

**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ  
ІМЕНІ ТАРАСА ШЕВЧЕНКА**  
**ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ**  
**Кафедра теорії та технології програмування**



**РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ  
СИСТЕМНЕ ПРОГРАМУВАННЯ**  
**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 **обов'язкова**

Форма навчання	денна
Навчальний рік	2022/2023
Семестр	5
Кількість кредитів ECTS	4
Мова викладання, навчання та оцінювання	українська
Форма заключного контролю	іспит

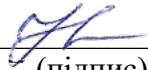
Викладачі: **к.ф.-м.н., доц. Волохов В.М.** (лекції, лабораторні заняття),  
**к.ф.-м.н., асистент Шишацька О.В.** (лабораторні заняття)  
**асистент Поліщук Н.В.** (лабораторні заняття)

Пролонговано: на 20<sup>23</sup>/20<sup>24</sup> н.р. «31» 08 20<sup>23</sup> р.  
на 20<sup>23</sup>/20<sup>24</sup> н.р. « » 20<sup>23</sup> р.

Розробник: Віктор ВОЛОХОВ, к.ф.-м.н., доцент кафедри «Теорії та технології програмування»


ЗАТВЕРДЖЕНО

Зав. кафедри «Теорії та технології програмування»

 Микола НІКІТЧЕНКО  
(підпис)  
(прізвище та ініціали)

Протокол № 1 від «31» серпня 2022 р.

Схвалено Гарантом освітньо-професійної програми «Інформатика»

 Людмила ОМЕЛЬЧУК «31» серпня 2022 року

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

Протокол від «31» серпня 2022 року №1

Голова науково-методичної комісії  Людмила ОМЕЛЬЧУК

© Волохов В.М. 2022 рік

**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 напрямку підготовки «Інформатика», спеціальності - 122 «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 алгоритмів 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 – 42 год., лабораторні – 14 год., консультації – 2 год., самостійна робота – 62 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 грамматики 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.

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

Зокрема:

- здатність бути критичним і самокритичним;
- 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:

<b>Результат навчання (РН)</b> (1. знати; 2. вміти; 3. комунікація; 4. автономність та відповідальність)		<b>Форми (та/або методи і технології) викладання і навчання</b>	<b>Методи оцінювання та пороговий критерій оцінювання (за необхідності)</b>	<b>Відсоток у підсумковій оцінці з дисципліни</b>
<b>Код</b>	<b>Результат навчання</b>			
РН1.1	<i>Знати та освоїти основні алгоритми розробки мовних процесорів</i>	<i>Лекція</i>	<i>Тест, 60% правильних відповідей, іспит</i>	<i>14%</i>
РН1.2	<i>Знати принципи роботи Java-орієнтованого інструментального комплексу розробки програмного забезпечення.</i>	<i>Лекція, лабораторне заняття</i>	<i>Тест, 60% правильних відповідей, іспит</i>	<i>10%</i>
РН1.3	<i>Знати та освоїти принципи функціонування JVM – середовища.</i>	<i>Лекція, лабораторне заняття,</i>	<i>Захист лабораторної роботи, іспит</i>	<i>10%</i>
РН2.1	<i>Вміти працювати з Java-орієнтованим інструментальним комплексом розробки програмного забезпечення.</i>	<i>Лабораторне заняття, самостійна робота</i>	<i>Захист лабораторної роботи, іспит</i>	<i>20%</i>
РН2.2	<i>Вміти працювати пакетом програм розробки компонент мовного процесора</i>	<i>Лабораторне заняття, самостійна робота</i>	<i>Захист лабораторної роботи</i>	<i>20%</i>
РН2.3	<i>Вміти самостійно описувати синтаксис мови програмування з орієнтацією на відповідний клас граматик.</i>	<i>Лабораторні роботи, самостійна робота</i>	<i>Захист лабораторної роботи</i>	<i>20%</i>
РН4.1	<i>Організовувати свою самостійну роботу для досягнення результату</i>	<i>Самостійна робота</i>	<i>Захист лабораторної роботи</i>	<i>6%</i>

**6. Співвідношення результатів навчання дисципліни із програмними результатами навчання**

<b>Результати навчання дисципліни Програмні результати навчання</b>	<b>РН 1.1</b>	<b>РН 1.2</b>	<b>РН 1.3</b>	<b>РН 2.1</b>	<b>РН 2.2</b>	<b>РН 2.3</b>	<b>РН 4.1</b>
<i>(з опису освітньої програми)</i>							
ПРН13, ПРН14. Володіти мовами системного програмування та методами розробки програм, що взаємодіють з компонентами комп'ютерних систем, знати мережні технології, архітектури комп'ютерних мереж, мати практичні навички технології адміністрування комп'ютерних мереж та їх 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): РН 1.2, РН 1.3 | - 10 балів/ 6 балів    |
| 3. Лабораторна робота (2 проекти): РН 2.1   | - 14 балів/ 8,4 балів. |
| 4. Лабораторна робота (2 проекти): РН 2.2   | - 14 балів/ 8,4 балів. |
| 5. Лабораторна робота (1проект): РН 2.3     | -10 балів/ 6 балів     |

#### - підсумкове оцінювання (у формі іспиту):

- максимальна кількість балів які можуть бути отримані студентом: 40 балів;
- результати навчання які будуть оцінюватись: РН 1.1, РН 1.2, 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 екзамену

1. Аспекти (парадигми) мов програмування.
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 скінченного автомата. Недетерміновані та детерміновані скінчені автомати.
12. Алгоритм пошуку недосяжних станів скінченного автомата.
13. Алгоритм пошуку тупикових станів скінченного автомата.
14. Еквівалентні автомати. Мінімальний еквівалентний скінчений автомат.
15. Праволінійні та ліволінійні граматиками.
16. Алгоритм побудови скінченного автомата на основі праволінійної граматиками.
17. Алгоритм побудови праволінійної граматиками на основі скінченного 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 аналізаторів 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  $\pi$  ланцюжка  $w$  в граматиці  $G$ . Відтворення синтаксичного дерева на основі аналізу  $\pi$  ланцюжка  $w$  (лівостороння стратегія виводу).
26. Алгоритм пошуку  $\varepsilon$ -нетерміналів.
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  $FIRST_k(A)$ ,  $A \in N$ .
34. Сильні  $LL(k)$ -граматики.
35. Означення множини  $FOLLOW_k(w)$ . Алгоритм пошуку  $FOLLOW_k(A)$ ,  $A \in N$ .
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,  $A \in N$ .
40. Алгоритм пошуку множини  $LOCAL(S,A)$ ,  $S$ -аксіома,  $A \in N$ .
41.  $LL(k)$ -синтаксичний аналіз,  $k > 1$ .
42. Алгоритм побудови таблиці управління  $LL(k)$ -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 оптимізація.
51. Генерація вихідного коду програми. Методики генерації вихідного коду програми.

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

***Студент не допускається до екзамену, якщо під час семестру набрав менше ніж 20 балів. Студент допускається до іспиту за умови виконання 70% передбачених планом лабораторних робіт.***

### Контрольні запитання до частини 1.

1. Блочна структура мовного процесора типу інтерпретатор.
2. Алгоритм пошуку недосяжних станів скінченного автомата.
3. Алгоритм мінімізації детермінованого скінченного 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 скінчених автоматах.
2. Побудувати скінченний автомат  $M$ , такий що
$$L(M) = L(M1) \cup L(M2),$$
де  $M1$  та  $M2$  - скінченні автомати.
3. Означення безпосереднього виведення та виведення в граматиці  $G$ .
4. Означення  $\epsilon$  - нетерміналу. Алгоритм пошуку  $\epsilon$  - 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 скінченного 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 скінченного автомата (СА). Алгоритм пошуку множини тупикових станів СА.
13. Вхідні та вихідні дані синтаксичного аналізатора. Способи завдання синтаксичної структури програми.
14. Конфігурації скінченного 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 скінченний 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  $X = \alpha X + \beta$ , де  $\alpha$  та  $\beta$  - 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 нетерміналу в граматиці  $G$ . Алгоритм пошуку ліворекурсивних нетерміналів.
22. Блок лексичного аналізу в мовному процесорі. Вхідні та вихідні дані блока лексичного аналізу.



### **Основні алгоритми**

1. Алгоритм пошуку недосяжних станів скінченного автомата.
2. Алгоритм пошуку тупикових станів скінченного автомата.
3. Алгоритм побудови мінімального еквівалентного даному скінченного 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 2.**

1. Типи граматик та відповідні їм автомати.
2. Аналіз  $\pi$ , побудова аналізу  $\pi$  на основі лівосторонньої стратегії виводу.
3. Алгоритм відтворення синтаксичного дерева на основі аналізу  $\pi$ .
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. Машинно-незалежні алгоритми оптимізації об'єктного коду програми.
16. Машинно-залежні алгоритми оптимізації об'єктного коду програми.

### **Основні теоретичні питання**

1. Алгоритм побудови таблиці управління  $LL(1)$  – синтаксичного аналізатора;
2. Алгоритм  $LL(1)$  – 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 файлу.

### **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 граматика з наступною схемою правил знайти  $\epsilon$ -нетермінали:  
$$\begin{array}{ll} S \rightarrow AB \mid aAB & C \rightarrow DA \mid aAB \\ A \rightarrow qwertyB \mid AB \mid \epsilon & D \rightarrow Sa \mid \epsilon \end{array}$$

$$B \rightarrow CDA \mid aAB$$

5. Сформулювати алгоритм пошуку множини Follow  $k(A_i)$ ,  $A_i \in N$ .

Для граматики  $G = \{N, \Sigma, P, S\}$  з схемою P:

$$S \rightarrow AB \mid \varepsilon$$

$$A \rightarrow Cb \mid a$$

$$B \rightarrow bB \mid \varepsilon$$

$$C \rightarrow SA$$

знайти множини Follow  $l_2(A_i)$ ,  $A_i \in N$ , де  $N = \{S, A, B, C\}$ ,  $\Sigma = \{a, b\}$ .

6. Для вище означеної граматики побудувати МП-автомат, який моделює лівосторонню стратегію виводу  $\omega$  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 (проєкт): до 4 тижня семестру.
4. Лабораторна робота 3, 4 (проєкт): до 8 тижня семестру.
5. Лабораторна робота 5 (проєкт): 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 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

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

## 8. Структура навчальної дисципліни. Тематичний план лекцій і лабораторних занять

№ лекцій	Назва лекції	Кількість годин		
		Лекції	Лаборат. занят	Сам. р-та
Частина 1. Об'єктно-орієнтоване програмування. Оглядів лекцій з мови програмування Java				
1.	<b>Тема 1.</b> 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. <i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i>	2		8
2	<b>Тема 2.</b> 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. <i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i>	2	2	4
3	<b>Тема 3.</b> 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. <i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i>	2		4
4	<b>Тема 4.</b> Колекції. Інтерфейси колекцій та відповідна ієрархія класів. Клонування та серіалізація об'єктів. Приклади. <i><b>Самостійна робота:</b> опрацювання лекційного матеріалу.</i>	2	4	4
Контроль за підсумками лабораторних робіт 1 та 2				
Всього за частиною 1		8	6	20
Частина 2. Розробка мовних процесорів мов програмування (лексичний аспект).				
5	<b>Тема 5.</b> 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. <i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i>	2		4

6,7	<p><b>Тема 6.</b> 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</p> <p><i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i></p>	4	2	4
8	<p><b>Тема 7.</b> 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.</p> <p><i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i></p>	2		4
9,10	<p><b>Тема 8.</b> Праволінійні граматики та скінчені автомати. Побудова скінченого автомата на основі праволінійної граматики. Система лінійних рівнянь в алгебрі регулярних виразів. Метод Гаусса. Приклади.</p> <p><i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i></p>	4	2	5
11	<p><b>Тема 9.</b> Програмування скінчених автоматів. Прямий та непрямий лексичний аналіз в мовних процесорах. Побудова лексичних аналізаторів на основі скінчених автоматів. Інструментальні системи побудови лексичних аналізаторів. Приклади.</p> <p><i><b>Самостійна робота:</b> опрацювання лекційного матеріалу.</i></p>	2	2	5
Контрольна робота 1		2		
Контроль за підсумками лабораторних робіт 3 та 4				
Всього за частиною 2		16	6	22
<b>Частина 3. Розробка мовних процесорів мов програмування (синтаксичний та семантичний аспекти).</b>				
12	<p><b>Тема 10.</b> Порождуючі граматики. Класифікація граматик по Хомському. Породжуючі граматики та еквівалентні їм класи автоматів. Аналіз ланцюжка <math>W</math> в граматиці <math>G</math>. Побудова дерева виводу на основі лівостороннього аналізу ланцюжка <math>W</math>. Приклади</p> <p><i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i></p>	2	2	4
13	<p><b>Тема 11.</b> Магазинні автомати. Побудова МП-автомата, що моделює лівосторонній вивід ланцюжка <math>W</math> в граматиці <math>G</math>. Приклади.</p> <p><i><b>Самостійна робота:</b> виконання лабораторної роботи, опрацювання лекційного матеріалу.</i></p>	2		4
14,15	<p><b>Тема 12.</b> Означення <math>LL(k)</math>-граматики. Властивості <math>LL(k)</math>-граматик. Приклади. Алгоритми побудови множин</p>	4		4

	FIRST(k) та FOLLOW(k). Приклади. <i>Самостійна робота: виконання лабораторної роботи, опрацювання лекційного матеріалу.</i>			
16	<b>Тема 13.</b> Перевірка LL(1)-умови для довільної КС-граматики. Побудова LL(1)-таблиці для управління роботою LL(1)-синтаксичним аналізатором. Приклади <i>Самостійна робота: виконання лабораторної роботи, опрацювання лекційного матеріалу.</i>	2		4
17, 18, 19	<b>Тема 14.</b> Перевірка LL(k)-умови для довільної КС-граматики, $k > 1$ . Побудова LL(k)-таблиці для управління роботою LL(k)-синтаксичним аналізатором. Приклади. <i>Самостійна робота: опрацювання лекційного матеріалу.</i>	6		4
	Контрольна робота 2	2		
Контроль за підсумками лабораторної роботи 5				
Всього за частиною 3		18	2	20
	<b>ВСЬОГО</b>	42	14	62
Консультація			2	
Іспит				

**Загальний обсяг 120 год.**, в тому числі:

Лекцій - 42 год.  
Лабораторні заняття - 14 год.  
Консультації - 2 год.  
Самостійна робота - 62 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 скінченного 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/ligpDLSW2YRCOPZhxdFkjjjx93LiO6gwU>

## 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. JavaCC (Java Compiler Compiler) is an open-source parser generator and lexical analyzer generator written in the Java programming language.
4. 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, 2015. – 52 с.