблем. На жаль, коли вірус вже діє, то він може виконувати довільну роботу, наприклад знищувати файли і програми. У тому і полягає основна небезпека, пов'язана з вірусами.

**Таблиця 3.8.2.** Дія вірусу

| |
|---|
| Приклад способу дії простого вірусу, який виконує тільки зараження програм і написаний на мові асемблера: |
| Читання інструкції програми |
| JCJ переходом перейти до місця в пам'яті, |
| яке є наступним за останньою інструкцією програми |
| Вставити в це місце код вірусу |
| Вірус симулює інструкцію, яка виникла |
| в результаті такого переходу |
| Перехід до другої інструкції програми |
| Закінчення виконання основної програми |
| Піл час кожного старту основної програми вірус може заразити інші програми, а потім виконати основну програму. Користувач не помітить нічого, крім короткого запізнення виконання. |

У табл. 3.8.3 показані найбільш відомі віруси і їхня злостивість.

За час свого існування вірус переходить через такі чотири фази:

1. *Фаза сну*, я якій вірус бездіяльний. Він починає свою діяльність, коли активізується за допомогою якоїсь події, наприклад досягнення певної дати, наявність іншої програми чи файла, переповнення певного об'єму пам'яті на диску і т. п. Не всі віруси проходять цю стадію.

2. *Фаза розповсюдження*, під час якої вірус розміщує свої ідентичні копії в інших програмах або в інших системних об'єктах. Кожна заражена програма має клон вірусу, який далі переходить у фазу розповсюдження.

3. *Фаза активності*, під час якої вірус виконує дії, запрограмовані зловмисником. Так як і у фазі сну, фаза активності вірусу може пробуджуватися за допомогою різних подій.

4. *Фаза виконання*, під час якої вірус виконує свою функцію. Може вона мати характер нешкідливий, наприклад, поява якогонебудь повідомлення на екрані, або бути шкідливою і знищувати програми й іншу інформацію в системі.

Більшість вірусів діють спеціальним чином для конкретної операційної системи, а в деяких випадках вони орієнтовані на специфіку засобів конкретної системи. Використовують при цьому окремі

слабкі місця в системах.

**Структура вірусу**. Вірус може приєднатися на початку або в кінці програми, яка виконується, або може бути вбудований до неї яким-небудь іншим способом. У результаті інфікована програма викликає код вірусу, а потім виконує власний код.

Загальний вигляд вірусу є таким:

```
    program V:=
{goto main;
      1234567;
        subroutine заразити: =
          {цикл:
           файл: = вибрана-програма-жертва;
           if (перший-оператор-файла = 1234567)
              then goto цикл
           else додати-до-початку файл код V;}
        subroutine шкодити: = {щось-що-шкодить}
        subroutine виключи: =
          {повернути true якщо виконується якась умова}
main: program-main: =
         {заразити;
          if старт then шкодити;
          goto далі;}
далі:
```

У цьому випадку код вірусу $V$ приєднується до початку зараженої програми і діє так, що місце приєднання до програми є його першим оператором.

Заражена програма розпочинає свою роботу з коду вірусу і діє таким чином. Перший оператор є оператором переходу до головної програми вірусу. Другий оператор – це спеціальний значок, який створює вірус, для позначення вже зараженої програми-жертви. У момент виклику програми, контроль на себе бере головна програма вірусу. Програма вірусу спочатку шукає незаражені файли і їх інфікує. Далі вірус може виконувати які завгодно дії, що, як правило, є шкідливими для системи. Дії можуть бути різними і виконуватися при кожному виклику програми, наприклад можуть носити характер логічної бомби, яка вибухне під час виконання спеціальних умов. У кінці своєї "діяльності" вірус передає контролюючі функції програмі-жертві. Якщо фаза інфікування триває недовго, то користувач не помітить жодної різниці між виконанням інфікованої і неінфікованої програми.

Наведений тип вірусу легко викрити, оскільки інфікована версія програми буде довшою, ніж неінфікована. Основним способом боротьби з викриттям вірусу є його компресія. Це означає, що інфіковану програму скомпресовують так, щоб її довжина не відрізнялася від неінфікованої програми. На рис. 3.8.2 зображено необхідний для таких дій алгоритм. Важливі рядки вірусу перенумеровано, а на рис. 3.8.3 показано його операції. Припускається, що програма $P_1$ інфікована вірусом $CV$. Під час виклику програми контроль на себе бере вірус, який виконує такі дії:

Вірус спочатку компресує кожний неінфікований файл $P_2$, отримує файл $P_2'$, який коротший від оригінальної програми на розмір вірусу.

Копія вірусу приєднується до початку скомпресованої таким чином програми.

Скомпресована версія інфікованої програми $P_1'$ розкомпресовується. Нескомпресований оригінал програми виконується.

```
program CV =
  {goto main,
    01234567,
    subroutine зарази-виконувану =
         {цикл:
          файл = довільна-програма-що-виконується;
          if (перший-рядок файл = 01234567) then goto цикл;
    (1)    скомпресуй файл
    (2)    додай CV до файл,  }
  main: program main =
             {if викликається then зарази-виконувану;
       (3)   розкомпресуй решту-файлів;
       (4)   виклич файл розкомпресований;}
       }
```

Рис. 3.8.2. Алгоритм компресаційного вірусу

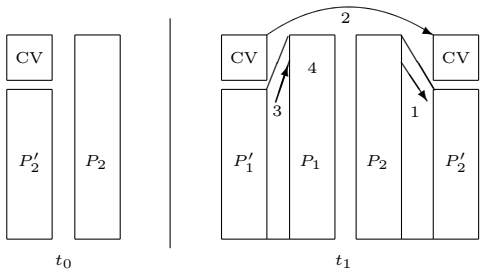



Рис. 3.8.3. Вірус компресаційний

У наведеному прикладі вірус тільки розмножується. Так само, як і в попередньому прикладі, вірус може бути логічною бомбою.

У табл. 3.8.3 наведено деякі відомі віруси.

**Таблиця 3.8.3.** Деякі міжнародні віруси

| **Віруси, що атакують комп'ютери IBM-PC** |
|---|
| **Пакистанський Розум** |
| Один із найпростіших вірусів, випродукований в Пакистані, звідки і походить його назва. Заражує boot sector диска PC-DOS і розмножується, заражуючи кожну флешку, приєднану до комп'ютера. Цей вірус бере на себе контроль над підсистемою управління флешками. Якщо відбувається операція читання, то він її відкладає і пробує знайти шлях до стартового коду. Якщо підтвержується, що то є стартовий код і він не є заражений, то вірус його модифікує, розміщуючи в ньому вірус. Деякі версії цього вірусу позначають частини диска як заражені, не дивлячись на те що ці частини справні. У результаті диск матиме тільки заражені сектори. |
| **Вірус Єрусалимський** |
| Цей вірус заражує програми, які виконуються, наприклад, файли з розширенням COM або EXE. Діє в пам'яті і заражує кожну таку програму. Знищує таблиці розміщення файлів, що унеможливлює доступ до файлів на диску і призводить до недоступності даних на диску. Розповсюджується через флешки і дискетки й атакує жорсткі диски. |
| **Вірус LeHigh** |
| Цей вірус заражує операційні системи і псує роботу процесора завдань. У кожному доступі до диску вірус перевіряє, чи процесор завдань цього диска вже заражений, чи ні. Якщо ні, то вірус його заражує. Якщо так, то збільшується значення лічильника, який контролюється вірусом. Коли значення лічильника стає рівним 10, вірус знищує всі дані на жорсткому диску. |
| **Вірус Alemeda** |
| Цей вірус заражує boot sector системи, а потім заражує кожну флешку чи дискетку, яка підключається під час рестарту системи, і знищує шлях до флешки. |
| **Віруси, що атакують комп'ютери Macintosh** |
| **Вірус Scores** |
| Розмножується через задане число днів, а потім ще кілька днів залишається бездіяльним. Далі, коли користувач пробує записати інформацію у файл, вірус не дозволяє цього зробити, чим блокує всю роботу в системі. |
| **nVIR** |
| Існує багато форм цього вірусу і викрито принаймні кілька десятків з них. Техніка діяльності вірусу пов'язана з розповсюдженням і є злостивою. Він атакує системні файли і, як тільки такий файл стає зараженим, заражує кожну з програм, яка стартує в системі. |

**Типи вірусів**. Відтоді як появилися віруси, триває запекла бо-

ротьба між тими, хто їх створює, і тими, хто створює антивірусні засоби. Створення ефективних антивірусних засобів провокує створення і нових типів вірусів. Серед найважливіших категорій вірусів виокремимо такі:

• *Вірус паразитичний:* традиційна і найбільш розповсюджена форма вірусу. Цей вірус приєднується до виконуваних програм, виконує інфіковану програму, розмножується, шукаючи нові виконувані програми.

• *Вірус резидентний:* гніздиться в оперативній пам'яті як частина резидентної системної програми. Із цього моменту він інфікує всі виконувані програми.

• *Вірус boot sectora:* інфікує основний запис, упроваджуючи так званий (*master boot record*), або запис, який упроваджує (*boot record*), і переноситься, коли система стартує з диска, на якому міститься вірус.

• *Таємний вірус:* ця форма вірусу уникає викривання свого існування за допомогою антивірусних програм.

• *Вірус поліморфічний:* вірус, який здатний до мутації кожного разу, коли інфікує якийсь файл, що складає труднощі його знешкодження.

Прикладом таємного вірусу є вищеописаний вірус: це був вірус, який компресує файли так, щоб інфікована і неінфікована програми мали однакові довжини. Можливі й інші, ще небезпечніші форми такого вірусу. Наприклад, вірус може розміщуватися у процедурах, які співпрацюють з диском. У такому випадку вірус перехоплює управління на себе і, коли стартує пошук інфікованої програми, він презентує оригінальну неінфіковану програму. Термін таємності не належить до таємного вірусу як такого, а тільки відноситься до способу уникання викриття.

**Вірус поліморфічний** створюється під час розмноження копій функціонально еквівалентних, але з іншим складом бітів. Як і у випадку таємного вірусa, основною метою є "перехоплення" програм пошуку вірусів. У такому випадку структура вірусу буде різною в кожній копії. Щоб отримати таку неоднорідність, вірус може випадковим чином розміщувати копії надлишкових інструкцій або

змінювати їхню послідовність. Ефективним методом є шифрування. Частина вірусу, звана *процедурою мутації (mutation engine)*, створює випадковий ключ шифрування для шифрування решти частин вірусу. Цей ключ переховується разом із вірусом і починається модифікація процедури. Коли інфікована програма викликається, вірус застосовує цей ключ для розшифрування. Під час розмноження вірусу, створюється інший випадковий ключ.

Іншим видом зброї творців вірусів є готова сукупність засобів створення вірусів. Така сукупність дає можливість особі, яка тільки починає створювати віруси, швидко розробляти різноманітні віруси. Часто таким чином створені віруси не є такими "вирафінованими", як віруси, створені хакерами, але сама можливість генерації нових вірусів складає проблему для антивірусних систем.

Ще одним засобом творців вірусів є ВВС, яка служить для обміну вірусами. У США та інших країнах появилася велика кількість таких ВВС. Вони пропонують копії вірусів, які доступні через інтернет, а також вказівки для створення нових вірусів.

### 3.8.2. Методи боротьби з вірусами

Ідеальним розв'язанням у боротьбі з вірусами є профілактика: перш за все не потрібно дозволяти того, щоб вірус потрапив на систему. Зрозуміло, що досягнути такої мети неможливо, хоча профілактичні дії зменшують імовірність вірусної атаки. Найкращим, після профілактики, є метод, який дає такі можливості:

• *Викриття:* якщо появилися інфіковані файли, то належить про те дізнатися і локалізувати вірусу.

• *Ідентифікація:* якщо викриття успішно закінчилося, то потрібно ідентифікувати конкретний тип вірусу, яким інфікована програма.

• *Вилучення:* після ідентифікації вірусу належить повністю вилучити його з кожної програми і відновити програму до первинного стану.

Якщо вдалося викрити вірус, але ідентифікація і вилучення неможливі, то залишається вилучити всі інфіковані програми і завдан-

тажити чисті версії.

Розвиток технололгій створення та боротьби з вірусами йдуть поряд. Раніше віруси були відносно простими фрагментами коду і їх можна було розпізнати й усувати за допомогою відносно простих пакетів антивірусних програм. Разом із розвитком вірусів, розвивалося також антивірусне програмування і з кожним роком воно все більш ускладнювалося.

Розрізняють наступні чотири генерації антивірусного програмного забезпечення:

- *Перша генерація:* прості сканери.
- *Друга генерація:* евристичні сканери.
- *Третя генерація:* перехоплення активності.
- *Четверта генерація:* повна охорона.

Сканер **першої генерації** служить для розпізнавання вірусу і потребує його структури. Вірус може мати знаки, які узагальнюють (глобальні) ознаки, але всі копії загалом мають таку саму структуру і склад бітів. Сканери, які опираються на структуру вірусу можуть тільки викривати відомі їм віруси. Інші типи сканерів першої генерації реєструють довжину програм і шукають зміни довжини.

Сканер **другої генерації** не опирається на структуру вірусу. Викриває він інфекцію, вживаючи певні евристичні правила. Сканери, які належать до одного з таких класів, шукають фрагменти коду, які часто входять у склад вірусів. Сканер може, наприклад, шукати початок циклу, який шифрує поліморфічний вірус, і відкрити ключ шифрування. Після відкриття ключа сканер може відшифрувати вірус, ідентифікувати його, усунути інфекцію і повернути оригінальність програмі. Друга генерація використовує інший метод, який перевіряє цілісність програми. До кожної програми можна додати контрольну суму. Якщо вірус інфікував програму, не змінюючи її контрольної суми, зміна стане відома механізму, що перевіряє її цілісність. Для боротьби з досить спритними вірусами використовують функції хешування. Ключ шифрування зберігається окремо від програми для того, щоб вірус не міг згенерувати нового результату хешування і згенерувати його заново. Завдяки функції хешування замість простої контрольної суми, вірус не може моди-

фікувати програму так, щоб був такий самий результат хешування, як і раніше.

Програми **третьої генерації** – це програми резидентні, які розпізнають вірус, не керуючись його структурою, і працюють на основі аналізу його діяльності. Програми ці мають ті переваги, що не потрібно займатися структурами й евристиками великої кількості вірусів. Потрібно лише розпізнавати невелику множину дій, які пов'язані з перевіркою інфікування.

Продукти **четвертої генерації** – це пакети, які реалізують багато антивірусних методів і технік, які пов'язані між собою. Серед основних функцій цих пакетів є виявлення і нейтралізація активності вірусу. Такий пакет, крім того, виконує контроль доступу, що обмежує можливість псування вірусом системи й обмежує здатність вірусу до модифікації файлів із метою їхнього зараження.

Боротьба триває. Пакети четвертої генерації реалізують всесторонню стратегію, збільшуючи межі оборони перед вірусами.

### 3.8.3. Хробаки

Мережні хробаки мають риси вірусу і риси зловмисника. Як і вірус, хробак є саморозповсюджуваним фрагментом коду, який займається принаймні продукуванням своїх копій, а, крім того, після кожного виконання може активізувати шкідливий фрагмент коду. Як і у випадку зловмисника, метою хробака є проникнення в систему: хробак старається розмістити свої копії в інших комп'ютерах у межах комп'ютерної мережі або інтермережі.

**Розповсюдження хробаків**. Мережний хробак проходить серію фаз, подібних до тих, які проходить комп'ютерний вірус: фазу сну, фазу розмноження, фазу активності та фазу виконання. Під час виконання він реалізує такі функції:

- Пошук інших систем для інфікування за допомогою аналізу команд комп'ютера або списків, в яких розміщено адреси віддалених систем.
- Нав'язування з'єднань із віддаленими системами.
- Копіювання самого себе на віддаленій системі. Нова копія про-

грами хробака починає виконуватися на віддаленій системі, де, крім виконання інших дій, розповсюджується далі таким самим способом.

Мережний хробак може також перевіряти перед копіюванням на тій, чи іншій системі, чи вона була раніше інфікована, чи ні. У системі багатопроцесорній може також приховувати свою присутність, присвоюючи собі назви системних процесів або які-небудь інші назви, на які не зважатиме операційна система.

Ступінь небезпеки зараження мережі залежить від здатності хробака до розповсюдження і перенесення на інші комп'ютери. А це, у свою чергу, залежить від ступеня зв'язку між системами. Чим тісніший зв'язок між системами, тим більше можливостей для хробака. Рис. 3.8.4 показує ступені зв'язків. Разом зі збільшенням ступеня зв'язку зростає ризик атаки хробаком.

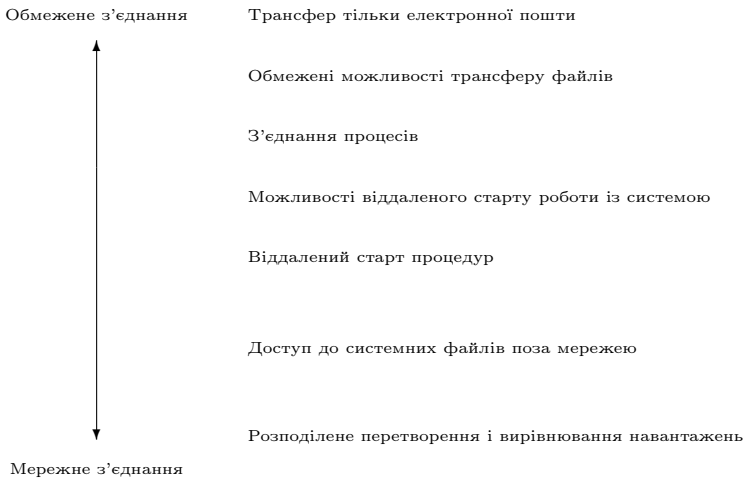



Рис. 3.8.4. Ступінь з'єднання систем

Найменш еластична форма зв'язку – електронна пошта. Хробак використовує її, висилаючи копію самого себе на інші системи. У більшості випадків застосування електронної пошти потребує людського втручання в якійсь точці циклу. Наприклад, хробак "Ялинка різдвяна", випущений у грудні 1987 р., широко розповсюджений у мережах $BITNET^*$, $EARN^*$ а також у внутрішній мережі IBM.

Цей хробак являв собою електронний список, який був повідомленням для отримувача разом із кодом, що виконувався на великих комп'ютерах IBM. У повідомленні говорилося, що коли код буде виконано, на екрані появиться різдвяна ялинка. Це дійсно так було, але крім ялинки хробак висилав копії самого себе до всіх користувачів, адреси яких містилися у списку адрес.

Цей хробак не був справжнім хробаком, оскільки вимагав втручання користувача. Був він, скоріше троянським конем із механізмом розмноження типу "ланцюжок". Можна, тим не менше, використати недоліки системи електронної пошти з метою перенесення автоматизованого розмноження. Це був один із методів застосування хробака інтернету.

Повертаючись до рис. 3.8.4, бачимо, що зі зростанням ступеня зв'язку хробак використовує все більшу область дії і має щораз більше можливостей переховуватися.

### 3.8.4. Хробаки інтернету

Найславетнішим хробаком є хробак, впроваджений в інтернет Робертом Морнсом в 1988 р. Цей хробак служить хорошим прикладом, оскільки він має багато рис ефективного хробака.

Хробак інтернету розповсюджується в системах UNIX і вживає багато різних технік перенесення. Коли хробак починає діяти, його першою дією був пошук інших комп'ютерів, які відомі даному комп'ютерові і до яких можна було знайти доступ. Хробак переглядав різні списки і таблиці, зокрема і системні таблиці, з метою пошуку машин, які належать до перевірених, файлів контролю передачі електоронної пошти користувачів, службових таблиць надання прав доступу до віддалених комп'ютерів, паролів, рахунків тощо. До кожного відкритого комп'ютера хробак пробував декілька методів отримання доступу:

• Пробував розпочати роботу з віддаленим комп'ютером під виглядом легального користувача. У межах цього методу пробував спершу зламати локальний файл паролів, а потім використовував відкриті паролі і відповідні їм ID користувача. Застосовуючи припущення, що багато користувачів вживають ті самі паролі в різних

системах, діяв далі. Для отримання пароля хробак викликав програму ламання паролів, яка продукувала:

(а) назву рахунку кожного користувача і його перестановку;

(б) список 432 вбудованих паролів, які автор вживав як правдоподібні (табл. 3.8.4);

(с) усі слова з локального системного словника.

• Використовував помилку у протоколі "finger", в якому містилися дані віддаленого користувача.

• Використовував бічний вхід в опцію, яка вилучає помилки віддаленого процесу, що отримує вислану пошту.

Якщо застосування якого-небудь із цих методів закінчувалось успіхом, то хробак шукав з'єднання з інтерпретатором виконання завдань операційної системи. Далі він висилав інтерпретаторові коротку проміжну програму і видавав інтерпретаторові завдання виконання цієї програми, а потім закінчував роботу із системою. Проміжна програма викликала головну програму і виконувала решту програми хробака. Нарешті він розшифровував блок, який створював нового хробака, і видавав команду на його виконання.

Хробак був створений так, щоб його важко було локалізувати і зрозуміти. Головний хробак пересилався між системами тільки в зашифрованому вигляді, не залишаючи жодних слідів у файловій системі: всі вживані файли копіювалися в основну пам'ять і жоден із них не був створений заново. Хробак виключав системні функції, які полягали у локалізації частин пам'яті на випадок помилки. Мав нешкідливу назву і часто змінював свій ідентифікатор процесу.

**Таблиця 3.8.4.** Паролі, тестовані хробаком інтернету

| | | | | | | |
|---|---|---|---|---|---|---|
| aaa | carmen | engineer | herbert | minimum | rainbow | super |
| academia | carolina | enterprise | hiawatha | minsky | raindrop | superstage |
| aerobics | caroline | enzyme | hibernia | moguls | raieigh | support |
| airplane | cascades | ersatz | honey | moose | random | supported |
| albany | castle | establish | horse | morley | rascal | surfer |
| albatross | cat | estate | horus | mozart | really | suzanne |
| albert | cayuga | euclid | hutchins | nancy | rebecca | swearer |
| alex | celtics | evelyn | imbroglio | napoleon | remote | symmetry |
| alexander | cerulean | extension | imperial | nepenthe | rick | tangerine |

### 3.8.5. Методи боротьби з хробаками

Звіт Національного Інституту Стандартів і Технологій (National Institute of Standards and Technology) описує такі основні риси хробаків:

• Хробаки, які використовують шляхи операційної системи або помилки в управлінні системами з метою розмноження.

• Поява хробака призводить до короткої, але шкідливої події, такої як блокування роботи цілих мереж.

Проти хробаків можна використовувати такі засоби боротьби:

• *Контроль доступу:* відповідна ідентифікація і перевірка ідентичності особи користувача запобігає перенесенню хробака на машини без дозволу. Засобами перевірки відповідної ідентифікації є методи охорони паролів, про які йшлося раніше.

• *Виявлення спроб ламання:* методи виявлення спроб ламання, про які йшлося вище, є засобами боротьби з хробаками.

• *Вогняні стіни (firewalls):* одна мережа LAN або міжмережний домен, що складається з великої кількості з'єднаних між собою мереж, може охоронятися за допомогою системи вогняних стін. Кожний, хто хоче розпочати роботу із системою, повинен пройти через вогняні стіни.

На щастя хробаки зустрічаються зрідка, оскільки їх важко створювати. Хробакам потрібне мережне середовище, яке складається з декількох систем, що мають відповідні недоліки. Автор хробака повинен знати, як можна використати недоліки системи і як створити хробака здатним до розмноження.

### 3.8.6. Підсумки

Зі сказаного випливає декілька рекомендацій користувачеві криптографічної системи.

• Вірити тому, хто запевняє вас, що може зробити абсолютно безпечну інформаційну систему, є великою помилкою. Така особа є або вашим конкурентом, або просто особою нерозумною. Спеціаліст завжди буде мати сумніви відносно рівня безпеки, створеної ним системи.

• Не повинно бути ілюзії на тему абсолютної безпеки системи. Не можна сподіватися тільки на свій інтелект і досвід. Із часом стане зрозумілим, що ваша система не надто безпечна.

• Не варто занадто рекламувати свою систему безпеки. Якщо буде забагато реклами, то напевно хто-небудь захоче її зламати.

• Не варто бути подібним до науковця! Навіть коли доведено твердження про неможливість зламання системи безпеки, то не повинно бути повної впевненості. Потрібно спробувати зрозуміти, чому і коли не виконуватимуться умови такого твердження.

• Безпека безпеки: якщо побудована дуже надійна система безпеки, то потрібно перевірити безпеку такої системи.

**Контрольні запитання**

1. Назвати основні небезпеки криптографічних систем.
2. Охарактеризувати злостиві програми хробак, вірус і бомба логічна.
3. Охарактеризувати злостиву програму – бактерія.
4. Описати структуру вірусу та фази його розвитку.

**Задачі і вправи**

1. Перерахуйте типи міжнародних вірусів та їхні злостиві дії.
2. Яка різниця між вірусом, хробаком і бактерією?
3. Які існують методи боротьби з вірусами та хробаками?
4. Чому немає абсолютної впевненості в надійності системи безпеки комп'ютера?
5. Які дії повинен виконати користувач перед початком роботи із системою і виконання якіх дій має вимагати система безпеки від користувача?
6. Створіть програму простого вірусу, який продукує лише одну свою копію.
7. Створіть програму простої бактерії, яка передається через електронну пошту лише одному з ваших приятелів.

# Список літератури


[1] *Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В.* Основы криптографии. – *М.:Гелиос АРВ. - 2001. - 480 с.*

[2] *Бородін О. І.* Теорія чисел. – Київ:"Радянська школа". – 1965. – 261 с.

[3] *Василенко О.Н* Теоретико-числовые алгоритмы в криптографии. – *М.: МЦНМО. – 2003.– 328 с.*

[4] *Венбо Мао.* Современная криптография. – *СПб.: Изд. дом "Вильямс". – 2005.– 763 с.*

[5] *Виноградов И.М.* Основы теории чисел. – *М.: Наука. – 1972.– 167 с.*

[6] *Вирт Н.* Систематическое программирование. – *М.: Мир. – 1985. – 183 с.*

[7] *Гилл А.* Линейные последовательные машины. Анализ, синтез и применение. – *М.: Наука. – 1974.– 210 с.*

[8] *Донец Г.А.* Решение задачи о сейфе на (0,1)-матрицах. *Кибернетика и системный анализ. – 2002. – № 1. – С. 98-105.*

[9] *Завадська Л.О., Савчук М.М.* Математичні методи захисту інформації: курс лекцій. Частина 1. – *Київ: НТУУ "КПІ", 2008. – 128 с.*

[10] *Калужнин Л.А.* Введение в общую алгебру. – *М.: Наука. – 1973.– 447 с.*

[11] *Кнут Д.* Искусство программирования для ЭВМ. т. 2. Получисленные алгоритмы. – *М.: Мир. – 1977.– 723 с.*

[12] *Коблиц Н.* Курс теории чисел и криптографии. – *М.:Изд. ТВП.– 2001.– 260 с.*

[13] *Коробейников А.Г., Гатчин Ю.А.* Математические основы криптологии. – *СПб.: Изд. ИТМО. – 2004.– 110 с.*

[14] *Кострикин А.И.* Введение в алгебру. – *М: Наука. – 1977.– 495 с.*

[15] *Кривий С.Л.* Вступ до методів створення програмних продуктів. – *Київ: Редакційно-видавничий відділ НаУКМА. – 2018. – 449 с.*

[16] *Кривий С.Л.* Криптосистема на основі абелевих груп і кілець. – *ж. Проблеми програмування. – № 2-3. – 2020. – С. 270-277.*

[17] *Кривий С.Л.* Лінійні діофантові обмеження та їх застосування. – *Київ: Інтерсервіс. – 2021. – 257 с.*

[18] *Липский В.* Комбинаторика для программистов. – М.:Мир. – 1988. – 201 с.

[19] *Молдовян А.А., Молдовян Н.А., Советов Б.Я.* Криптография. – *СПб: Изд. "Лань". – 2001. – 218 с.*

[20] *Рябко Б.Я., Фионов А.Н.* Криптографические методы защиты информации. – *М: Горячая линия – Телеком. – 2005. – 229 с.*

[21] *Схрейвер А.* Теория линейного и целочисленного программирования. т. 1. – *М.: Мир. – 1991. – 380 с.*

[22] *Шеннон К.* Работы по теории информации и кибернетике (Теория связи в секретных системах). – *М.:ИЛ. – 1963. – С. 333–369.*

[23] *Шнейер Б.* Криптография для практиков. – *СПб: Изд. "Вильямс". – 2002. – 899 с.*

[24] *Черемушкин А. В.* Лекции по арифметическим алгоритмам в криптографии. – *М:МЦНМО. – 2002. – 103 с.*

[25] *Чио К., Фримэт Д.* Машинное обучение и безопасность. – *М:ДМК Прес. – 2020. – 388 с.*

[26] *Ященко В.В. и др.* Введение в криптографию. – *М:МЦНМО. – 2000. – 287 с.*

[27] *Bellovin S.* Packets Found on an Internet. – *Computer Communication Review. – July. – 1993. – P. 143–151.*

[28] *Bloom B.* Space/time Trade-offs in Hash Coding with Allowable Errors. – *Communications of the ACM. – july. – 1970. – P. 32–41.*

[29] *Diffie W., Hellman M.E.* New direction in cryptography. – *IEEE Transaction on Information Theory. – 1976. v. 22. – P. 644–654.*

[30] *Matyas S., Le A., Abracham D.* A Key Management Scheme Based on Control Vektors. – *IBM System Journ. № 3. – 1991. – P. 21–29.*

[31] *Menezes A., van Oorschot P., Vanstons S.* Handbook of Applied Cryptography. – *CRC Press. – 1996. – 661 p.*

[32] *Merkle R. and Hellman H.* Hiding Information and Signatures in Trap Door Knapsacks. – *IEEE Transac. on Inform. Theory, September, – 1978. P. 241–245.*

[33] *Miller G.* Riman's Hypothesis and Tests for Primality. – *Procieedings of the Seventh Annual ACM Symposium on the Theory of Computing, May. 1975. – P. 47–49. P. 36-45.*

[34] *Papadimtriou C. H.* Złożoność obliczeniowa. – *Wydawnictwo Naukowo-Techniczne, Warszawa. – 2002. – 540 s.*

[35] *Rabin M.* Probabilistic Algorithm for Primality Testing. – *Journal of Number Theory, December. 1980. – P. 70–79.*

[36] *Rivest R., Shamir A., Adleman L.* A Method for Obtaining Digital Signature and Public Key Cryptosystems. – *Communic. of the ACM, February. – 1978. P. 36–45.*

[37] *K.Rosen, J. Michaels, J. Gross, J. Grossman, D. Shier.* Handbook of Discrete and Combinatorial Mathematics. – *CRC Press.- 2000. - chap.2.4. - p. 219.*

[38] *Safford D., Shales D., Hess D.* The TAMU Security Package. An Ongoing Response to Internet Intruders in an Academic Environment. – *Proceedings UNIX Security Simposium IV. – October. – 1993. – P. 17 – 22.*

[39] *Seberry J., Pierzyk J.* Cryptogaphy: An Introduction to Computer Security. – *Englewood Cliffs, NY: Prentice-Hall. – 1989. – 351 p.*

[40] *Stallings William.* Ochrona danych w sieci i intersieci. – *Warszawa: Wydawnictwo Naukowo Techniczne. – 1997. – 474 s.*

[41] *Stoll C.* Stalking the Wily Hackers. – *Communications of the ASM. – May. – 1988. – P. 16–19.*

# Предметний покажчик

# АЛФАВІТИ, ЛІТЕРИ ЯКИХ ВИКОРИСТАНО В ПОСІБНИКУ

**Українська абетка**

| | | | | | |
|---|---|---|---|---|---|
| Аа | Бб | Вв | Гг | Ґґ | Дд |
| Ее | Єє | Жж | Зз | Ии | Іі |
| Її | Йй | Кк | Лл | Мм | Нн |
| Оо | Пп | Рр | Сс | Тт | Уу |
| Фф | Хх | Цц | Чч | Шш | Щщ |
| Ьь | Юю | Яя | | | |

**Англійська абетка**

| | | | | | |
|---|---|---|---|---|---|
| Aa | Bb | Cc | Dd | Ee | Ff |
| Gg | Hh | Ii | Jj | Kk | Ll |
| Mm | Nn | Oo | Pp | Qq | Rr |
| Ss | Tt | Uu | Vv | Ww | Xx |
| Yy | Zz | | | | |

**Грецька абетка**

| | | | | |
|---|---|---|---|---|
| Aα | Bβ | Γγ | Δδ | Eϵ |
| Zζ | Hη | Θθ | Iι | Kκ |
| Λλ | Mμ | Nν | Ξξ | Oo |
| Ππ | Pρ | Σσ | Tτ | Υυ |
| Φφ | $X\chi$ | Ψψ | Ωω | |

**Англійська абетка каліграфічна**

| | | | | | |
|---|---|---|---|---|---|
| $\mathcal{A}$ | $\mathcal{B}$ | $\mathcal{C}$ | $\mathcal{D}$ | $\mathcal{E}$ | $\mathcal{F}$ |
| $\mathcal{G}$ | $\mathcal{H}$ | $\mathcal{I}$ | $\mathcal{J}$ | $\mathcal{K}$ | $\mathcal{L}$ |
| $\mathcal{M}$ | $\mathcal{N}$ | $\mathcal{O}$ | $\mathcal{P}$ | $\mathcal{Q}$ | $\mathcal{R}$ |
| $\mathcal{S}$ | $\mathcal{T}$ | $\mathcal{U}$ | $\mathcal{V}$ | $\mathcal{W}$ | $\mathcal{X}$ |
| $\mathcal{Y}$ | $\mathcal{Z}$ | | | | |

*Наукове видання*

**Кривий Сергій Лук'янович**

**ВСТУП ДО МАТЕМАТИЧНИХ ОСНОВ ЗАХИСТУ ІНФОРМАЦІЇ**

Редактор Л.В. Магда
Комр'ютерна верстка авторська

Підписано до друку 30.09.2022.
Формат 60×90 1/16. Друк офсет.
Гарнітура TimesNewRoman. Умовн.-друк. арк. 20,46.
Тираж 100 прим. Зам. № 222-10443.

Видавець і вигтовлювач: ВПЦ "Київський університет"
б-р Тараса Шевченка, 14, м. Київ, 01601, Україна
тел. (38044) 239 32 22; (38044) 239 31 72; тел./факс (38044) 239 31 28
e-mail: vpc.div.chief@univ.net.ua; redaktor@univ.net.ua
http: vpc.univ.kiev.ua
Свідоцтво суб'єкта видавничої справи ДК № 1103 від 31.10.02