

**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА**

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра програмних систем і технологій

«ЗАТВЕРДЖУЮ»

Заступник декана

з навчально-виховної роботи

Наталія ТМЄНОВА

«29» серпня 2026 року

**РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
ФУНКЦІОНАЛЬНЕ ПРОГРАМУВАННЯ**

для студентів

галузь знань **12 Інформаційні технології**
спеціальність **121 Інженерія програмного забезпечення**
освітній рівень **Бакалавр**
освітня програма **Інженерія програмного забезпечення**

вид дисципліни **вибіркова**

Форма навчання	денна
Навчальний рік	2026/2027
Семестр	7
Кількість кредитів ECTS	7
Мова викладання, навчання та оцінювання	українська
Форма заключного контролю	екзамен

Викладач: Ковалюк Т.В., к.т.н., доц.

Пролонговано:

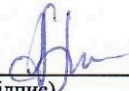
на 20__/20__ н.р. _____ (_____) «__» ____ 20__ р

на 20__/20__ н.р. _____ (_____) «__» ____ 20__ р

КИЇВ – 2026

Розробник: **Ковалюк Тетяна Володимирівна**, кандидат технічних наук, доцент,
доцент кафедри програмних систем і технологій

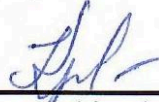
Завідувач кафедри
програмних систем і технологій

 проф. Олексій БИЧКОВ
(підпис)

Протокол № 18 від 17 червня 2026 року

Схвалено науково-методичною комісією факультету інформаційних
технологій

Протокол № 11 від 26 червня 2026 року

Голова науково-методичної комісії  Ганна КРАСОВСЬКА
(підпис)

© Т.В. Ковалюк, 2026 рік
© _____, 20__ рік
© _____, 20__ рік

1. Мета дисципліни

Мета викладання навчальної дисципліни «Функціональне програмування» полягає в тому, щоб сформувати у студентів декларативне мислення, навчити студентів принципів і підходів функціонального програмування; застосовувати засоби функціонального програмування для розв'язання неформалізованих інтелектуальних (управлінських) задач, аналізувати та інтерпретувати отримані результати автоматизованого розв'язання інтелектуальних задач.

2. Попередні вимоги до опанування або вибору навчальної дисципліни

Знати парадигми програмування, зокрема процедурне та об'єктно-орієнтоване програмування; теорію алгоритмів, основи прикладної математики та чисельних методів.

Вміти програмувати, використовуючи процедурний та об'єктно-орієнтований підходи, розв'язувати задачі з математичного аналізу, лінійної алгебри, дискретної математики, чисельних методів.

3. Анотація навчальної дисципліни

Навчальна дисципліна «Функціональне програмування» є складовою освітньо-професійної програми «Інженерія програмного забезпечення» за освітньо-кваліфікаційним рівнем «бакалавр» галузі знань 12 «Інженерія програмного забезпечення» зі спеціальності F2 «Інженерія програмного забезпечення». Предметом навчальної дисципліни є загальні принципи функціонального програмування, базові термінологія та навички програмування на функціональних мовах програмування, класичні задачі та алгоритми їх розв'язання із використанням функціонального підходу у програмуванні, оволодіння технологією розробки програм мовами програмування Scheme та Racket (діалекти Lisp).

В рамках навчальної дисципліни **вивчається:**

- побудова абстракцій за допомогою процедур: процедури і породжувані ними процеси, процедури вищих порядків;
- побудова абстракцій за допомогою даних: подання послідовностей, ієрархічні структури, символічні дані, множинні подання для абстракції даних, системи з узагальненими операціями.

В рамках навчальної дисципліни набуваються практичні **вміння:**

- побудови програм мовами Scheme та Racket,
- тестування та інтерпретація програм мовами Scheme та Racket.
- використання середовищ інтерпретації програм мовами Scheme та Racket.

4. Завдання вивчення дисципліни

В рамках дисципліни «Функціональне програмування» **вивчаються:**

- процедури та процеси, що ними породжуються: лінійна рекурсія та ітерація, деревоподібна рекурсія;
- програмування абстракцій за допомогою процедур вищих порядків: процедури в якості параметрів, побудова процедур за допомогою lambda-форми, процедури як значення, що повертаються;
- введення в абстракцію даних: пари, подання раціональних чисел, селектори та конструктори для пари, операції зі списками;
- ієрархічні структури: подання та відображення дерев, операції обробки дерев;
- символічні дані: символічне подання алгебраїчних виразів, подання множин як неупорядкованих та упорядкованих списків, символічні множини, арифметика многочленів.
- обробка файлів;
- обробка текстів.
- основні типи даних, команди введення і виведення даних;

У результаті вивчення дисципліни «Функціональне програмування» мають бути сформовані такі загальні та фахові компетентності:

Загальні компетентності:

ЗК-1. Здатність до абстрактного мислення, аналізу та синтезу

ЗК-2. Здатність застосовувати знання у практичних ситуаціях

ЗК-5. Здатність вчитися та оволодівати сучасними знаннями та проводити самостійні дослідження на відповідному рівні.

ЗК-6. Здатність до пошуку, оброблення та аналізу інформації з різних джерел.

ЗК-7. Здатність працювати в команді.

Спеціальні (фахові, предметні) компетентності

ФК-1. Здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення.

ФК-2Л Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування.

ФК-3. Здатність розробляти архітектури, модулі та компоненти програмних систем.

ФК-4. Здатність формулювати та забезпечувати вимоги щодо якості програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами.

ФК-5. Здатність дотримуватися специфікацій, стандартів, правил і рекомендацій в професійній галузі при реалізації процесів життєвого циклу.

ФК-7. Володіння знаннями про інформаційні моделі даних, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних.

ФК-13. Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення.

ФК-14. Здатність до алгоритмічного та логічного мислення.

5. Результати навчання за дисципліною:

Результат навчання (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 ¹ (за необхідності)	Відсоток у підсумковій оцінці з дисципліни
Код	Результат навчання			
1.1	Знати основні поняття методології розроблення комп'ютерних програм за парадигмою декларативного програмування. (Тема 1)	Словесний метод (дискусія, співбесіда); наочний метод (метод ілюстрацій і метод демонстрацій); практичний метод (лабораторні заняття, розрахункові роботи); відеометод у сполученні з інформаційними технологіями та комп'ютерними засобами навчання (дистанційні, мультимедійні, веб-орієнтовані тощо); індивідуальні та групові консультації; робота з навчально-методичною літературою та іншими джерелами інформації (конспектування, тезування, анотування, складання реферату, есе); самостійна робота	Експрес-контроль; стандартизовані тести;	5%
1.2	Знати теоретичні засади функціонального програмування у вигляді лямбда форм і процедур першого та вищих порядків (Теми 2, 3)		усний контроль (усне опитування); письмовий контроль (модульні контрольні роботи);	5%
1.3	Знати теоретичні основи рекурсивних процесів та обробки рекурентних співвідношень (Теми 4, 5)		захист лабораторних робіт, STEM-проектів	5%
1.4	Знати теоретичні засади обробки рядків та символів (Тема 6)		контроль рівня виконавських умінь для перевірки практичної підготовки, комп'ютерне тестування за допомогою тестів Google forms; іспит	5%
1.5	Знати теоретичні засади та типові алгоритми обробки пар, списків і векторів (Тема 7)			5%
1.6	Знати теоретичні засади та типові алгоритми обробки ієрархічних структур типу дерев (Тема 8)			5%
1.7	Знати методи обробки текстових файлів (Тема 9)			5%
1.8	Знати технологію символьних обчислень (диференційні рівняння, обчислення поліномів) (Тема 10)			5%
2.1	Вміти працювати в IDE DrRacket та інших середовищах, що реалізують інтерпретацію програм функціональними мовами програмування (Тема 1)		Експрес-контроль; стандартизовані тести; усний контроль (усне опитування); письмовий контроль (модульні контрольні роботи);	5%
2.2	Вміти застосовувати теоретичні засади функціонального програмування у вигляді лямбда форм і процедур першого та вищих порядків (Теми 2, 3)			5%

¹ Критерії оцінювання див. пункт 7.1.

2.3	Вміти програмувати рекурсивні програми, використовувати рекурентні співвідношення (Теми 4, 5)	(індивідуальні творчі завдання, STEM-проекти, розв'язання програмних завдань). Лабораторне завдання	захист лабораторних робіт, контроль рівня виконавських умінь для перевірки практичної підготовки, комп'ютерне тестування за допомогою тестів Google forms; іспит	5%
2.4	Вміти обробляти рядки та символи (Тема 6)			5%
2.5	Вміти застосовувати типові алгоритми обробки пар, списків і векторів (Тема 7)			5%
2.6	Вміти обробляти ієрархічні структури даних (дерева, множини) (Тема 8)			5%
2.7	Вміти працювати з текстовими файлами (Тема 9)			5%
2.8	Вміти застосовувати технологію символьних обчислень (диференційні рівняння, обчислення поліномів) (Тема 10)			5%
3	Комунікація: здатність працювати в команді, вироблення у студентів практичних навиків проведення дослідження і оптимального групового вирішення актуальних проблем із застосуванням відповідних методів і прийомів.	Лабораторне заняття,	захист лабораторних робіт	10%
4	Автономність: продемонструвати розуміння особистої відповідальності за професійні та/або управлінські рішення.	Лабораторне заняття,	захист лабораторних робіт	10%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Результати навчання дисципліни	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	2.1	2.2	2.3	2.4	2.5	2.6	2.7	2.8	3	4
Програмні результати навчання																		
ПР-7. Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення.	+	+	+	+	+	+	+	+	+	+	+	+	+	+				+
ПР-12. Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення.	+	+	+	+	+	+	+	+	+	+	+	+	+	+				+
ПР-14. Застосовувати на практиці інструментальні програмні засоби доменного аналізу, проектування, тестування, візуалізації, вимірювань та документування програмного забезпечення.	+	+	+	+	+	+	+	+	+	+	+	+	+	+				+

7. Схема формування оцінки

Форми оцінювання: рівень досягнення всіх запланованих результатів навчання визначається за результатами написання письмових контрольних робіт і виконання практичних лабораторних завдань.

Питома вага результатів навчання у підсумковій оцінці за умови її опанування на належному рівні така:

результати навчання – 1 (знання)	– до 40%;
результати навчання – 2 (вміння)	- до 40%;
результати навчання – 3 (комунікація)	- до 10%
результати навчання – 4 (автономність)	- до 10%

7.1. Організація оцінювання:

Контроль знань студентів здійснюється за модульно-рейтинговою системою. Результати навчальної діяльності студентів оцінюються за 100-бальною шкалою. Підсумковий контроль (екзамен) проводиться у формі письмової роботи. Максимальна сума балів на іспиті складає 40 балів.

Обов'язковим для іспиту є виконання та захист студентом лабораторних робіт, передбачених робочою програмою навчальної дисципліни, здача двох модульних контрольних робіт. При цьому сумарна мінімальна кількість балів складає 36.

Поточне семестрове оцінювання:

- оцінюванню підлягають дві модульні контрольні роботи, звіти та захист лабораторних робіт, домашніх завдань, експрес-опитування, ініціативні роботи студентів, індивідуальні творчі роботи, STEM-проекти;
- завдання з лабораторних робіт поділяються на обов'язкові та необов'язкові.
- за виконані та захищені в заданий термін лабораторні роботи, здані звіти з лабораторних робіт упродовж семестру студент може отримати 24 бали за виконання обов'язкових лабораторних завдань або 45 балів за виконання обов'язкових і необов'язкових лабораторних завдань. Студенту надається можливість визначити своє навантаження самостійно відповідно до його здібностей, бажання, креативності тощо;
- упродовж семестру орієнтовно на 7 та 12 тижнях навчання проводяться модульні контрольні роботи (МКР) № 1 та № 2, які включають тести та завдання для програмування. Загальна кількість балів за дві МКР складає 15 (7 за МКР1 та 8 за МКР2) балів;
- для розвитку креативності на вибір студентів пропонується індивідуальне творче завдання (STEM-проект) за тематикою інтелектуальної системи з обробкою природної мови, за виконання якого нараховується до 15 балів в залежності від якості та складності завдання;

Критично-розрахунковий мінімум складає 36 балів, які забезпечують студенту доступ до екзамену.

За роботи, здані не в зазначений термін без поважних причин, нараховуються штрафні бали по 10% від балів за лабораторну роботу за кожний тиждень запізнення.

Код лабораторної роботи (СТЕМ-проєкту), який згенерований генеративним штучним інтелектом, не зараховується. Лабораторна робота (СТЕМ-проєкт), згенерований код яких студент не може пояснити, вважається не зданими студентом. Лабораторна робота (СТЕМ-проєкт), згенерований ГШІ код яких студент може пояснити, зараховується зі штрафом у 50%.

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».

Перескладення лабораторної роботи, індивідуального СТЕМ-проєкту чи модульної контрольної роботи з метою підвищення оцінки не допускається.

Оцінювання СТЕМ-проєкту здійснюється за методикою, затвердженою на засіданні кафедри 20.08.2026 р.

Студенти, які набрали упродовж семестр менше 20 балів через невиконання жодної МКР, невідвідування занять, невиконання лабораторних робіт до іспиту не допускаються.

Студенти, які за семестр набрали меншу кількість балів, ніж критично-розрахунковий мінімум (36 балів), але більше 20-ти балів, для доступу до іспиту повинні написати додаткову контрольну роботу, вага якої 20 балів.

Підсумкове оцінювання (у формі екзамену):

- екзаменаційний білет складається з двох частин: теоретична частина містить 1 теоретичне запитання, за яке нараховується 5 балів, та практична частина, яка містить 1 задачу (15 балів) та 15 тестових завдань, які оцінюються в 20 балів;
- студент не допускається до екзамену, якщо під час семестру набрав 20 або менше балів;
- критично-розрахунковий мінімум складає 36 балів;
- максимальний бал за семестр складає 60 балів;
- для отримання загальної підсумкової позитивної оцінки з дисципліни за семестр оцінка за екзамен не може бути меншою за 24 балів;
- максимальна кількість балів, які можуть бути отримані студентом на іспиті, складають 40 балів за 100-бальною шкалою.

7.2. 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

№ п/п	Назва теми	Кількість годин		
		лекції	Лаб заняття	СР

1	Тема 1. Огляд парадигм програмування та мов функціонального програмування: відмінності імперативних програм від декларативних; особливості функціонального програмування; математичні функції як базис функціонального програмування; основні принципи функціонального програмування; генеалогія та порівняння мов програмування.	2	2	10
2	Тема 2. Використання мов Scheme та Racket: основні типи мови Scheme; вирази та значення виразів; комбінаційні форми; інфіксний та префіксний запис арифметичних виразів; імена, зв'язування та оточення; складені процедури; умовні вирази та предикати; елементарні предикати та операції логічної композиції.	2	2	10
3	Тема 3. Рекурсивні та ітераційні процеси обчислення в SCHEME та Racket: форми; процедури; лінійні рекурсія та ітерація; лінійно-рекурсивний процес обчислень; лінійно-ітеративний процес обчислення; особливості реалізації рекурсій; деревоподібна рекурсія; lambda-форма; створення локальних змінних за допомогою форми let.	4	4	10
4	Тема 4. Вирази, стандартні процедури та процедури вищого порядку в SCHEME/Racket: стандартні форми в R5RS Scheme; бібліотечні форми в Scheme; стандартні (вбудовані) процедури в Scheme: предикати, процедури для обробки чисел, процедури перетворення; умовні вирази; процедури вищого порядку: процедури як аргументи, процедури як значення, що повертаються.	4	4	10
5	Тема 5. Введення в абстракцію даних. Списки та дерева: введення в абстракцію даних; пари; ієрархічні дані і властивість замикання; подання послідовностей; операції зі списками; дерева як ієрархічні структури; подання дерев та типові алгоритми їх обробки.	4	4	10
	Модульна контрольна робота 1	0	0	2
6	Тема 6. Комп'ютерна арифметика мовою Scheme (Racket). Раціональні числа в ФП: Поняття пари як складової списку; конструктори та селектори пари; поняття та арифметика раціональних чисел; моделювання арифметики раціональних чисел; створення дробу; додавання та віднімання дробів; множення та ділення дробів; обернення дробу. Прикладні задачі.	4	4	12
7	Тема 7. Арифметика комплексних чисел та програмування, кероване даними мовою Scheme/Racket; моделювання арифметики комплексних чисел; структура загальної системи комплексної арифметики; подання комплексного числа в декартовій системі; подання комплексного числа в полярних координатах; позначені дані. мітки типу; програмування, кероване даними на прикладі обробки комплексних чисел.	4	4	12
8	Тема 8. Присвоєння в мові Scheme (Racket). Черги та Стеки: Внутрішні змінні стану: форма set!, форма begin; приклад моделювання банківських операцій; приклад методу Монте-Карло; переваги та недоліки, пов'язані з введенням присвоювання; спискові структури, що	4	4	12

	змінюється; подання черг (стеків); операції над чергами (стеками)			
9	Тема 9. Структури даних у функціональному програмуванні (вектори, рядки, дерева, хеш-таблиці) : Вектори (процедури для роботи з векторами, приклади обробки векторів); рядки (конструктор рядка, процедури обробки рядків, приклади програми обробки рядків); дерева (приклад дерева, конструктор дерева, селектори дерева, базові операції над деревами); таблиці (операції над таблицями, двовимірні таблиці, операції з двовимірними таблицями)	4	4	12
10	Тема 10. Обробка файлів в Scheme (Racket): поняття файлів і портів; виведення у вихідний порт файлів; відкриття та закриття файлу; запис у файл; читання та запис значень в SCHEME; читання з клавіатури; виведення на екран; запис та читання текстових файлів; приклад статистичної обробки результатів вимірювання із читанням з файлу.	4	4	12
11	Тема 11 .Символьні обчислення: символьна обробка многочленів (поліномів), множини та символьне диференціювання: символи як об'єкти даних; примітиви для роботи з символами; символьне диференціювання; програма диференціювання з абстрактними даними; символьне подання множин; множини як впорядковані символьні списки; множини як бінарні дерева	4	4	12
Підсумкова модульна контрольна робота 2		0	0	2
ВСЬОГО		40	40	126

9. Перелік тем лабораторних робіт

1. Використання рекурсії для організації повторювальних процесів
2. Рекурентні співвідношення для тригонометричних, експоненціальних функцій та ланцюгові дробы
3. Форми lambda та let, вираз присвоєння set! для розв'язання нелінійних рівнянь та чисельного інтегрування функцій
4. Програмування списків мовами функціонального програмування
5. Обробка раціональних та комплексних чисел мовами функціонального програмування
6. Обробка структур типу векторів і матриць, стеків та черг мовами функціонального програмування
7. Обробка рядків та файлів мовами функціонального програмування. Задачі обробки природної мови
8. Символьні обчислення мовами функціонального програмування. Диференціювання та операції з многочленами.

Загальний обсяг 210 год., у тому числі:

Лекції – 40 год.;

Лабораторні заняття – 40 год.;

Самостійна робота – 126 год.

10. РЕКОМЕНДОВАНА ЛІТЕРАТУРА**Основна (Базова)**

1. Harold Abelson and Gerald Jay Sussman. Structure and Interpretation of Computer Programs. The Massachusetts Institute of Technology. 2016. 883 p.
2. George Springer and Daniel P. Friedman. Scheme and The Art of Programming. The MIT Press Cambridge, Massachusetts London, Englan. 2012. 632 p.
3. Kurt Normark. Functional Programming in Scheme with Web Programming Examples. Aalborg University, Denmark. 2003. 219 p.
4. R. Kent Dybvic. The Scheme Programming Language. The MIT Press. 2009. 400 p.
5. Matthew Flatt and Robert Bruce Findler. The Racket Guide. Version 8.7.0.1. 2022. 406 p.

Додаткова

6. Simon Thompson. Haskell. The Craft of Functional Programming. Addison-Wesley. 2018. 504 p.
7. Conrad Barski. Land of Lisp: learn to program in Lisp, one game at a time! No Starch Press, Inc. 2011. 508 p.
8. Allen C., Moronuki J. Haskell Programming from first principles. New York: Christopher Allen and Julie Moronuki, 2016. 1228 p
9. Miran Lipovaca. Learn You a Haskell for Great Good. Access mode: <https://learnyouahaskell.com/chapters>
10. Matthias Felleisen, Robert Bruce Findler, Matthew Flatt, Shriram Krishnamurt. How to Design Programs. MIT Press. 2014. Access mode: <https://htdp.org/2024-11-6/Book/index.html>