КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ
КАФЕДРА МАТЕМАТИЧНОЇ ІНФОРМАТИКИ

«ЗАТВЕРДЖУЮ»
Заступник декана
з навчальної роботи
_____ Олена КАШПУР
«__»_____20__ року

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
ОСНОВИ КОМП'ЮТЕРНИХ АЛГОРИТМІВ

для студентів

галузь знань **12 «Інформаційні технології»**
(шифр і назва)
спеціальність **122 «Комп'ютерні науки»**
(шифр і назва спеціальності)
освітній рівень **бакалавр**
(молодший бакалавр, бакалавр, магістр)
освітня програма **«Інформатика»**
(назва освітньої програми)
вид дисципліни **вибіркова**
вибірковий блок **«Інтелектуальні інформаційні технології»**

Форма навчання	**денна**
Навчальний рік	**2022/2023**
Семестр	**7**
Кількість кредитів ECTS	**4**
Мова викладання, навчання та оцінювання	**українська**
Форма заключного контролю	**іспит**

Викладач: **к.ф.-м.н., професор Вергунова І. М.**

Пролонговано: на 20__/20__ н.р. _____(_____) «__»___ 20__р.
(підпис, ПІБ, дата)

на 20__/20__ н.р. _____(_____) «__»___ 20__р.
(підпис, ПІБ, дата)

КИЇВ – 2021

Розробник: Вергунова І. М., канд. ф.-м. н., проф., проф. кафедри математичної інформатики

ЗАТВЕРДЖЕНО
Зав. кафедри «Математичної інформатики»
_____ Василь ТЕРЕЩЕНКО
Протокол № 10 від «24» 04 2024 р.

Схвалено гарантом освітньо-професійної програми «Інформатика»
«6» травня 2024 р._____Людмила ОМЕЛЬЧУК

Схвалено науково-методичною комісією факультету комп'ютерних наук та кібернетики
Протокол від «6» травня 2024 року № 10
Голова науково-методичної комісії _____Людмила ОМЕЛЬЧУК

1. Мета дисципліни – засвоєння базових знань з основ побудови комп'ютерних алгоритмів, включаючи основні поняття та принципи аналізу працеємності комп'ютерних алгоритмів, одержання знань про класи складності задач, набуття вміння будувати раціональні алгоритми для програмних систем з обраної множини алгоритмів заданого класу складності.

2. Попередні вимоги до опанування або вибору навчальної дисципліни:
1. Знати: основи з дисциплін "Програмування", "Математичний аналіз", "Побудова та аналіз алгоритмів".
2. Вміти: застосовувати базові поняття та методи оцінювання функцій та розв'язувати задачі за допомогою фундаментальних алгоритмів, вміти використовувати та одержувати асимптотичні оцінки у вирішенні задач для фундаментальних та деяких вдосконалених алгоритмів.
3. Володіти елементарними навичками: розкладу задач на складові, навичками програмування.

3. Анотація навчальної дисципіни:

Навчальна дисципліна "Основи комп'ютерних алгоритмів" є складовою освітньо-професійної програми підготовки фахівців за першим (*бакалаврським*) рівнем вищої освіти *галузі знань* 12 „Інформаційні технології" зі *спеціальності* 122 „Комп'ютерні науки", *освітньо-професійної програми* „Інформатика".

Дана дисципліна є обов'язковою навчальною дисципліною за ***програмою*** "***Інформатика***".

Викладається у **7** семестрі **3** курсу в **обсязі – 120 год.**

(4 кредити ECTS) зокрема: *лекції – 42 год., консультації – 2 год., самостійна робота – 76 год.* У курсі передбачено **2** *частини* та 2 *контрольні роботи*. Завершується дисципліна – **іспитом у 7 семестрі.**

В результаті вивчення навчальної дисципліни студент повинен:

знати основні поняття та методи аналізу працеємності комп'ютерних алгоритмів, ресурсної ефективності програмної реалізації алгоритму, основні класи складності задач та класифікації комп'ютерних алгоритмів.

вміти оцінювати працеємність алгоритмів за основними параметрами, будувати раціональні алгоритми для програмних систем з обраної множини алгоритмів заданого класу складності з урахуванням ресурсної ефективності програмної реалізації, обґрунтовувати власний погляд на задачу, спілкуватися з колегами з питань проектування та розробки програмних систем.

Для допуску до дисципліни „Основи комп'ютерних алгоритмів" освітньо-професійної програми «Інформатика» студент повинен опанувати компетентності та результати навчання, які надають дисципліни „Побудова та аналіз алгоритмів" та „Програмування" програми «Інформатика». Дисципліна „Основи комп'ютерних алгоритмів" є базовою для засвоєння дисципліни „Складність алгоритмів", дисциплін спеціалізації та дисциплін вільного вибору студента за програмою «Інформатика».

4. Завдання (навчальні цілі):
набуття знань, умінь та навичок (компетентностей) на рівні новітніх досягнень у програмуванні, відповідно до кваліфікації фахівець з інформаційних технологій. Зокрема, розвивати:

- здатність до математичного та логічного мислення, формулювання та досліджування математичних моделей, зокрема дискретних математичних моделей, обґрунтовування вибору методів і підходів для розв'язування теоретичних і прикладних задач у галузі комп'ютерних наук, аналізу та інтерпретування;

- здатність опанувати сучасні методи математичного моделювання об'єктів, процесів і явищ, розробляти моделі й алгоритми чисельного розв'язування задач математичного моделювання з урахуванням похибок наближеного чисельного розв'язування професійних задач;
- здатність до алгоритмічного мислення.

5. **Результати навчання за дисципліною:**

Результат навчання		**Форми (та/або методи і технології) викладання і навчання**	**Методи оцінювання та пороговий критерій оцінювання (за необхідності)**	**Відсоток у підсумковій оцінці з дисципліни**
Код	**Результат навчання**			
РН1.1	Знати основні поняття та методи аналізу працеємності комп'ютерних алгоритмів	Лекція	*Контрольна робота 1, 60% правильних відповідей, іспит*	18%
РН1.2	Знати методи аналізу ресурсної ефективності програмної реалізації алгоритму	Лекція	*Контрольна робота 2, 60% правильних відповідей, іспит*	16%
РН1.3	Знати основні класи складності задач та класифікації комп'ютерних алгоритмів	Лекція	*Контрольна робота 2, 60% правильних відповідей, іспит*	10%
РН2.1	Вміти оцінювати працеємність алгоритмів за основними параметрами	Лекція самостійна робота	*Захист самостійної лабораторної роботи 2, 3, контрольна робота 2, 60% правильних відповідей, іспит*	12%
РН2.2	Вміти будувати раціональні алгоритми для програмних систем з множини алгоритмів заданого класу складності з урахуванням ресурсної ефективності програмної реалізації	Лекція	*Захист самостійної лабораторної роботи 1, 2, 3, контрольна робота 2, 60% правильних відповідей*	20%
РН3.1	Обґрунтовувати власний погляд на задачу, спілкуватися з колегами з питань проектування та розробки програмних систем	Лекція, самостійна робота	*Захист самостійної лабораторної роботи 1, 2, 3*	8%
РН4.1	Організовувати свою самостійну роботу для досягнення результату	Лабораторне заняття, самостійна робота	*Захист самостійної лабораторної роботи 1, 2, 3*	8%
РН4.2	Відповідально ставитися до виконуваних робіт, нести відповідальність за їх якість	Лабораторне заняття, самостійна робота	*Захист лабораторної роботи 1, 2, 3*	8%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання (необов'язково для вибіркових дисциплін, які не входять до блоків спеціалізації)

Результати навчання дисципліни (код) / Програмні результати навчання (назва)	**РН 1.1**	**РН 1.2**	**РН 1.3**	**РН 2.1**	**РН 2.2**	**РН 3.1**	**РН 4.1**	**РН 4.2**

ПРН18.1. Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.	+	+	+	+	+	+	+	+

7. Схема формування оцінки.

7.1. Форми оцінювання студентів:

- семестрове оцінювання:

1. Контрольна робота 1: РН 1.1., РН 2.2 — 15 балів/9 балів.

2. Контрольна робота 2: РН 1.2, Р.Н 1.3, РН 2.1, РН 2.2 - 15 балів/9 балів.

3. Самостійна лабораторна робота 1 (проект): РН 2.2, РН3.1, РН4.1 – 10 балів/6 балів.

4. Самостійна лабораторна робота 2 (проект): РН2.1, РН 2.2, РН3.1, РН4.1, РН4.2 – 10 балів/6 балів.

5. Самостійна лабораторна робота 3 (проект): РН2.1, РН 2.2, РН3.1, РН4.1, РН4.2 – 10 балів/6 балів.

- підсумкове оцінювання (у формі іспиту) *вказується***:**

- максимальна кількість балів які можуть бути отримані студентом: 40 балів;

- результати навчання які будуть оцінюватись: РН1.1, РН1.2, РН1.3, РН2.1;

- форма проведення і види завдань: письмова робота.

Види завдань: 4 письмових завдань.

Критерії оцінювання на іспиті

Завдання	Тема завдання	Максимальний відсоток від 40 балів	Всього відсотків
Завдання 1, 2	Письмове запитання	По 20%	40%
Завдання 3	Завдання з побудови	30%	30%
Завдання 4	Завдання з оцінки	30%	30%
			100%

Запитання для підготовки до іспиту

1. Часова та ємнісна складність комп'ютерного алгоритму. Часова складність та функція працеємності, ємнісна складність та функція об'єму пам'яті.
2. Циклічний надлишковий код. Табличний прямий алгоритм CRC.
3. Циклічний надлишковий код. Основні модифікації прямого алгоритму CRC).
4. Способи побудови комплексних критеріїв оцінки якості комп'ютерних алгоритмів. Врахування вимог точності.
5. Клас задач з поліноміальною складністю (поліноміально-розв'язуваних задач). Клас NP (задач, що поліноміально перевіряються). Клас NPC (NP-повних задач).
6. Способи побудови комплексних критеріїв оцінки якості комп'ютерних алгоритмів. Врахування вимог за часовою стійкістю.
7. Фільтр Блума. Його застосування.
8. Функціональні вартісні коефіцієнти у функції ресурсної ефективності.
9. Методи стиснення даних. Кодування Хафмана. Адаптивний код Хафмана.
10. Класифікація комп'ютерних алгоритмів за вимірнісною чутливістю.
11. Класифікація комп'ютерних алгоритмів за вимогами до додаткової пам'яті.
12. Хеш-таблиці та хеш-функції. Пряма адресація.
13. Класи відкритих та закритих задач і теоретична нижня границя часової складності. Оцінка складності найкращого відомого алгоритму. Класифікація задач за співвідношенням

теоретичної нижньої границі часової складності задачі та складності найбільш ефективного з існуючих алгоритмів.

14. Часові оцінки та прогнозування часу виконання програмних реалізацій на основі функції працеємності. Теоретична побудова часових оцінок.
15. Хешування з ланцюжками. Аналіз продуктивності хешування з ланцюжками. Теореми про вдалий та невдалий пошук при вирішенні колізій методом ланцюжків.
16. Метод типових задач як підхід до одержання часових оцінок програмних реалізацій.
17. Класифікація комп'ютерних алгоритмів за інформаційною чутливістю.
18. Метод пооперaційного аналізу працеємності алгоритму.
19. Універсальне хешування. Побудова універсального класу хеш-функцій.
20. Експериментальний метод одержання часових оцінок на основі функції працеємності.
21. Поліноміальні хеш-функції. Їх застосування.
22. Аналіз впливу вимірності задачі на середній час виконання узагальненої базової операції.
23. NP-повнота та привідність.
24. Модифікований алгоритм зозулі з блочним хешуванням.
25. Поняття теоретичної нижньої границі часової складності алгоритмів (на прикладі алгоритмів сортування). Класифікація задач за співвідношенням теоретичної нижньої границі часової складності задачі та складності найбільш ефективного з існуючих алгоритмів.
26. Циклічний надлишковий код. Дзеркальний алгоритм CRC.
27. Класифікація комп'ютерних алгоритмів та задач за впливом на працеємність особливостей входів. Класифікація за працеємністю в типі L_c.
28. Класифікація комп'ютерних алгоритмів на основі кутової міри асимптотичного зростання функцій.
29. Відкрита адресація. Аналіз методу хешування з відкритою адресацією (вдалий та невдалий пошук).
30. Класифікація комп'ютерних алгоритмів за складністю функції працеємності.
31. Кутова міра асимптотичного зростання функцій. Кутова міра близькості ресурсних оцінок для порівняльного аналізу ресурсних функцій комп'ютерних алгоритмів
32. Класифікація комп'ютерних алгоритмів та задач за впливом на працеємність особливостей входів. Типізація задач та їх комп'ютерних алгоритмів за довжиною входу.
33. Ідеальне хешування. Аналіз методики ідеального хешування.
34. Класифікація комп'ютерних алгоритмів та задач за впливом на працеємність особливостей входів. Підкласи класу PR в типі L_c.
35. Циклічний надлишковий код. Базовий прямий алгоритм CRC.
36. Класифікація комп'ютерних алгоритмів та задач за впливом на працеємність особливостей входів. Класифікації за працеємністю в типі L_n.
37. Модифікації алгоритму зозулі. Хешування із запасом.
38. Класифікація комп'ютерних алгоритмів та задач за впливом на працеємність особливостей входів. Підкласи класу PR в типі L_n.
39. Алгоритм зозулі. Основний алгоритм.
40. Класифікація комп'ютерних алгоритмів та задач за впливом на працеємність особливостей входів. Підкласи класу NPR за ступенем впливу на працеємність параметричної компоненти.
41. Універсальне хешування. Універсальний клас хеш-функцій, його середня продуктивність, побудова хеш-функцій з універсального класу хеш-функцій.
42. Класифікація комп'ютерних алгоритмів за впливом на працеємність особливостей входів. Підкласи класу NPR за характеристичними особливостями множини вихідних даних.
43. Метод порівняльного аналізу алгоритмів за ресурсними функціями.
44. Вимірнісна чутливість алгоритмів за працеємністю.
45. Хешування «класики». Основний алгоритм та модифікації.
46. Вимірнісна чутливість алгоритмів у підкласах класу NPR.
47. Поняття ε-наближеного алгоритму. Оцінка якості ε-наближених алгоритмів. Приклади.
48. Метод аналізу ресурсних функцій комп'ютерних алгоритмів на кінцевому інтервалі.

49. Двовимірні поліноміальні хеш-функції та їх застосування.
50. Циклічний надлишковий код. Дзеркальний табличний алгоритм CRC.
51. Ресурсна характеристика та ресурсна складність комп'ютерного алгоритму. Функції ресурсної ефективності комп'ютерного алгоритму та його програмної реалізації.
52. Стиснення даних. Словникові методи: алгоритм LZ77.
53. Стиснення даних. Словникові методи: алгоритм LZ87.
54. Стиснення зображень. Алгоритм LZW, його застосування.
55. Стиснення чорно/білих зображень. Алгоритм Хафмана з фіксованою таблицею.
56. Стиснення зображень. Алгоритм JPEG.
57. Універсальний клас хеш-функцій, побудова хеш-функцій з універсального класу хеш-функцій.
58. Алгоритми стиснення зображень. Алгоритм Хафмана, його застосування.
59. Циклічний надлишковий код. Алгоритм побітового зсуву.
60. Методи стиснення даних. Арифметичне стиснення.
61. Методи стиснення даних. Адаптивний алгоритм арифметичного стиснення.
62. Стиснення даних. Словникові методи: алгоритм LZSS.
63. Шляхи покращення стиснення для методів родин LZ.
64. Методи контекстного моделювання для стиснення даних.
65. Стиснення даних. Методи контекстного моделювання. Алгоритми PPM.
66. Алгоритми стиснення зображень – адаптивний алгоритм Хафмана.

Студент не допускається до іспиту, якщо під час семестру набрав менше ніж 24 балів. Студент допускається до іспиту за умови виконання 70% передбачених планом лабораторних робіт.

7.2. Організація оцінювання:

Терміни проведення форм оцінювання:

1. Контрольна робота 1: до 10 лекційного заняття.
2. Контрольна робота 2: до 21 лекційного заняття.
3. Самостійна лабораторна робота 1 (проект): до 10 лекційного заняття.
4. Самостійна лабораторна робота 2 (проект): до 10 лекційного заняття.
5. Самостійна лабораторна робота 3 (проект): до 21 лекційного заняття.

Студент має право на одне перескладання кожної контрольної роботи із можливістю отримання максимально 80% початково визначених за цю контрольну роботу балів. Термін перескладання визначається викладачем.

У разі неякісного виконання лабораторної роботи, викладач має право не зарахувати лабораторну роботу, або знизити за неї бали.

Студент має право здавати лабораторні роботи після закінчення визначеного для них терміну, але з втратою одного балу за кожен тиждень, який пройшов з моменту закінчення терміну її здачі.

7.3 Шкала відповідності оцінок

Відмінно / Excellent	90-100
Добре / Good	75-89
Задовільно / Satisfactory	60-74
Незадовільно / Fail	0-59

8. СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
ТЕМАТИЧНИЙ ПЛАН ЛЕКЦІЙ І СЕМІНАРСЬКИХ ЗАНЯТЬ

№ лекції	Назва лекції	Кількість годин		
		Лекції	Лаб. р.	Самост. робота
Частина 1. Алгоритми над простими структурами даних				
1.	**Тема 1.** Алгоритми у зв'язаному списку. *Самостійна робота:* Базові алгоритми у зв'язаному списку. Реалізація динамічних множин. Розробити та реалізувати ефективний алгоритм перетворення розрідженої матриці з двомірного масиву у мультисписок з вузлами з ненульових значень. Завдання 3.66 [3]. Зробити аналіз часової та ємнісної складності алгоритму.	2		4
2.	**Тема 2.** Алгоритми реалізації динамічних множин. *Самостійна робота:* Хешування та побудова хеш-функцій. Виконання лабораторної роботи 1, а. Задача 11.4 [1].	2		4
3.	**Тема 3.** Хешування. Хеш-таблиці та хеш-функції. Пряма адресація. Хешування з ланцюжками. Хеш-функції, їх якість. Прості схеми побудови некріптографічних хеш-функцій. *Самостійна робота:* Степеніві хеш-функції. Виконання лабораторної роботи 1, а.	2		4
4.	**Тема 4.** Універсальне хешування. Побудова універсального класу хеш-функцій. *Самостійна робота:* Реалізація динамічних множин, дерамід. Задача 13.1 (а-г) [1]. Задача 13.4 [1]. Виконання лабораторної роботи 1, б.	2		4
5.	**Тема 5.** Відкрита адресація. Ідеальне хешування. Аналіз. *Самостійна робота:* Відкрита адресація. Застосування. Виконання лабораторної роботи 1, б.	2		4
6.	**Тема 6.** Хешування зозулі. Аналіз. Поліноміальні хеш-функції, їх використання *Самостійна робота:* Алгоритми з використанням поліноміальних хеш-функцій. Виконання лабораторної роботи 1, б.	2		4
7.	**Тема 7.** Алгоритми стиснення. Адаптивні коди Хафмана. Арифметичне кодування. Алгоритм RLE. *Самостійна робота:* Хешування та аутентифікація. Виконання лабораторної роботи 2.	2		4
8.	**Тема 8.** Алгоритми стиснення. Алгоритми LZ77, LZ78. Алгоритм LZW. *Самостійна робота:* Модифікації алгоритмів Алгоритми LZ77, LZ78. Виконання лабораторної роботи 2.	2		4
9.	**Тема 9.** Алгебраїчне дерево обчислень. Часова складність алгоритму та функції працеємності. Ресурсна характеристика та ресурсна складність	2		4

	алгоритму. Виконання лабораторної роботи 2. *Самостійна робота:* Конкурентний аналіз, його реалізація. Виконання лабораторної роботи 2.			
10	*Контрольна робота 1*	1		
Контроль за підсумками лабораторної роботи 1, 2		1		
Всього по частині 1		20		36
Частина 2. Складні структури даних				
11.	**Тема 10.** Ємнісна складність. Ресурсна ефективність комп'ютерних алгоритмів. Функції ресурсної ефективності комп'ютерних алгоритмів та їх програмних реалізацій. *Самостійна робота:* Реалізація алгоритму зворотної перестановки бітів. Виконання лабораторної роботи 3, а.	2		4
12.	**Тема 11.** Алгоритм CRC, його модифікації. Породжуючі поліноми. Застосування. Пошук помилки. *Самостійна робота:* Задача 17.1 [1]. Виконання лабораторної роботи 3, а.	2		4
13.	**Тема 12.** Цифрові фільтри. *Самостійна робота:* Амортизовані збалансовані за вагою дерева. Виконання лабораторної роботи 3, а.	2		4
14.	**Тема 13.** NP-повнота. Абстрактна задача. Клас складності. Поняття про алгоритми верифікації. *Самостійна робота:* Амортизовані збалансовані за вагою дерева. Виконання лабораторної роботи 3, а.	2		4
15.	**Тема 14.** NP-повнота та приводимість. Виконуваність схем. Доведення NP-повноти деяких задач. *Самостійна робота:* Задача 34.1.1, 34.2.1 [1]. Виконання лабораторної роботи 3, а.	2		4
16.	**Тема 15.** Способи побудови комплексних критеріїв оцінки якості комп'ютерних алгоритмів. Часові оцінки та прогноз часу виконання програмної реалізації на основі функції працеємності. Аналіз впливу вимірності задачі на середній час виконання узагальненої базової операції. *Самостійна робота:* Задача 34.3.1-34.3.2 [1]. Виконання лабораторної роботи 3, а.	2		4
17.	**Тема 16.** Спеціальні класифікації алгоритмів. *Самостійна робота:* Задача 34.4.1-34.4.2 [1]. Виконання лабораторної роботи 3, а.	2		4
18.	**Тема 17.** Поняття кутової міри асимптотичного зростання функцій. Застосування. *Самостійна робота:* Застосування кутової міри. Виконання лабораторної роботи 3, б.	2		4
19.	**Тема 18.** Класифікація алгоритмів за врахуванням основних характеристик. *Самостійна робота:* Конкурентний аналіз та самоорганізовувані списки. Виконання лабораторної роботи 3, б.	2		4
20.	**Тема 19.** Методика порівняльного аналізу та раціонального вибору алгоритмів (за ресурсними	2		4

	функціями). *Самостійна робота:* Виконання лабораторної роботи 3, б.			
21	*Контрольна робота 2*	1		
Контроль за підсумками лабораторної роботи 3		1		
Всього по частині 2		22		40
Консультація				
	Всього	**42**		**76**

Загальний обсяг *год*. – **120**, в тому числі:
Лекцій – **42** год.
Консультацій – **2** год.
Самостійна робота **– 76** год.

Умови самостійних лабораторних робіт:
Самостійна лабораторна робота 1:
а: Розробити та реалізувати алгоритм, що з використанням роздільного зв'язування вставляє N випадкових чисел у таблицю розміром N/100, а потім визначає довжину самого довгого та самого короткого списків, N = 103, 104, 105, 106. Завдання 14.19 [3]. Проаналізувати середній час виконання узагальненої базової операції.
б. Реалізувати структуру даних типу «множина рядків» (кількість символів у рядку та максимальна довжина рядків, підтримувані операції, а також максимальна кількість рядків у множині задаються).Реалізувати ефективний пошук повторюваних рядків та поділ їх на групи (однакові рядки) (асимптотична оцінка O(NM+NlogN)). Визначити кількість різних рядків. Виконати аналіз працеємності для різних випадків.
Самостійна лабораторна робота 2: Реалізувати структуру даних фільтр Блума. Елементи-рядки – непусті послідовності довжиною до 15 символів латинських літер. Максимальна кількість елементів у множині не перевищує $n = 1 \cdot 10^6$. Ймовірність хибнопозитивних спрацювань – 1%.
Структура даних повинна підтримувати операції:

- перевірки належності рядка множині;
- додавання рядка до множини (не гарантується, що цього рядка немає у множині).

За бажанням можна реалізувати з операцією вилучення (з підрахунком).

Самостійна лабораторна робота 3:
а. Реалізувати фільтр Блума за умовами варіанту.
б. Нехай задана деяка архітектура комп'ютера, в межах якої необхідно скласти процедуру, що дає ідеальних хеш для заданої множини чисел.
Комп'ютер має $4 \cdot 2^{20}$ байт пам'яті, що доступна тільки для читання, та містить програму, яку буде виконувати комп'ютер, а також дані. Процесор його містить 256 регістрів (від r0 до r255), причому кожен може зберегти 32-бітне ціле число. Перед виконанням процедури значення кожного регістру крім r0 дорівнює 0, r0 містить вхідні дані процедури.
Комп'ютер має спеціальний регістр – instruction pointer (знаходиться окремо). Процедура зберігається в пам'яті та на початку значення цього регістру дорівнює також 0.
Виконання процедури проходить наступним чином. Спершу зчитується номер інструкції, що зберігається в пам'яті за адресою значення instruction pointer, потім зчитуються аргументи інструкції та виконується інструкція. Якщо під час виконання інструкції не трапився jump, то значення instruction pointer збільшується на розмір прочитаної інструкції, включаючи аргументи.
Процедура має бути відображенням з множини деяких заданих n цілих чисел $A=\{a_0,\ldots,a_n\}$ у множину цілих чисел від 0 до $2n - 1$. Процедура має завершатися після не більше ніж 30 виконаних інструкцій (за будь-яких вхідних даних з множини A процедура не повинна виконувати ділення на 0, а також намагатися звертатися за межі діапазону $4 \cdot 2^{20}$ байт пам'яті). Відображення не повинно мати колізій.

Тому, відповідно до архітектури комп’ютера, має виконуватися:

- Якщо процедурі на вхід подати одне з чисел з множини A, вона має завершитися за не більш ніж 30 інструкцій та вивести ціле число від 0 до $2n - 1$;
- Для будь-яких двох різних чисел a_i та a_j з множини A значення на виході повинні бути різними.

Крім того: - всі регістри містять 32-бітні цілі числа; - порядок збереження для всіх чисел: від молодших байт до старших; - для додавання, віднімання та бітових операцій не суттєво, числа знакові чи ні; - команди jump з умовою представляють числа як знакові; - результатом множення або ділення є знакове 64-бітне число, що зберігається у два регістри, регістр, що одержує молодшу частину результату зберігає його як беззнакове число; - ділення та одержання залишку беруть ділене в якості 64-бітних чисел, де молодша частина зберігається як беззнакове число, а старша – із знаком; дільник є знаковим числом, ділення та одержання залишку для знакових чисел відбувається аналогічно архітектурі x86.

Рекомендовані джерела:

Основні:

1. Кормен Т. Алгоритмы. Построение и анализ. 3-е изд. / Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. – М. : ИД "Вильямс", 2013. – 1328 с.
2. Клейнберг Дж., Тардос Е. К48 Алгоритмы: разработка и применение. Классика Computers Science / Дж. Клейнберг, Е. Тардос. : Пер. с англ. Е. Матвеева. — СПб.: Питер, 2016. — 800 с.
3. Блейхут Р. Теория и практика кодов, контролирующих ошибки / Р. Блейхут. – М. : Мир, 1986. – 576 с.
4. Д. Сэломон. Сжатие данных, изображений и звука / Д. Сэломон. – М.: Техносфера, 2014. — 368 с.
5. Ульянов М.В. Ресурсноэффективные компьютерные алгоритмы. / М.В. Ульянов. – М.: Наука, 2007. – 376 с.

Додаткові:

6. Кларк Дж, Кейн Дж. Кодирование с исправлением оцибок в системах цифровой связи. М.: Радио и связь, 1987. – 392 с.
7. Панос Л. Алноритмы для начинающих: Теория и практика для разработчика. / Л. Панос. – М.: ЭКСМО, 2018. – 608 с.
8. Скиена Стивен. Алгоритмы. Руководство по разработке. – СПб.: БХВ-Петербург, 2011. - 722 с.

10. Додаткові ресурси:

Система Online-Judge E-OLIMP: http://www.e-olimp.com.