

**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА**

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра програмних систем і технологій

«ЗАТВЕРДЖУЮ»

Заступник декана

з навчально-виховної роботи

Наталія ТМЧОВА

«29» серпня 2026 року

**РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
ОСНОВИ ПРОГРАМУВАННЯ**

для студентів

галузь знань **І Інформаційні технології**
спеціальність **І2 Інженерія програмного забезпечення**
освітній рівень **Бакалавр**
освітня програма **Інженерія програмного забезпечення**

вид дисципліни **Обов'язкова**

Форма навчання	денна
Навчальний рік	2026/2027
Семестр	1
Кількість кредитів ECTS	7
Мова викладання, навчання та оцінювання	українська
Форма заключного контролю	екзамен

Викладач: Ковалюк Т.В., к.т.н., доц.

Пролонговано:

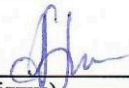
на 20___/20___ н.р. _____ (_____) «___» 20___ р

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

Метою викладання навчальної дисципліни «Основи програмування» є формування у студентів теоретичних знань і практичних навичок використання принципів структурного, процедурного та модульного програмування, опанування основних структурних конструкцій алгоритмічних мов програмування, а також набуття компетентностей, необхідних для розроблення алгоритмів розв'язання прикладних задач, їх реалізації засобами сучасних мов програмування, налагодження програмного забезпечення та оцінювання достовірності отриманих результатів.

2. Попередні вимоги до опанування або вибору навчальної дисципліни

Студент повинен:

знати:

- теоретичні основи інформатики, математики та геометрії за програмою закладу загальної середньої освіти;
- основи математичного аналізу, аналітичної геометрії та лінійної алгебри в обсязі програми закладу вищої освіти.

уміти:

- розв'язувати задачі з математичного аналізу, лінійної алгебри та дискретної математики;
- використовувати базові математичні методи для аналізу й розв'язання прикладних задач.

володіти:

- навичками роботи з персональним комп'ютером;
- навичками роботи в операційних системах Windows, macOS;
- практичними навичками використання програм пакета Microsoft Office для підготовки текстових документів, роботи з електронними таблицями та створення презентацій.

3. Анотація навчальної дисципліни

Навчальна дисципліна «Основи програмування» є обов'язковою складовою освітньо-професійної програми «Інженерія програмного забезпечення» та забезпечує формування базових знань і практичних навичок, необхідних для подальшого вивчення професійно орієнтованих дисциплін.

Предметом навчальної дисципліни є теоретичні засади та практичні методи застосування базових алгоритмічних конструкцій і структур даних із використанням сучасних технологій розроблення програмного забезпечення.

Під час вивчення дисципліни розглядаються основні принципи та етапи процедурно-орієнтованого (структурного) програмування, технологія розроблення програм

алгоритмічними мовами високого рівня, підходи до проєктування, аналізу та реалізації алгоритмів розв'язання математичних й інформаційних задач, а також можливості сучасних інтегрованих середовищ розроблення програмного забезпечення, зокрема **Visual Studio Community**.

Особливу увагу приділено побудові алгоритмів пошуку, сортування та опрацювання даних різних типів, використанню сучасних засобів програмування, формуванню навичок написання якісного програмного коду та розвитку алгоритмічного мислення.

Практична підготовка студентів здійснюється шляхом виконання циклу лабораторних робіт, організованого за проєктно-орієнтованим інкрементальним підходом, який передбачає поетапне розроблення, розширення та вдосконалення єдиного програмного проєкту.

Важливою складовою навчальної дисципліни є формування культури програмування, дотримання принципів академічної доброчесності, вимог до структуризації програмного коду, стандартів кодування та документування програм відповідно до сучасної професійної практики розроблення програмного забезпечення мовами C/C++.

4. Завдання вивчення дисципліни

У межах вивчення навчальної дисципліни студенти опановують:

- основні типи даних, засоби введення та виведення інформації;
- розгалужені обчислювальні процеси та оператори розгалуження алгоритмічних мов програмування;
- циклічні обчислювальні процеси та оператори циклу;
- технологію використання функцій і застосування рекурентних співвідношень;
- рекурсивні алгоритми та особливості їх реалізації за допомогою рекурсивних функцій;
- технології опрацювання структурованих типів даних: масивів, рядків і структур;
- методи опрацювання текстових і бінарних файлів;

У результаті опанування навчальної дисципліни «Основи програмування» у здобувачів вищої освіти мають бути сформовані такі загальні та спеціальні (фахові) компетентності.

Загальні компетентності:

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

ЗК-4. Здатність спілкуватися іноземною мовою як усно, так і письмово.

ЗК-5. Здатність вчитися та оволодівати сучасними знаннями.

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

ЗК-10. Здатність діяти соціально відповідально та свідомо.

Спеціальні (фахові) компетентності:

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

СК-3. Здатність розробляти архітектуру, модулі та компоненти програмних систем.

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

СК-10. Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом усього життя.

СК-12. Здатність здійснювати процес інтеграції системи, застосовувати стандарти і процедури управління змінами для підтримки цілісності загальної функціональності і надійності програмного забезпечення.

СК-14. Здатність до алгоритмічного та логічного мислення.

5. Результати навчання за дисципліною

1. Знання

1.1. Знати основні поняття та принципи технології розроблення комп'ютерних програм. *(Тема 1).*

1.2. Знати призначення, принципи використання та особливості застосування основних операторів алгоритмічних мов програмування (оператори присвоєння, розгалуження, циклу, введення та виведення даних). *(Теми 2–3).*

1.3. Знати теоретичні засади процедурного програмування, механізми виклику функцій, принципи використання рекурентних співвідношень та особливості реалізації рекурсивних процесів. *(Тема 4).*

1.4. Знати теоретичні засади побудови та реалізації рекурсивних алгоритмів. *(Тема 5).*

1.5. Знати теоретичні засади опрацювання масивів і застосування типових алгоритмів роботи з ними. *(Тема 6).*

1.6. Знати принципи використання покажчиків і посилань у мовах програмування C/C++. *(Тема 7).*

1.7. Знати теоретичні засади опрацювання символьних даних і рядків. *(Тема 8).*

1.8. Знати теоретичні засади опрацювання структурованих типів даних та алгоритми роботи зі структурами. *(Тема 9).*

1.9. Знати принципи організації роботи з текстовими та бінарними файлами, *(Тема 10).*

2. Уміння

2.1. Уміти працювати в інтегрованому середовищі розроблення Microsoft Visual Studio та використовувати його інструменти під час створення, налагодження й тестування програм. *(Тема 1).*

2.2. Уміти реалізовувати алгоритми з використанням базових логічних конструкцій програмування: послідовності, розгалуження та циклу. (Теми 2–3).

2.3. Уміти розробляти програми з використанням функцій, рекурсії та рекурентних співвідношень. (Теми 4–5).

2.4. Уміти реалізовувати алгоритми опрацювання одновимірних і багатовимірних масивів. (Тема 6).

2.5. Уміти застосовувати покажчики та посилання під час розроблення програм мовами C/C++. (Тема 7).

2.6. Уміти опрацьовувати символьні дані та рядки засобами мов програмування C/C++. (Тема 8).

2.7. Уміти розробляти програми з використанням структур, масивів структур, а також реалізовувати операції введення, виведення й опрацювання текстових і бінарних файлів. (Теми 9–10).

2.8. Уміти тестувати та налагоджувати програмні проєкти, аналізувати результати їх виконання та інтегрувати нову функціональність у попередню версію програмного проєкту.

3. Комунікація

Здатність ефективно працювати індивідуально та в команді під час виконання програмних проєктів, аргументовано представляти результати власної роботи, брати участь у колективному розв'язанні професійних завдань, застосовуючи сучасні засоби комунікації та співпраці.

4. Автономність і відповідальність

Здатність самостійно приймати обґрунтовані рішення під час розроблення програмного забезпечення, оцінювати результати власної діяльності, нести відповідальність за прийняті технічні рішення, дотримуватися принципів академічної доброчесності, професійної етики та культури програмування.

Таблиця 1 –Результати, форми навчання та методи оцінювання

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 підсумков ій оцінці з дисциплін и
Код	Результат навчання			
1.1	Знати основні поняття та принципи технології розроблення комп'ютерних програм. (Тема 1)	Словесний метод (дискусія, співбесіда);	Експрес-контроль; стандартизовані тести;	5 %
1.2	Знати призначення, принципи використання та особливості застосування основних операторів алгоритмічних мов програмування (оператори присвоєння, розгалуження, циклу, введення та виведення даних). (Теми 2–3).	наочний метод (метод ілюстрацій і метод демонстрацій); практичний метод (лабораторні заняття,	усний контроль (усне опитування); письмовий контроль (модульні контрольні роботи); захист інкрементів проєктно-орієнтова	5 %

¹ Критерії оцінювання див. пункт 7.1.

1.3	Знати теоретичні засади процедурного програмування, механізми виклику функцій, принципи використання рекурентних співвідношень та особливості реалізації рекурсивних процесів. (Тема 4).	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, індивідуальні творчі завдання (STEM проекти).	ного інкрементального проєкту (лабораторних робіт), захист індивідуального творчого завдання (STEM проєктів); контроль рівня виконавських умінь для перевірки практичної підготовки, комп'ютерне тестування за допомогою тестів Google forms; іспит	5 %
1.4	Знати теоретичні засади побудови та реалізації рекурсивних алгоритмів. (Тема 5).			5 %
1.5	Знати теоретичні засади опрацювання масивів і застосування типових алгоритмів роботи з ними. (Тема 6).			5 %
1.6	Знати принципи використання показників і посилань у C/C++ (Тема 7)			5 %
1.7	Знати теоретичні засади опрацювання символьних даних і рядків. (Тема 8).			5 %
1.8	Знати теоретичні засади опрацювання структурованих типів даних та алгоритми роботи зі структурами. (Тема 9).			5 %
1.9	Знати принципи організації роботи з текстовими та бінарними файлами (Тема 10).			5 %
2.1	Уміти працювати в інтегрованому середовищі розроблення Microsoft Visual Studio та використовувати його інструменти під час створення, налагодження й тестування програм. (Тема 1).	Експрес-контроль; стандартизовані тести; усний контроль (усне опитування); письмовий контроль (модульні контрольні роботи); захист лабораторних робіт, захист індивідуального творчого завдання, STEM проєктів; контроль рівня виконавських умінь для перевірки практичної підготовки, комп'ютерне тестування за допомогою тестів Google forms; іспит		5 %
2.2	Уміти реалізовувати алгоритми з використанням базових логічних конструкцій програмування: послідовності, розгалуження та циклу. (Теми 2–3).			5 %
2.3	Уміти розробляти програми з використанням функцій, рекурсії та рекурентних співвідношень. (Теми 4–5).			5 %
2.4	Уміти реалізовувати алгоритми опрацювання одновимірних і багатовимірних масивів. (Тема 6).			5 %
2.5	Уміти застосовувати показники та посилання під час розроблення програм мовами C/C++. (Тема 7).			5 %
2.6	Уміти опрацьовувати символьні дані та рядки засобами мов програмування C/C++. (Тема 8).			5 %
2.7	Уміти розробляти програми з використанням структур, масивів структур, а також реалізовувати			5 %

	операції введення, виведення й опрацювання текстових і бінарних файлів. (Теми 9–10).			
2.8	Уміти тестувати та налагоджувати програмні проекти, аналізувати результати їх виконання та інтегрувати нову функціональність у попередню версію програмного проекту.	Інкременти (лабораторні роботи) проектно-орієнтованого інкрементального навчання; практичне тестування й налагодження програм; аналіз результатів; поетапна інтеграція нової функціональності	Захист інкрементів проектно-орієнтованого інкрементального проекту (лабораторних робіт),	5 %
3	Комунікація: здатність ефективно працювати індивідуально та в команді під час виконання програмних проектів, аргументовано представляти результати власної роботи, брати участь у колективному розв'язанні професійних завдань, застосовуючи сучасні засоби комунікації та співпраці.	Лабораторне заняття, індивідуальне творче завдання, STEM проекти;	захист лабораторних робіт, захист індивідуального творчого завдання, STEM проектів	10 %
4	Автономність: здатність самостійно приймати обґрунтовані рішення під час розроблення програмного забезпечення, оцінювати результати власної діяльності, нести відповідальність за прийняті технічні рішення, дотримуватися принципів академічної доброчесності, професійної етики та культури програмування.	Лабораторне заняття, індивідуальне творче завдання (STEM-проекти)	Захист інкрементів проектно-орієнтованого інкрементального проекту (лабораторних робіт), захист індивідуального творчого завдання (STEM-проектів)	5 %

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Таблиця 2 – Результати навчання з дисципліни та програмні результати навчання згідно із стандартом вищої освіти

Результати навчання дисципліни	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.1	2.2	2.3	2.4	2.5	2.6	2.7	2.8	3	4
Програмні результати навчання																			
ПРН3. Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення.	+	+	+	+	+	+	+	+	+		+	+	+	+	+	+	+	+	+
ПРН7. Знати і застосовувати на практиці фундаментальні концепції,	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+

Результати навчання дисципліни	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.1	2.2	2.3	2.4	2.5	2.6	2.7	2.8	3	4
Програмні результати навчання																			
парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення.																			
ПРН15. Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення.	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+

7. Схема формування оцінки

Форми оцінювання. Рівень досягнення запланованих результатів навчання визначається за результатами поточного та підсумкового оцінювання, зокрема виконання письмових контрольних робіт, лабораторних робіт, домашніх завдань та інших видів навчально-виховної діяльності, передбачених робочою програмою.

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

- результати навчання – 1.1 – 1.9 (знання) – до 5 % за кожен, разом до 45%;
- результати навчання – 2 (вміння) – до 40 %;
- результати навчання – 3 (комунікація) – до 10 %
- результати навчання – 4 (автономність) – до 5 %

Разом – 100 %.

7.1. Організація оцінювання

Контроль результатів навчання здобувачів вищої освіти здійснюється за **модульно-рейтинговою системою**.

Результати навчально-виховної діяльності оцінюються за **100-бальною шкалою**.

Навчальна робота протягом семестру поділяється на два змістові модулі:

- змістовий модуль 1 (ЗМ1) — теми 1–5;
- змістовий модуль 2 (ЗМ2) — теми 6–10.

Максимальна кількість балів за роботу в семестрі становить 60 балів.

Підсумковий контроль проводиться у формі **письмового екзамену**. Максимальна кількість балів за екзамен становить **40 балів**.

Поточне семестрове оцінювання

Поточне оцінювання результатів навчання протягом семестру (табл. 3) передбачає:

- виконання та захист двох модульних контрольних робіт (МКР);
- виконання та захист інкрементів відповідно до проєктно-орієнтованого інкрементального підходу (лабораторних робіт);
- виконання домашніх завдань;
- експрес-опитування;
- виконання індивідуальних творчих завдань (STEM-проєктів);
- виконання додаткових ініціативних завдань, зокрема олімпіадних задач.

Підсумковий бал за кожний інкремент визначається на основі інтегрального показника з урахуванням:

- функціональної повноти та відповідності поставленому завданню;
- коректності алгоритмів і програмної реалізації;
- якості та структурованості програмного коду;
- коректності інтеграції нового інкременту з попередньою версією проєкту;
- тестування та налагодження програмного продукту;
- якості звітної документації;
- результатів захисту та здатності обґрунтувати прийняті програмні рішення;
- дотримання встановлених термінів виконання.

За виконання та захист у встановлені терміни лабораторних робіт (інкрементів відповідно до проєктно-орієнтованого інкрементального підходу) і подання звітів за кожним інкрементом протягом семестру здобувач вищої освіти може отримати:

- 24 бали — за виконання обов'язкових інкрементів проєкту (лабораторних робіт);
- до 34 балів — за виконання обов'язкових (лабораторних робіт) і необов'язкових інкрементів проєкту

Такий підхід дає змогу здобувачеві вищої освіти самостійно визначати обсяг додаткового навчального навантаження відповідно до власних можливостей, інтересів і прагнення до поглиблення знань та практичних навичок.

Модульні контрольні роботи

Протягом семестру проводяться дві модульні контрольні роботи:

МКР № 1 — за матеріалом змістового модуля 1 (теми 1–5), орієнтовно на 7-му тижні навчання;

МКР № 2 — за матеріалом змістового модуля 2 (теми 6–10), орієнтовно на 12-му тижні навчання.

Кожна модульна контрольна робота містить одне теоретичне запитання, на яке потрібно дати правильну відповідь (1 бал), фрагмент коду, який потрібно проаналізувати (2 бали), 15 тестових завдань (3 бали). Отже, максимальна кількість балів за одну МКР становить **6 балів**.

У разі відсутності здобувача вищої освіти з поважних причин відпрацювання та перескладання модульних контрольних робіт здійснюються відповідно до чинних нормативних документів Університету.

Домашні завдання

Протягом семестру передбачено виконання **40 домашніх вправ**, за які можна отримати до **4 балів** — по **0,1 бали за кожну вправу**.

Із 40 вправ **20 є обов'язковими**. Здобувач вищої освіти самостійно визначає, які додаткові вправи виконувати.

Індивідуальні творчі завдання (STEM-проекти)

Для розвитку творчого та критичного мислення здобувачам вищої освіти пропонується на вибір виконання **індивідуального творчого завдання (STEM-проекту)**. За виконання такого завдання може бути нараховано до **8 балів** залежно від складності та якості виконаної роботи.

Ініціативна навчальна діяльність

За самостійну ініціативну роботу, зокрема розв'язання олімпіадних задач рівня АСМ із відкритих ресурсів, перелік яких наведено в методичних вказівках до лабораторних занять, може бути нараховано **від 0,1 до 0,5 бала за кожну задачу** залежно від її складності.

Максимальна кількість додаткових балів за цей вид діяльності становить **5 балів**. Виконання олімпіадних задач не є обов'язковим.

Таблиця 3 – Види семестрових робіт та їх оцінювання

№ п.п.	Види семестрових робіт та їх оцінювання в семестрі			
	Обов'язкові роботи		Необов'язкові роботи	
	Назви робіт	Оцінювання (бали)	Назви робіт	Оцінювання (бали)
1	Виконання та захист інкрементів відповідно до проектно-орієнтованого інкрементального підходу. (лабораторні роботи). Обов'язкова частина	24	Виконання та захист інкрементів відповідно до проектно-орієнтованого інкрементального підходу. (лабораторні роботи). Необов'язкова частина	10
2	Дві модульні контрольні роботи	12		
4	Домашні завдання. Обов'язкова частина	2	Домашні завдання. Необов'язкова частина	2
5	—	0	Індивідуальне творче завдання (STEM-проект)	8
6	—	0	Олімпіадні задачі	5
Разом		38		До 25

Для отримання максимальної кількості 60 балів за результатами семестрової роботи здобувач вищої освіти має виконати всі обов'язкові види

навчально-виховної діяльності з дисципліни «Основи програмування», за які передбачено 38 балів. Решту 22 балів здобувач може набрати за рахунок виконання необов'язкових видів навчально-виховної діяльності на власний вибір, без необхідності виконання всіх запропонованих додаткових завдань. Якщо сума балів, отриманих за виконання обов'язкових та необов'язкових видів навчально-виховної діяльності з дисципліни «Основи програмування», перевищує 60 балів, до семестрового рейтингу зараховується не більше 60 балів.

Умови допуску до екзамену

За результатами роботи протягом семестру здобувач вищої освіти може набрати **до 60 балів**.

Критично-розрахунковий мінімум становить 36 балів, що забезпечує допуск до екзамену.

За роботи, подані із запізненням без поважних причин, застосовуються штрафні бали у розмірі 10 % від кількості балів, передбачених за відповідну лабораторну роботу, за кожний тиждень прострочення.

Перескладання лабораторної роботи, індивідуального завдання або модульної контрольної роботи з метою підвищення оцінки не допускається.

Здобувачі вищої освіти, які набрали від 21 до 35 балів, можуть виконати додаткову контрольну роботу для досягнення мінімального порогу 36 балів.

Здобувачі, які набрали 20 або менше балів, до екзамену **не допускаються** відповідно до встановленого порядку організації освітнього процесу в Університеті.

Нормативні засади використання генеративного штучного інтелекту

Нижче подані витяги із Закону України «Про академічну доброчесність» (№ 4742-IX), статті 7, 8, 9 та 29.

Здобувачі вищої освіти у своїй навчальній діяльності зобов'язані дотримуватися таких правил:

- виконувати кожне навчальне завдання самостійно, крім випадків, якщо його виконання передбачає участь декількох осіб та/або правилами виконання відповідних завдань дозволено отримання допомоги від інших осіб, використання допоміжних матеріалів та засобів, у тому числі з інтернет-ресурсів;
- у разі використання у власному академічному творі результатів інтелектуальної творчої діяльності іншого автора (текстів, зображень, ідей, розробок, тверджень, відомостей, думок тощо) зазначати посилання на відповідне джерело інформації та/або відповідного автора;
- у разі використання в академічному творі об'єкта, згенерованого штучним інтелектом, автор повинен повідомити про це в такому творі із зазначенням методики генерування та/або посилання на відповідну комп'ютерну програму чи її опис згідно з вимогами щодо оформлення та/або оприлюднення відповідних академічних творів, визначених відповідним суб'єктом академічної діяльності;

- оцінюванню підлягають персональні (індивідуальні) результати діяльності особи, досягнуті нею особисто, або результати діяльності авторського колективу, що їх отримав;
- недоброчесне використання результатів, згенерованих штучним інтелектом, це оприлюднення текстів, зображень, моделей, даних, інших результатів, згенерованих штучним інтелектом, як результатів власної академічної діяльності, якщо цей факт не зазначено в академічному творі чи супровідних матеріалах до нього;
- за недоброчесне використання результатів, згенерованих штучним інтелектом, до особи застосовуються санкції, визначені внутрішніми актами, умовами конкурсу, якщо інше не встановлено законом.

Використання генеративного ШІ під час виконання лабораторних робіт

Код інкременту (лабораторної роботи), згенерований засобами генеративного ШІ, **не зараховується** за таких умов:

- здобувач вищої освіти не може пояснити принцип роботи програми;
- здобувач вищої освіти не може обґрунтувати використані програмні рішення;
- здобувач вищої освіти не включив у код документаційні та пояснювальні коментарі для доведення свого розуміння згенерованого коду;
- здобувач вищої освіти не зазначив посилання на використаний інструмент генеративного ШІ.

Часткове зарахування інкременту проєкту (лабораторної роботи) можливе за таких умов:

- якщо здобувач вищої освіти може пояснити згенерований код, обґрунтувати використані програмні рішення, код може бути зарахований із **50 % штрафом** за умови наявності документаційних і пояснювальних коментарів і посилання на використаний інструмент генеративного ШІ;
- якщо генеративний ШІ використовувався для виправлення логічних або синтаксичних помилок чи рефакторингу власного коду здобувача, робота може бути зарахована із **30 % штрафом** за умови наявності пояснювальних коментарів і посилання на використаний інструмент.

7.2 Підсумкове оцінювання

Підсумкове оцінювання здійснюється у формі **екзамену**.

Екзаменаційний білет складається з:

- **теоретичної частини** – 1 теоретичне запитання, максимальна оцінка **5 балів**;
- **практичної частини** – 1 задача, максимальна оцінка **15 балів**;
- **20 тестових завдань** – максимальна оцінка **20 балів**.

Максимальна кількість балів за екзамен становить **40 балів**.

Для отримання позитивної підсумкової оцінки з дисципліни оцінка за екзамен має становити **не менше 24 балів**.

Отже:

- максимальна кількість балів за семестр – 60 балів;
- максимальна кількість балів за екзамен – 40 балів;
- максимальна підсумкова оцінка – 100 балів;
- мінімальна кількість балів за семестр для допуску до екзамену (критично-розрахунковий мінімум) – 36 балів;
- мінімальна кількість балів за екзамен для позитивної підсумкової оцінки – 24 бали;
- кількість балів, за яких здобувач вищої освіти не допускається до іспиту, – 20 або менше балів.

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

№ п/ п	Назва теми	Кількість годин		
		лекції	Лаб занятт я	СР
Змістовий модуль 1. Основи мови C/C++				
1	Тема 1. Середовище програмування Visual Studio. Компілятори, інтерпретатори, технологія розроблення комп'ютерних програм	2	4	8
2	Тема 2. Основні типи даних в алгоритмічних мовах C/C++	2	4	8
3	Тема 3. Призначення, принципи використання та особливості застосування основних операторів алгоритмічних мов програмування (оператори присвоєння, розгалуження, циклу, введення та виведення даних)	2	6	12
4	Тема 4. Теоретичні основи процедурного програмування, механізм виклику функцій, обробка рекурентних співвідношень	4	6	14
5	Тема 5. Теоретичні та практичні основи рекурсивних процесів	4	6	10
	Модульна контрольна робота 1			2
Змістовий модуль 2. Програмування з використанням структурованих типів даних				
6	Тема 6. Теоретичні засади обробки масивів. Практичні основи застосування типових алгоритмів обробки масивів	4	8	14
7	Тема 7. Показники та посилання в мовах програмування C/C++	2	6	10
8	Тема 8. Алгоритми та програмні засоби обробки рядків та символів	4	6	14

9	Тема 9. Типові алгоритми обробки структур	4	6	12
10	Тема 10. Методи обробки текстових та бінарних файлів	4	8	12
	Підсумкова модульна контрольна робота 2			2
	ВСЬОГО	32	60	118

9. Перелік тем інкрементів проєкту (лабораторних робіт)

Проектно-орієнтований інкрементальний підхід до виконання лабораторних робіт ґрунтується на принципах ітеративної та інкрементальної розробки програмного забезпечення, характерних для гнучких підходів до розроблення програмних систем. У межах дисципліни ці принципи адаптовано до освітнього процесу: кожна лабораторна робота є окремим інкрементом, який розширює функціональність попередньої версії одного цілісного програмного проєкту. Після виконання кожного інкременту здійснюються його тестування, налагодження, документування та захист, після чого реалізована функціональність інтегрується в наступну версію проєкту. Такий підхід забезпечує поступове ускладнення програмного продукту та формування у здобувачів практичних навичок послідовного розроблення програмного забезпечення.

Нижче поданий перелік інкрементів проєкту (лабораторних робіт):

1. Типи даних в мовах C та C++. Функції введення - виведення. Обчислення виразів
2. Розгалужені обчислювальні процеси
3. Циклічні обчислювальні процеси
4. Рекурентні співвідношення. Степеневі ряди. Ланцюгові дроби
5. Рекурсивні функції
6. Одновимірні масиви
7. Багатовимірні масиви
8. Показчики та посилання. Показчики на функції. Масиви показчиків в мовах C та C++
9. Обробка рядків в мовах C та C++
10. Структури, масиви структур та об'єднання в мовах C та C++
11. Текстові та бінарні файли

Кількість інкрементів програмного проєкту може відрізнятися від кількості тем навчальної дисципліни, оскільки окремі теми передбачають виконання двох послідовних інкрементів, спрямованих на поетапне розширення функціональності програмного проєкту.

Критерії оцінювання кожного інкременту проєкту деталізовані в методичних вказівках до лабораторних робіт, що реалізуються за проектно-орієнтованим інкрементальним підходом.

Загальний обсяг дисципліни 210 год., у тому числі:

Лекції – 32 год.;
Лабораторні заняття – 60 год.;
Самостійна робота – 118 год.

10. Література та інформаційні ресурси

10.1. Основна література

1. Ковалюк Т. В. Алгоритмізація та програмування. Львів : Магнолія 2006, 2026. 400 с.
2. Бичков О. С. Основи сучасного програмування. Київ : ВПЦ «Київський університет», 2007. 260 с.
3. Halterman R. Fundamentals of C++ Programming. Southern Adventist University, 2018. 766 p.
4. Deitel P., Deitel H. C++ How to Program: An Introduction to C++. Pearson Education Limited, 2017.
5. International Standard for Programming Language C++. ISO/IEC 14882:2020.
6. Programming Languages – C. ISO/IEC 9899:2023.
7. Бєлов Ю. А., Карнаух Т. О., Коваль Ю. В., Ставровський А. Б. Вступ до програмування мовою C++. Організація обчислень : навч. посіб. Київ : Видавничо-поліграфічний центр «Київський університет», 2012. 175 с.

10.2. Додаткова література

8. Татарчук Д. Д., Діденко Ю. В. Програмування мовами C та C++ : навч. посіб. Київ, 2012. 112 с.
9. Трофименко О. Г., Прокоп Ю. В., Логінова Н. І., Задерейко О. В. C++. Алгоритмізація та програмування : підручник. 2-ге вид., перероб. і допов. Одеса : Фенікс, 2019. 477 с.
10. Sommerville I. Software Engineering. 10th ed. Pearson, 2017. 414 p.
11. Aho A. V., Hopcroft J. E., Ullman J. D. The Design and Analysis of Computer Algorithms. Addison-Wesley Publishing Company, 1974. 470 p.
12. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C. Introduction to Algorithms. 3rd ed. MIT Press, 2009. 1312 p.
13. Weiss M. A. Data Structures and Algorithm Analysis in C++. 4th ed. Pearson, 2013.
14. Knuth D. E. The Art of Computer Programming. Vol. 3: Sorting and Searching. 2nd ed. Addison-Wesley Professional, 1998. 800 p.

10.3. Інформаційні ресурси

15. C and C++ in Visual Studio. Microsoft Learn documentation з C/C++. URL: <https://learn.microsoft.com/en-us/cpp/overview/visual-cpp-in-visual-studio?view=msvc-170>
16. Visual Studio documentation. URL: <https://learn.microsoft.com/en-us/visualstudio/windows/?view=visualstudio>
17. C++ reference. URL: https://cppreference.com/Main_Page.