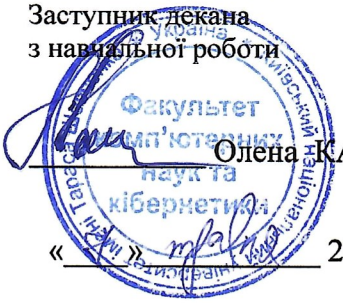
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ

КАФЕДРА ТЕОРІЇ ТА ТЕХНОЛОГІЇ ПРОГРАМУВАННЯ

«ЗАТВЕРДЖУЮ»
Заступник декана
з навчальної роботи



_____ Олена КАШПУР

«____» _____ 2021 року

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
СИСТЕМНЕ ПРОГРАМУВАННЯ
для студентів денної форми навчання

галузь знань	**12 «Інформаційні технології»**
спеціальність	**122 «Комп'ютерні науки»**
освітній рівень	**бакалавр**
освітня програма	**«Інформатика»**
вид дисципліни	**обов'язкова**

Форма навчання	**денна**
Навчальний рік	**2021/2022**
Семестр	**5**
Кількість кредитів ECTS	**4**
Мова викладання, навчання та оцінювання	**українська**
Форма заключного контролю	**іспит**

Викладач: **к.ф.-м.н., доц. Волохов В.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: Волохов Віктор Миколайович, к.ф.-м.н., доцент кафедри «Теорії та технології програмування»

ЗАТВЕРДЖЕНО
Зав. кафедри «Теорії та технології програмування»
_____ Микола НІКІТЧЕНКО
(підпис) (прізвище та ініціали)

Протокол № 10 від «27» квітня 2021 р.

Схвалено гарантом освітньо-професійної програми «Інформатика»

_____ Людмила ОМЕЛЬЧУК «6» травня 2021 року

Схвалено науково-методичною комісією факультету комп'ютерних наук та кібернетики

Протокол від «6» травня 2021 року № 10

Голова науково-методичної комісії _____ Людмила ОМЕЛЬЧУК
(підпис) (прізвище та ініціали)

1. Мета дисципліни. Навчальна дисципліна “Системне програмування” є складовою освітньо-професійної програми підготовки фахівців за освітньо-кваліфікаційним рівнем «бакалавр» галузі знань «12 «Інформаційні технології» з напряму підготовки «Інформатика», спеціальності - 122 «Комп’ютерні науки».

Дана дисципліна базова нормативна за спеціальністю «Комп’ютерні науки».

Викладається у 5 семестрі 3 курсу в **обсязі – 120 год.**

2. Попередні вимоги до опанування або вибору навчальної дисципліни (*за наявності*):

1. Знати: загальні принципи проектування алгортимів обробки нечислової інформації; мови програмування C++ та C# на базовому рівні; технології та методи проектування та програмування.

2. Вміти: розробляти специфікації з урахуванням встановлених вимог; демонструвати процеси та результати професійної діяльності.

3. Володіти елементарними навичками: програмування мовою C++ та C# з використанням інструментальних середовищ розробки програмного забезпечення.

3. Анотація навчальної дисципліни:

Навчальна дисципліна “Системне програмування” 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».

Предмет навчальної дисципліни «Системне програмування» включає в себе розгляд теоретичних аспектів проектування та створення мовних процесорів мов програмування, вивчення відповідних класів граматик, опанування алгоритмів та їх програмування з метою отримання практичних навиків реалізації мовних процесорів.

Дана дисципліна є обов’язковою навчальною дисципліною за програмою “Інформатика”.

Викладається у 5 семестрі 3 курсу в **обсязі – 120 год.**

(4 кредити ECTS) зокрема: *лекції – **40** год., лабораторні – **14** год., консультації – **2** год., самостійна робота – **64** год.* У курсі передбачено ***3*** *частини* та ***2*** *контрольні роботи*. Завершується дисципліна – **іспитом в 5 семестрі.**

4. У результаті вивчення навчальної дисципліни студент повинен:

знати: засоби проектування систем; стандарти моделювання, методи аналізу потреб; методи розробки програмного забезпечення; принципи проектування користувацьких інтерфейсів; інструментальні засоби мови програмування; передові технології; мову програмування Java;

вміти: розробляти та аналізувати граматики мов програмування розробляти, аналізувати та реалізувати алгоритми побудови компонентів мовного процесора, розробляти специфікації з урахуванням встановлених вимог; роз'яснювати і представляти проекти / розробки замовникам з використанням сучасних технологій розробки програмних систем;.

5. Зв’язок з іншими дисциплінами. Дисципліна «Системне програмування» є базовою для засвоєння всіх інших курсів та спецкурсів програміського спрямування, окремих розділів теорії алгоритмів та математичної логіки.

1. **6. Завдання (навчальні цілі):** набуття знань, умінь та навичок (компетентностей) на рівні новітніх досягнень у програмуванні, відповідно освітньої кваліфікації «Бакалавр з комп’ютерних наук».
2. Зокрема:

- здатність оцінювати та забезпечувати якість виконуваних робіт;
- здатність до побудови логічних висновків, використання формальних мов і моделей алгоритмічних обчислень, проектування, розроблення й аналізу алгоритмів, оцінювання їх ефективності та складності, розв'язності та нерозв'язності алгоритмічних проблем для адекватного моделювання предметних областей і створення програмних та інформаційних систем;
- здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління;
- здатність реалізовувати високопродуктивні обчислення на основі хмарних сервісів і технологій, паралельних і розподілених обчислень при розробці й експлуатації розподілених систем паралельної обробки інформації.

7. Результати навчання за дисципліною:

Результат навчання (РН) (1. знати; 2. вміти; 3. комунікація; 4. автономність та відповідальність)		**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**			
РН 1.1	*Знати та освоїти основні алгоритми розробки мовних процесорів*	*Лекція*	*Тест, 60% правильних відповідей, іспит*	*14%*
РН 1.2	*Знати принципи роботи Java-орієнтованого інструментального комплексу розробки програмного забезпечення.*	*Лекція, лабораторне заняття*	*Тест, 60% правильних відповідей, іспит*	*10%*
РН 1.3	*Знати та освоїти принципи функціонування JVM – середовища.*	*Лекція, лабораторне заняття,*	*Захист лабораторної роботи, іспит*	*10%*
РН 2.1	*Вміти працювати з Java-орієнтованим інструментальним комплексом розробки програмного забезпечення.*	*Лабораторне заняття, самостійна робота*	*Захист лабораторної роботи, іспит*	*20%*
РН 2.2	*Вміти працювати пакетом програм розробки компонент мовного процесора*	*Лабораторне заняття, самостійна робота*	*Захист лабораторної роботи*	*20%*
РН 2.3	*Вміти самостійно описувати синтаксис мови програмування з орієнтацією на відповідний клас граматик.*	*Лабораторні роботи, самостійна робота*	*Захист лабораторної роботи*	*20%*
РН 4.1	*Організовувати свою самостійну роботу для досягнення результату*	*Самостійна робота*	*Захист лабораторної роботи*	*6%*

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Результати навчання дисципліни **Програмні результати навчання**	**РН 1.1**	**РН 1.2**	**РН 1.3**	**РН 2.1**	**РН 2.2**	**РН 2.3**	**РН 4.1**
(з опису освітньої програми)							
ПРН13. Володіти мовами системного програмування та методами розробки програм, що взаємодіють з компонентами комп'ютерних систем, демонструвати знання мережних технологій, архітектури комп'ютерних мереж і практичні навички технології адміністрування комп'ютерних мереж та їх програмного забезпечення.	+	+	+	+			
ПРН16. Виконувати паралельні та розподілені обчислення, застосовувати чисельні методи та алгоритми для паралельних структур, мови паралельного програмування при розробці та експлуатації паралельного та розподіленого програмного забезпечення.					+	+	+

7. Схема формування оцінки.

7.1 Форми оцінювання студентів:

- семестрове оцінювання:

1. Контрольна робота (тест): РН 1.1 - 12 балів/ 7,2 балів
2. Контрольна робота (тест): РН1.2, РН 1.3 - 10 балів/ 6 балів
3. Лабораторна робота (2 проекти): РН1.3, РН 2.1, РН4.1 - 14 балів/ 8,4 балів.
4. Лабораторна робота (2 проекти): РН1.3, РН 2.2, РН4.1 - 14 балів/ 8,4 балів.
5. Лабораторна робота (1 проект): РН1.3, РН 2.3, РН4.1 - 10 балів/ 6 балів

- підсумкове оцінювання (у формі іспиту)*:*

- максимальна кількість балів які можуть бути отримані студентом: 40 балів;
- результати навчання які будуть оцінюватись: РН1.1, РН1.2, РН 1.3, РН 2.1.
- форма проведення і види завдань: письмова робота.

Види завдань:4 теоретичних та 4 письмових завдання

Критерії оцінювання екзаменаційної роботи

Завдання	Вид завдання	Максимальний відсоток (бал)	Всього відсотків (балів)
Завдання 1, 2, 3, 4	Письмове запитання з теорії курсу	по 6% (1.6 балів)	50% (20 балів)
Завдання 5, 6, 7,8	Письмове завдання: розв'язування задач на основі теорії курсу	по 5% (2 бали)	50% (20 балів)
Всього			**100% (40 балів)**

Запитання для підготовки до іспиту

1. Аспекти (парадигми) мов програмуванняж
2. Класифікація мов програмування за критерієм «Прагматика».
3. Семантика та синтаксис мов програмування;
4. Означення породжуючої граматик.
5. Класифікація граматик за Хомським.
6. Автоматна характеристика граматик.
7. Термінологія в області формальних мов та граматик.
8. Структура мовного процесора типу транслятор та типу інтерпретатор.
9. Характеристика основних блоків мовного процесора (вхід –вихід).
10. Лексичний аналіз в мовних процесорах. Поняття лексеми.
11. Означення скінченого автомата. Недетерміновані та детерміновані скінчені автомати.
12. Алгоритм пошуку недосяжних станів скінченого автомата.
13. Алгоритм пошуку тупикових станів скінченого автомата.
14. Еквівалентні автомати. Мінімальний еквівалентний скінчений автомат.
15. Праволінійні та ліволінійні граматики.
16. Алгоритм побудови скінченого автомата на основі праволінійної граматики.
17. Алгоритм побудови праволінійної граматики на основі скінченого автомата.
18. Регулярні множини та регулярні вирази. Алгебра регулярних виразів. Тотожності в алгебрі регулярних виразів.

19. ПОЛІЗ регулярного виразу. Інтерпретація ПОЛІЗ регулярного виразу.
20. Програмування скінчених автоматів.
21. Методика розробки лексичних аналізатора мов програмування на основі скінчених автоматів.
22. Інструментальні комплекси автоматизації побудови лексичних аналізаторів мов програмування: LEX, FLEX, BISON, JCC.
23. Контекстно-вільні граматики. Безпосереднє виведення та виведення в граматиці ***G***.
24. Стратегії виведення в граматиці ***G.*** Дерево виведення ланцюжка ***w*** в граматиці ***G.***
25. Аналіз π ланцюжка ***w*** в граматиці ***G.*** Відтворення синтаксичного дерева на основі аналізу π ланцюжка ***w*** (лівостороння стратегія виводу).
26. Алгоритм пошуку ε-нетермінілів.
27. Алгоритм пошуку недосяжних нетерміналів.
28. Алгоритм пошуку непродуктивних правил.
29. Алгоритм пошуку ліво- (право-) рекурсивних нетерміналів.
30. Означення LL(k)-граматики. Конструктивні означення LL(k)-граматики.
31. Наслідки означення LL(k)-граматики.
32. Означення множини $FIRST_k(w)$. Алгоритм пошуку $FIRST_k(w_1w_2)$.
33. Алгоритм пошуку $FIRST_k(A)$, A □ N.
34. Сильні LL(k)-граматики.
35. Означення множини $FOLLOW_k(w)$. Алгоритм пошуку $FOLLOW_k(A)$, A □ N.
36. Таблиця управління LL(1)-синтаксичним аналізом.
37. Алгоритм LL(1)-синтаксичного аналізу.
38. Методика побудови синтаксичних аналізаторів мов програмування за умови, що побудувати LL(1)-граматику неможливо (використання сильної LL(2)-граматики).
39. Означення множини LOCAL(S,A), S-аксіома, A □ N.
40. Алгоритм пошуку множини LOCAL(S,A), S-аксіома, A □ N.
41. LL(k)-синтаксичний аналіз, k > 1.
42. Алгоритм побудови таблиці управляння LL(k)-синтаксичного аналізу, k > 1.
43. Алгоритм LL(k)-синтаксичного аналізатора, k > 1.
44. Семантичний аналіз в мовних процесорах. Форми семантичного терму.
45. ПОЛІЗ семантичного терму програми.
46. Алгоритм побудови семантичного терму програми на основі синтаксичного терму програми.
47. Таблиці мовного процесора. Структура таблиці мовного процесора для програм з блочною структурою.
48. Асемблери. Нотація асемблер програми. Команди та директиви Асемблера.
49. Макроасемблер та макрокоманди.
50. Оптимізація семантичного терму програми. Машинно незалежна та машинно залежна оптимізації.
51. Генерація вихідного коду програми. Методики генерації вихідного коду програми.

Для отримання загальної позитивної оцінки з дисципліни оцінка за іспит не може бути меншою 24 балів

Студент не допускається до іспиту, якщо під час семестру набрав менше ніж 36 балів. Студент допускається до іспиту за умови виконання 70% передбачених планом лабораторних робіт.

Контрольні запитання до частини 1.

1. Поняття класу. Синтаксис визначення класу. Синтаксис визначення методу класу.

2. Відкрита та закрита частина класу. Конструктори класу. Створення класів без можливостей створення об'єктів відповідного класу.
3. Статичні класи. Статичні методи класів.
4. Терміни overload та override: перевантаження методів та їх переписування.
5. Базові та похідні класи.
6. Абстрактні класи. Поліморфізм.
7. Інтерфейси. Конструктори для похідних класів.
8. Поняття виключення. Синтаксис обробки виключення.
9. Проектування власних виключень та їх використання.

Типове завдання контрольної роботи №1.

1. Внутрішні локальні класи. Синтаксис. Приклади.

2. Класи. Визначення класу. Базовий клас та інтерфейси.

3. Не використовуючи інших методів (окрім конструктора, довжина рядка та доступ до і-го символу) запрограмуйте наступні методи класу **String** :

3.1. Boolean **startsWith(String prefix, int toffset)**

Метод повертає true, якщо prefix є префіксом рядка. Що починається з позиції toffset, інакше – false.

3.2. int **compareTo**(String anotherString) //лексикографічне порівняння.

Метод повертає 0 – рядки рівні, >0 – перший рядок «більший», <0– перший рядок «менший»

3.3. int **indexOf(String str)**

Метод повертає індекс першого входження str у рядок.

4. Програміст визначив список об'єктів наступним чином (на перспективу):

ArrayDeque <Object> testDeque;

Фактично в цей список були записані екземпляри класів Date та екземпляри інших класів.

Визначте конструкцію управління для обробки лише даних типу Date.

Контрольні запитання до частини 2.

1. Блочна структура мовного процесора типу інтерпретатор.
2. Алгоритм пошуку недосяжних станів скінченого автомата.
3. Алгоритм мінімізації детермінованого скінченого автомата.
4. Алгоритм розв'язування системи лінійних рівнянь в алгебрі регулярних виразів.
5. Алгоритм інтерпретації ПОЛІЗ регулярного виразу.
6. Методика програмування лексичних аналізаторів на основі скінчених автоматів.

Основні теоретичні питання

1. Скінченні автомати (СА). Детерміновані та недетерміновані СА. Повністю визначені СА. Недетермінізм в скінченних автоматах.
2. Побудувати скінченний автомат М, такий що

$$L(M) = L(M1) * L(M2),$$

де М1 та М2 - скінченні автомати.

3. Означення безпосереднього виведення та виведення в граматиці G.
4. Означення $\sum$ - нетермінала. Алгоритм пошуку $\sum$ - нетерміналів для КС-граматики.
5. Означення мовного процесора. Типи мовних процесорів. Вхідні та вихідні дані інтерпретатора.
6. Регулярні множини та регулярні вирази. Тотожності в алгебрі регулярних виразів. Співвідношення між регулярними множинами та регулярними виразами.
7. Алгоритм побудови скінченого автомата на основі праволінійної граматики.
8. Поняття фази мовного процесора. Одно- та багато-прохідні мовні процесори.
9. Обернений польський запис для арифметичних виразів (ПОЛІЗ). Алгоритм перетворення арифметичного виразу в форму оберненого польського запису.

10. Означення недосяжного стану в скінченому автоматі. Алгоритм пошуку недосяжних станів в скінченому автоматі.
11. Означення породжуючої граматики G. Структура правил в схемі P для різних типів граматик.
12. Означення тупикового стану для скінченого автомата (СА). Алгоритм пошуку множини тупикових станів СА.
13. Вхідні та вихідні дані синтаксичного аналізатора. Способи завдання синтаксичної структури програми.
14. Конфігурації скінченого автомата (СА). Початкова та заключна конфігурація СА. Такт роботи СА. Мова, котру розпізнає (сприймає) СА.
15. Означення еквівалентності двох скінчених автоматів (СА). Алгоритм побудови мінімального СА для даного детермінованого СА.
16. Побудувати скінчений автомат M, такий що

$$L(M) = L(M1) * L(M2),$$

де M1 та M2 - скінченні автомати.

17. Структура мовного процесора інтерпретуючого типу (інтерпретатора).
18. Найменший розв`язок рівняння X=⟨X + ®, де ⟨ та ® - регулярні вирази. Системи лінійних рівнянь з регулярними коефіцієнтами. Метод Гауса розв`язування лінійних рівнянь з регулярними коефіцієнтами.
19. Побудувати скінченний автомат M, такий що

$$L(M) = \{ L(M1) \},$$

де M1 - скінченний автомат.

20. Інтерпретація ПОЛІЗ для регулярного виразу.
21. Означення рекурсивного нетермінала в граматиці G. Алгоритм пошуку ліворекурсивних нетерміналів.
22. Блок лексичного аналізу в мовному процесорі. Вхідні та вихідні дані блока лексичного аналізу.
23.

Основні алгоритми

1. Алгоритм пошуку недосяжних станів скінченого автомата.
2. Алгоритм пошуку тупикових станів скінченого автомата.
3. Алгоритм побудови мінімального еквівалентного даному скінченого автомата.
4. Методика та алгоритми програмування лексичних аналізаторів на основі скінчених автоматів.
5. Алгоритм перетворення виразу у ПОЛІЗ.
6. Алгоритм інтерпретації ПОЛІЗ регулярного виразу.

Контрольні запитання до частини 3.

1. Типи граматик та відповідні їм автомати.
2. Аналіз π, побудова аналізу π на основі лівосторонньої стратегії виводу.
3. Алгоритм відтворення синтаксичного дерева на основі аналізу π.
4. Означення LL(k)- граматики. Наслідки означення.
5. Алгоритм побудови множини First.
6. Алгоритм побудови множини Follow.
7. Алгоритм побудови таблиці управління LL(1) – синтаксичним аналізатором.
8. Алгоритм LL(1) – синтаксичного аналізатора.
9. Алгоритм побудови таблиці управління LL(k) – синтаксичним аналізатором (k>1).
10. Алгоритм LL(k) – синтаксичного аналізатора (k>1).
11. Поняття семантичного терму програми. Співвідношення між семантичним та синтаксичним термами програми.
12. Види семантичного терму програми.
13. ПОЛІЗ для конструкцій управління.
14. Перегляди Асемблера. Таблиці управління на першому перегляді Асемблера.
15. Структура об’єктного модуля програми.

16. Оверлейна структура *.exe – файла. Виконання оверлейного модуля. Управління на рівні операційної системи.
17. Стратегії підвантаження сторінок віртуальної пам'яті в оперативну пам'ять.
18. Стратегії вивантаження сторінок з оперативної пам'яті.
19. Машинно-незалежні алгоритми оптимізації об'єктного коду програми.
20. Машинно-залежні алгоритми оптимізації об'єктного коду програми.

Основні теоретичні питання

1. Алгоритм побудови таблиці управління LL(1) – синтаксичного аналізатора;
2. Алгоритим LL(1) – синтаксичного аналізатора (магазинний автомат);
3. Методика побудови синтаксичного аналізатора за умови невідповідності декількох правил LL(1) – умові.
4. Метод рекурсивного спуску побудови синтаксичного аналізатора для класу LL(1) – граматик.
5. Алгоритм пошуку контексту $Local_k(S,A_i)$, де A_i - нетермінал.
6. Алгоритм перевірки на властивість LL(k) – граматики, де $k > 1$.
7. Алгоритм побудови таблиці управління LL(k) – синтаксичного аналізатора, де $k > 1$.
8. Алгоритм функціонування LL(k) – синтаксичного аналізатора, де $k > 1$.
9. Семантичний терм програми в формі ПОЛІЗ. ПОЛІЗ для конструкцій управління в мові С.
10. Семантичний терм програми в формі дерева. Алгоритм побудови семантичного терму програми і формі дерева.
11. Атрибутний метод побудови семантичного терму програми. Синтезовані та успадковані атрибути.
12. Асемблер. Перший та другий перегляди. Структура об'єктного файла.

Типове завдання контрольної роботи №2.

1. Означення LL(k) – граматики. Наслідки означення LL(k) – граматики;
2. Алгоритм побудови таблиці управління LL(1) – синтаксичного аналізатора;
3. Умови, за яких граматика G буде LL(1) – граматикою.
4. Для граматики з наступною схемою правил знайти ∑-нетермінали:

S □ AB | aAB C □ DA | aAB

A □ qwertyB | AB | ∑ D □ Sa | ∑

B □ CDA | aAB

5. Сформулювати алгоритм пошуку множини Follow k(Ai), Ai □ N.

Для граматики G={N, ©, P, S} з схемою P:

S □ A B | ∑ A □ C b | a

B □ b B | ∑ C □ SA

знайти множини $Followl_2(A_i)$, A_i □ N, де N={S, A, B, C}, © = {a, b}.

6. Для вище означеної граматики побудувати МП-автомат, який моделює лівосторонню стратегію виводу ω в граматиці G.

7.2 Організація оцінювання:

Терміни проведення форм оцінювання:

1. Контрольна робота (тест): до 7 тижня семестру.

2. Контрольна робота (тест): до 12 тижня семестру.

3. Лабораторна робота 1,2 (проект): до 4 тижня семестру.

4. Лабораторна робота 3,4 (проект): до 8 тижня семестру.
5. Лабораторна робота 5 (проект): до 12 тижня семестру.

Студент має право на одне перескладання контрольної роботи із можливістю отримання максимально 5 балів за кожну. Термін перескладання визначається викладачем.
У випадку відсутності студента з поважних причин відпрацювання та перездачі МКР здійснюються у відповідності до „Положення про порядок оцінювання знань студентів при кредитно-модульній системі організації навчального процесу" від 1 жовтня 2010 року.
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 лабораторних занять

№ лек-ції	Назва лекції	Кількість годин		
		Лекції	Лаборат. занять	Сам. р-та
Частина 1. **Об'єктно-орієнтоване програмування. Оглядові лекції з мови програмування Java**				
1.	Поняття об'єкта в мові програмування Java. Структура об'єкта. Відкрита та закрита компоненти об'єкта. Методи - члени об'єкта. Опис та означення функцій - членів об'єкта. Визначення та опис методів класу. Властивості методів у залежності від атрибутів, які їм надаються. Приклади. Самостійна робота: опрацювання лекційного матеріалу.	4	1	6
2.	Поняття класу як приватного типу . Об'єкт типу. Синтаксис визначення типу та об'єкта типу. Конструктори. Метод finalize(). Методи класу. Сигнатура методу. Перевантаження методів. Роль суперкласу при перевантаженні та перевизначенні методів. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	4	1	4
3	Узагальнені масиви. Об'єктні оболонки. Поняття рефлексії. Роль JVM при реалізації рефлексії. Методи рефлексії. Виключення та їх обробка. Ієрархія виключень. Побудова системи власних виключень. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	2	1	4
4	Колекції. Інтерфейси колекцій та відповідна ієрархія класів. Клонування та серілізація об'єктів. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	2	1	4
	Контрольна робота 1			2
Контроль за підсумками лабораторних робіт 1 та 2				
Всього за частиною 1		12	4	20
Частина 2. **Розробка мовних процесорів мов програмування (лексичний аспект).**				
5	Поняття мовного процесора. Типи мовних процесорів. Основні фази мовного процесора. Структура мовного процесора типу транслятор та типу інтерпретатор. Основні блоки мовного процесора.. Перегляди мовного процесора. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	4	1	4
6	Лексичний аналіз. Функції лексичного аналізатора. Вхідні та вихідні структури даних лексичного аналізатора. Скінчені автомати. Способи завдання скінчених автоматів. Недетерміновані та детерміновані скінчені автомати. Алгоритми перетворення недетермінованого скінченого автомата в детермінований. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	4	1	4
7	Регулярні множини та регулярні вирази. Основні тотожності над регулярними виразами. Регулярні множини та скінчені автомати. Алгоритм перетворення регулярного виразу у ПОЛІЗ. Інтерпретація ПОЛІЗ регулярного виразу. Приклади.	4	1	4

	Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.			
8	Праволінійні граматики та скінчені автомати. Побудова скінченого автомата на основі праволінійної граматики. Система лінійних рівнянь в алгебрі регулярних виразів. Метод Гауса. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	2	1	5
9	Програмування скінчених автоматів. Прямий та непрямий лексичний аналіз в мовних процесорах. Побудова лексичних аналізаторів на основі скінчених автоматів. Інструментальні системи побудови лексичних аналізаторів. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	2	1	5
Контроль за підсумками лабораторних робіт 3 та 4				
Всього за частиною 2		16	5	22
***Частина 3*. Розробка мовних процесорів мов програмування (синтаксичний та семантичний аспекти).**				
10	Породжуючі граматики. Класифікація граматик по Хомсьному. Породжуючі граматики та еквівалентні їм класи автоматів. Аналіз ланцюжка W в граматиці G. Побудова дерева виводу на основі лівостороннього аналізу ланцюжка W. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	4	1	4
11	Магазинні автомати. Побудова МП-автомата, що моделює лівосторонній вивід ланцюжка W в граматиці G. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	2	1	4
12	Означення LL(k)-граматики. Властивості LL(k)-граматик. Приклади. Алгоритми побудови множин FIRST(k) та FOLLOW(k). Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	2	1	4
13	Перевірка LL(1)-умови для довільної КС-граматики. Побудова LL(1)-таблиці для управління роботою LL(1)-синтаксичним аналізатором. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	2	1	4
14	Перевірка LL(k)-умови для довільної КС-граматики, k>1. Побудова LL(k)-таблиці для управління роботою LL(k)-синтаксичним аналізатором. Приклади. Виконання лабораторної роботи. Самостійна робота: опрацювання лекційного матеріалу.	2	1	4
	Контрольна робота 2			2
Контроль за підсумками лабораторної роботи 5				
Всього за частиною 3		12	5	22
	ВСЬОГО	40	14	64
Консультація			2	
Іспит				

Загальний обсяг *120 год.,* в тому числі:
Лекцій – **40** *год.*
Лабораторні заняття - 14 *год.*

Консультації – ***2*** *год.*
Самостійна робота **-** ***64*** *год.*

Завдання для самостійної роботи. Проектування та програмування основних функцій для реалізації лабораторної роботи Програмування основних модулів лабораторної роботи. Підготовка тестів. Тестування.

Виконання лабораторних робіт 1-5.
Умови лабораторних робіт:
Лабораторна робота 1: Побудова лексичного аналізатора.
Лабораторна робота 2: Розробити та реалізувати представлення скінченого автомата.
Лабораторна робота 3:Реалізувати лексичний аналізатор мови програмування.
Лабораторна робота 4: Побудова синтаксичного аналізатора.
Лабораторна робота 5**:** Побудова синтаксичного аналізатора.

Деталізовані умови розміщено за посиланнями:
https://drive.google.com/drive/u/1/folders/1igpDLSW2YRCOPZhxdFkjijx93LiO6gwU

9. Рекомендовані джерела:

Основні

1. Java 2. http://www.oracle.com/technetwork/java/javase/documentation/index.html
2. Презентаційні матеріали до оглядових лекцій з мови Java.
3. Ахо А. Ульман Дж. Теория синтаксического анализа, перевода и компиляции. Т1. М. Мир. 1978.
4. Грис Д. Построение компиляторов для ЦЭВМ. М. Мир. 1976.
5. Льюис Ф., Стирнз Р., Розенкранц Д. Теоретические основы построения компиляторов. М. Мир. 1979.
6. Вирт Н. Систематическое программирование. Введение. М.Мир. 1977.
7. В.В. Волохов, Б.І. Бойко, В.Ф. Кузенко, С.С. Шкільняк. Методичні рекомендації до лабораторного практикуму побудови мовних процесорів з курсу „Системне програмування" – К. Київський національний університет імені Тараса Шевченка, 2001. – 52 с.

Додаткові

8. Агафонов В.Н. Синтаксический анализ языков программирования. Новосибирск. Из-во НГУ. 1981.
9. Братчиков И.А. Синтаксис языков программирования. М. Наука. 1975.
10. Вайнгартен Ф. Трансляция языков программирования. М. Мир. 1977.
11. Вирт Н. Систематическое программирование. Введение. М.Мир. 1977.
12. Глушков В.М., ЦейтлинГ.Е., Ющенко Е.Л. Алгебра, языки, программирование. Киев. Наукова думка. 1974.
13. Ингерман П. Синтаксически ориентированный транслятор. М. Мир. 1969.
14. Лебедев В.Н. Введение в системы программирования. М. Статистика. 1975.
15. Миккиман У., Хорнинг Дж., Уортман Д. Генератор компиляторов. М. Статистика. 1980.
16. Пратт Т. Языки программирования: разработка и реализация. М. Мир. 1979.
17. Чантер Р. Проектирование и конструирование компиляторов. М. финансы и статистика. 1984.
18. Бек Д. Введение в системное программирование. М. Мир. 1988.
19. Копитко М.Ф., Іванків К.С. Основи програмування мовою Java: Тексти лекцій. − Львів: Видавничий центр ЛНУ ім. Івана Франка, 2002.− 83 с.
20. Horstman Cay. Core Java SE9 for the Impatient. – Second edition.– Addison Wesley, 2018.– 1818 p/
21. Кадомський К.К., Ніколюк П.К. JAVA. ТЕОРІЯ І ПРАКТИКА. – Вінниця, 2018. https://r.donnu.edu.ua/bitstream/123456789/72/1/JAVA%20ТЕОРІЯ%20І%20ПРАКТИКА.pdf