

**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

«ОСНОВИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ»

для здобувачів вищої освіти

галузь знань
спеціальність
освітній рівень
освітня програма
вид дисципліни

F «Інформаційні технології»

F2 «Інженерія програмного забезпечення»

бакалавр

Інженерія програмного забезпечення

обов'язкова

Форма навчання

денна

Навчальний рік

2026/2027

Семестр

1

Кількість кредитів ECTS

3

Мова викладання, навчання

українська

та оцінювання

Форма заключного контролю

іспит

Викладачі: **Вечерковська Анастасія Сергіївна**, кандидат технічних наук, доцент

Пролонговано: на 2027/2028 н.р. _____ (_____) «____» _____ 2027 р.
(підпис) (ПІБ) (дата)

на 2028/2029 н.р. _____ (_____) «____» _____ 2028 р.
(підпис) (ПІБ) (дата)

на 2029/2030 н.р. _____ (_____) «____» _____ 2029 р.
(підпис) (ПІБ) (дата)

Розробник: **Вечерковська Анастасія Сергіївна**, кандидат технічних наук, доцент
кафедри програмних систем і технологій

Завідувач кафедри
програмних систем і технологій


(підпис) проф. Олексій БИЧКОВ

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


(підпис) Ганна КРАСОВСЬКА

1. Мета дисципліни – дати студентам знання з основ наукових концепцій, розкриття сучасних наукових концепцій, понять, інженерних методів та технологій розробки програмного забезпечення.

2. Попередні вимоги до опанування або вибору навчальної дисципліни:

1. Знати теоретичні основи щодо основних етапів створення програмного забезпечення, основи групової комунікацій та конфліктології, а також основні принципи об'єктно-орієнтованого програмування.
2. Вміти використовувати системний підхід до аналізу вимог предметної області.
3. Володіти елементарними навичками пошуку та обробки Інтернет-інформації та навичками роботи в групах.

3. Анотація навчальної дисципліни / референс:

Дисципліна «Основи програмної інженерії» формує у фахівців з Інженерії програмного забезпечення бачення процесів створення та реалізації програмного забезпечення, які спрямована на розробку програмних систем, які працюють надійно та ефективно, вартість розробки та супроводу яких є доступною, і які задовольняють вимогам, висунутим до них замовниками. Для цього вивчаються основні інженерні підходи щодо розробки і супроводження програмного забезпечення шляхом розглядання усіх фаз його життєвого циклу; оволодіння кількісними методами та засобами їх реалізації; ознайомлення з практичними навичками та прийомами, які накопичено у цій галузі.

Набуваються практичні вміння:

- застосовувати методи та відповідні засоби при аналізі, проектуванні, розробці та супроводженні програмного забезпечення.

4. Завдання (навчальні цілі) –

Інтегральна компетентність. Здатність розв'язувати складні спеціалізовані завдання або практичні проблеми інженерії програмного забезпечення або у процесі навчання, що характеризуються комплексністю та невизначеністю умов, із застосуванням теорій та методів інформаційних технологій.

Загальні компетентності:

ЗК4. Здатність спілкування іноземною мовою як усно, так і письмово.

ЗК5. Здатність вчитися та оволодівати сучасними знаннями.

ЗК6. Здатність до пошуку, оброблення та аналізу інформації з різних джерел.

ЗК8. Здатність діяти на основі етичних міркувань.

Спеціальні (фахові, предметні) компетентності

СК3. Здатність розробляти архітектури, модулі та компоненти програмних систем.

СК5. Здатність дотримуватись специфікацій, стандартів, правил і рекомендацій в професійній галузі при реалізації процесів життєвого циклу.

СК10. Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя.

СК14. Здатність до алгоритмічного та логічного мислення.

5. Результати навчання за дисципліною: (описуються з детальною достовірністю для розробки заходів оцінювання)

Результат навчання (1. знати; 2. вміти; 3. комунікація ^{1*} ; 4. автономність та відповідальність ^{2*})		Методи викладання і навчання	Методи оцінювання	Відсоток у підсумковій оцінці з дисципліни
Код	Результат навчання			
1.1	Знати основні поняття програмної інженерії (Тема 1)	Лекція	Стандартизовані тести; усний контроль (усне опитування); письмовий контроль (модульні контрольні роботи); захист лабораторних робіт, контроль рівня виконавських умінь для перевірки практичної підготовки, іспит	4%
1.2	Знати історичний аспект інженерії ПЗ (Тема 2)	Лекція		4%
1.3	Знати поняття життєвого циклу та його складові (Тема 3)	Лекція		4%
1.4	Знати ключові ролі на проекті з розробки ПЗ, та розуміти принципи їх взаємодії (Тема 5-6)	Лекція		4%
1.5	Знати основні принципи та засоби інженерії вимог до програмного забезпечення(Тема 7)	Лекція		4%
1.6	Знати поняття вимог(Тема 7)	Лекція		4%
1.7	Знати принципи аналізу функціональних вимог до ПЗ за допомогою засобів UML Use Case Diagram (Тема 7-9)	Лекція		4%
1.8	Знати загальні принципи проектування (Тема 10-11)	Лекція		4%
1.9	Знати принципи детального проектування (Тема 12-14)	Лекція		4%
1.10	Знати поняття та особливості гнучких методологій розробки програмного забезпечення (Тема 15)	Лекція		4%
1.11	Знати допоміжні процеси при створенні ПЗ (Тема 16-17)	Лекція		4%
1.12	Знати поняття якості ПЗ, та основні процеси, які відбуваються під час тестування ПЗ(Тема 18)	Лекція		4%
1.13	Знати різні види інженерії програмного забезпечення, розуміти їх особливості(Тема 19)	Лекція		4%
2.1	Вміти проводити оцінку розміру та вартості проекту за COCOMO	лабораторні заняття	Стандартизовані тести; усний контроль (усне опитування); письмовий контроль	6%
2.2	Вміти планувати операцій та ресурсів проекту зі створення програмного забезпечення	лабораторні заняття		6%

^{1*} заповнюється за необхідністю, наприклад для практик, лабораторних курсів тощо.

			(модульні контрольні роботи); захист лабораторних робіт, контроль рівня виконавських умінь для перевірки практичної підготовки, іспит	
2.3	Вміти специфікувати вимоги до програмного забезпечення та організація проекту для його створення	лабораторні заняття		6%
2.4	Вміти встановлювати вимоги до функціональності програмного забезпечення	лабораторні заняття		6%
2.5	Вміти проводити специфікування предметної галузі проекту засобами	лабораторні заняття		6%
2.6	Вміти проводити кількісну оцінку якості Діаграм UML	лабораторні заняття		6%
2.7	Вміти моделювати поведінку ПЗ у вигляді діаграма діяльності та стану	лабораторні заняття		6%
2.8	Вміти проводити проектування архітектури ПЗ	лабораторні заняття		6%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання (необов'язково для вибіркових дисциплін)

Результати навчання дисципліни	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2
Програмні результати навчання	1	2	3	4	5	6	7	8	9	10	11	12	13	1	2	3	4	5	6	7	8
ПРН1. Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки.															+	+	+	+	+	+	+
ПРН3 Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення.	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
ПРН4 Знати і застосовувати професійні стандарти і інші нормативно правові документи в галузі інженерії програмного забезпечення.															+	+	+	+	+	+	+
ПРН6 Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення.	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
ПРН8. Вміти розробляти людино-машинний інтерфейс.															+	+	+	+	+	+	+
ПРН9. Знати та вміти використовувати методи та засоби збору, формулювання та аналізу вимог до програмного забезпечення.																+	+	+	+	+	+
ПРН10 Проводити передпроектне обстеження предметної області, системний аналіз об'єкта проектування.															+	+					
ПРН11 Вибирати вихідні дані для проектування, керуючись формальними методами опису вимог та моделювання.																	+	+	+	+	+
ПРН12 Застосовувати на практиці сфективні підходи щодо проектування програмного забезпечення.																		+	+	+	+
ПРН14 Застосовувати на практиці інструментальні програмні засоби доменного аналізу, проектування, тестування, візуалізації, вимірювань та документування програмного забезпечення.															+	+	+	+	+	+	+
ПРН16 Мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації.															+	+	+	+	+	+	+
ПРН17 Вміти застосовувати методи компонентної розробки програмного забезпечення.																		+	+	+	+
ПРН23 Вміти документувати та презентувати результати розробки програмного забезпечення.																+	+	+	+	+	+

7. Схема формування оцінки:

7.1 Форми оцінювання студентів: (зазначається перелік видів робіт та форм їх контролю / оцінювання із зазначенням Min. – рубіжної та Max. кількості балів чи відсотків)

Контроль знань здобувачів вищої освіти здійснюється за модульно-рейтинговою системою. Результати навчальної діяльності здобувачів вищої освіти оцінюються за 100-бальною шкалою. Робота в семестрі поділяється на два змістових модуля. Підсумковий контроль (іспит) – 40 балів.

У змістовий модуль 1 (ЗМ1) входять теми 1 – 6, а у змістовий модуль 2 (ЗМ2) – теми 7 – 12. Обов'язковим для допуску до іспиту є виконання та захист здобувачем вищої освіти усіх лабораторних робіт, що передбачено робочою програмою навчальної дисципліни, здача модульних контрольних робіт. Під час цього сумарна мінімальна рекомендована кількість балів для допуску становить 36.

	ЗМ1/Частина 1	ЗМ2/Частина 2	іспит	Підсумкова оцінка
<i>Мінімум</i>	18	18	24	60
Максимум	30	30	40	100

Оцінка за іспит не може бути меншою **24 балів** для отримання загальної позитивної оцінки за курс. Якщо здобувач на іспит набрав менше 24 балів (60% від 40 балів, відведених на іспит), то вони не додаються до семестрової оцінки незалежно від кількості балів, отриманих під час семестру, а в відомості у графі «результуюча оцінка» переноситься лише кількість балів, отриманих під час семестру.

7.2. Організація оцінювання:

Поточне семестрове оцінювання:

- оцінюванню підлягає модульна контрольна робота, звіти та захист лабораторних робіт;
- упродовж семестру, після завершення лекційного матеріалу за змістовними модулями 1 та 2 (теми 1-6 та 7-12) проводяться письмові модульні контрольні роботи на лабораторних заняттях, на яких здобувачі вищої освіти отримують завдання за варіантами та виконують їх письмово. Максимальна кількість балів за кожну з модульних контрольних робіт не перевищує 10, модульні контрольні роботи не перескладаються. Перевірка завдань, оприлюднення та аналіз результатів здійснюється не пізніше наступного лабораторного заняття;
- кожна лабораторна робота оцінюється в 5 балів, якщо всі завдання виконано, продемонстровано їх працездатність, звіт оформлено відповідно до вимог. Максимальна кількість балів за усі лабораторні роботи не перевищує 40. Кінцевий термін здачі кожної лабораторної роботи є наступне лабораторне заняття після виконання зазначеної лабораторної роботи;
- у разі виконання проєктного STEM завдання, оцінювання здійснюється згідно до Додатка 1, кількість балів не перевищує 40, тоді лабораторна робота оцінюється як зарахована чи ні, якщо всі завдання виконано, продемонстровано їх працездатність, звіт оформлено відповідно до вимог;
- для здобувачів вищої освіти, які набрали сумарно меншу кількість балів ніж критично-розрахунковий мінімум (36 балів), для допуску до іспиту обов'язковими є виконання та захист запланованих лабораторних робіт.

Підсумкове оцінювання (у формі екзамену):

- білет складається із завданнями, що виконуються письмово;

- екзаменаційний білет складається з 2-х теоретичних та 1 практичного питання;
- максимальна кількість балів які можуть бути отримані студентом - 40 балів на екзамені;
- здобувач не допускається до іспиту, якщо під час семестру набрав менше ніж 36 балів;
- ля отримання загальної позитивної оцінки з дисципліни оцінка за екзамен не може бути меншою 24 балів.

7.3. Шкала відповідності:

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

8. Структура навчальної дисципліни

№ п/ п	Назва теми	Кількість годин		
		лекції	Лабора торні	С Р
Змістовий модуль 1. Базові поняття				
1	Тема 1. Основні поняття програмної інженерії. Види інженерій програмного забезпечення: пряма, зворотна та емпірична	2	-	4
2	Тема 2. Життєвий цикл та його складові. Моделі життєвого циклу	2	4	4
3	Тема 3. Планування проектів з розробки ПЗ	2	2	4
4	Тема 4. Рольова модель команди рпоекту з розробки ПЗ.	2	-	2
5	Тема 5. Інженерія вимог до програмного забезпечення	2	-	4
6	Тема 6. Проектування програмного забезпечення засобами UML. Діаграми варіантів використання (UML Use Case Diagram)	2	2	4
7	Модульна контрольна робота 1	2		
8	Тема 7. Використання <i>UML</i> при проектуванні програмної системи.(Діаграми взаємодії, діаграми класів)	2	2	4
9	Тема 8. Діаграми діяльності та кооперації	2	2	4
10	Тема 9. Діаграми станів, компонентів, розгортання.	2	2	4
11	Тема 10. Гнучкі методології розробки програмного забезпечення	2	-	4
12	Тема 11. Технологія та процес розробки ПЗ. Системи контролю версій.	2	2	4
13	Тема 12. Система відслідковування задач JIRA	2	-	4
14	Модульна контрольна робота 2	2	-	
	ВСЬОГО	28	16	4 6

Загальний обсяг 90 год., в тому числі:

лекцій – 28 год.;
практичні заняття – 16 год.;
Самостійна робота – 46 год.

Теми лабораторних занять

1. Планування операцій та ресурсів проєкту зі створення програмного забезпечення та організація проєкту
2. Специфікація та визначення вимог до функціональності програмного забезпечення. Діаграми варіантів використання (Use Case Diagram)
3. Моделювання поведінки програмного забезпечення
4. Етап розробки програмного забезпечення у життєвому циклі ПЗ
5. Тестування та управління змінами програмного забезпечення
6. Технології та процеси розробки програмного забезпечення. Системи контролю версій

Теми самостійних робіт

1. Аналіз проєкту на основі користувацьких вимог.
2. Визначення методології розробки ПЗ та планування.
3. Складання функціональної специфікації програмного продукту.
4. Розробка UML діаграм щодо функціонування системи.
5. Розробка кошторису проєкту.
6. Планування роботи з персоналом відповідно до обраної методології.
7. Визначення критерії та метрик якості ПЗ.
8. Планування ризиків проєкту.
9. Атестація програмних систем.
10. Керування ризиками.
11. Види архітектурного проєктування.
12. Керування вимогами
13. Необхідні якості керівника проєкту
14. Програмування. Аналіз програм
15. Забезпечення якості на етапі планування проєкту
16. Методологія Скрам
17. Методологія Канбан

РЕКОМЕНДОВАНА ЛІТЕРАТУРА:

Основна: (Базова)

до 10 фундаментальних, базових джерел

1. Бабенко Л.П. Основи програмної інженерії : Навчальний посібник для студ. вищ. навч.закладів / Л.П. Бабенко, К.М. Лавріщева; Ін-тут програмних систем НАН України; КНУТШ. - Київ : Знання, 2001. - 269 с. - (Вища освіта 21 століття). - ІБВИ 966-620-095-3 (бібліотека Університету)
2. Лавріщева К.М. Програмна інженерія : підручник / К.М. Лавріщева ; НАН України, Ін-т програм, систем. - Київ : Академперіодика, 2008. - 319, [1] с. : іл., табл. - Термінол. слов.: с. 312-317. - Бібліогр. наприкінці розд. - ІБВИ 978-966-02-5052-9 (бібліотека Університету)
3. СИДОРОВ М.О. Вступ до інженерії програмного забезпечення. - К.: НАУ, 2008. - 65с.
4. M.Fowler. UML Distilled: A Brief Guide to the Standard Object Modeling Language
5. – Longman (Pearson Education), 2003. - 192с.
6. Кучеров, Д. П. Інженерія програмного забезпечення : навч. посіб. / Д. П. Кучеров, Є. Б. Артамонов. – Київ : НАУ, 2017. – 386 с. : іл.
7. Левус, Є. В. Вступ до інженерії програмного забезпечення : навч. посіб. / Є. В. Левус, Н. Б. Мельник. – Львів : Видавництво Львівської політехніки, 2018. – 246 с. : іл

8. Бандура В.В., Храбатин Р.І. Архітектура та проектування програмного забезпечення: конспект лекцій. – Івано-Франківськ: ІФНТУНГ, 2012. — 240 с.

Додаткова:

1. Jacobson I. Object-Oriented Software Engineering. A use Case Driven Approach, Revised Printing. – New York: Addison-Wesley Publ. Co., 1994. – 529 p.
2. Sommerville I. Software Engineering, Global Edition. - Pearson Education Limited, 2015. - 860 с.

ОЦІНЮВАННЯ СЕМЕСТРОВОЇ ПРОЄКТНОЇ РОБОТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Формалізовані критерії семестрова оцінка 60 = проєкт (40) + лабораторні / ДЗ (20) ECTS

1. Призначення і принципи

Оцінювання побудовано так, щоб кожен бал був підкріплений об'єктивним, перевірюваним фактом, а не враженням. Це знімає питання «а чому така оцінка?»: поруч із кожним критерієм наведено правило, яке однозначно зіставляє спостереження з рівнем виконання. Нижче – дві незалежні рубрики: для дисциплін із програмування та баз даних і для математичних дисциплін.

1. Вагові коефіцієнти, а не середнє. Кожен критерій має власну вагу в балах; бал критерію = вага × рівень виконання (рівень у відсотках, 0–100). Підсумок – сума цих балів, тож вклад кожного критерію прозорий і не «розмивається» усередненням.

2. Об'єктивність. Рівень виконання визначається вимірюваними ознаками: частка пройдених тест-кейсів, покриття тестами, кількість зауважень linter (програма, яка перевіряє код: **cpp lint** – перевірка відповідності стилю (від Google), **Roslyn Analyzers** / **.NET analyzers** – вбудовані прямо в компілятор C#, працюють у Visual Studio «з коробки»), частка вчасних задач, чек-листи «виконано / не виконано». Мінімум суб'єктивних формулювань.

3. Складність індивідуальної зони. Замість суб'єктивної «доданої вартості» на Зрізі 0 фіксується зона відповідальності здобувача вищої освіти та її рівень складності. Блок D оцінює складність × повноту виконання саме цієї зони: у команді проєкт може бути «експертним» для всіх, але особиста робота різна. Добре виконаний базовий рівень НЕ карається (80 %), а вищий рівень дає повний бал – це стимулює брати складніше.

4. Добросовісність як зовнішній множник. Порушення діє як множник M до підсумку ($M \times \text{MIN}(100; \Sigma + \text{Бонус})$), а не як окремих критерій, який можна «перекрити» іншими балами; за $M = 0$ підсумок дорівнює 0.

2. Формула розрахунку проєкту

Бал критерію = Вага × Рівень, % ÷ 100.

Рівень виконання вводиться у відсотках 0–100 (у правилах нижче 1.0 = 100 %, 0.8 = 80 % тощо).

Σ = сума балів усіх критеріїв проєкту (сума ваг = 100).

Підсумок проєкту (0–100) = $M \times \text{MIN}(100; \Sigma + \text{Бонус})$,

де M – множник академічної добросовісності (1.0 / 0.5 / 0.0), Бонус – over-score 0–5 балів (див. розділ 7). Множник застосовується ЗОВНІ, тому за $M = 0$ (груба фальсифікація) підсумок = 0 навіть із бонусом.

Цей підсумок проєкту (0–100) входить у семестрову оцінку як 40 балів ($\times 0.4$); решту 20 балів дають лабораторні / домашні роботи (розділ 10). Разом – 60 балів поточної семестрової оцінки (розділ 11). Груповий журнал-калькулятор рахує все це автоматично й одразу переводить у ECTS.

Переведення у ECTS та національну оцінку (за 100 %)

Бали (0–100)	ECTS	Національна оцінка
90–100	A	5 (відмінно)
82–89	B	4 (добре)
74–81	C	4 (добре)
64–73	D	3 (задовільно)
60–63	E	3 (задовільно)

35–59	FX	2 (незадовільно, з правом перескладання)
-------	----	--

3. Командне проєктне оцінювання: що спільне, а що індивідуальне

Ключовий принцип:

Оцінка за проєкт – це оцінка ЗДОБУВАЧА ВИЩОЇ ОСВІТИ за роботу в командному проєкті, а не оцінка команди.

Тому бали розділені на дві складові:

- Командна складова – 30 балів проєкту (блок С, технічна/математична якість продукту; у семестрі → 12 балів). Команда створила один продукт, тож його якість – спільна відповідальність і оцінюється однаково для всіх членів команди.
- Індивідуальна складова – 70 балів проєкту (блоки А + В + D + E: процес, виконання власної зони відповідальності, складність цієї зони та захист; у семестрі → 28 балів). Ці бали свої в кожного здобувача вищої освіти.

Формула для кожного здобувача вищої освіти:

Оцінка_і = Продукт_команди (спільне) + Індивідуальне_і.

Це прибирає проблему «пасажира»: один написав половину системи, інший зробив два коміти – і вони не можуть отримати однаковий бал. Водночас 30 спільних балів створюють правильний стимул: мені недостатньо зробити лише свою частину – я зацікавлений, щоб команда довела продукт до робочого стану. Свідомо НЕ застосовується множник «командний бал × коефіцієнт внеску»: такі коефіцієнти швидко дають викривлення (коаліції, покарання team lead).

Для оцінювання цілої команди разом у калькуляторі є окремі аркуші «Команда ...»: продукт С вводиться один раз, блоки А/В/D/E – на кожного здобувача вищої освіти, підсумок рахується кожному окремо.

4. Рубрика А · Програмування / Баз даних

Блок С – командна складова (однакова для всіх у команді). Блоки А, В, D, E – індивідуальні. Джерела даних: система контролю версій (Git: issue → branch → PR → коміти → review → merge), трекер задач, звіти інструментів (coverage, лінтер), фіксований набір приймальних тест-кейсів, захист.

Код	Критерій	Вага	Об'єктивне правило нарахування рівня (у %, 0–100)
А. Процес і дисципліна (індивід.)			
A1	Дотримання проміжних дедлайнів чекпоінти) (3	6	Рівень за дискретною шкалою (не частка): усі 3 чекпоінти вчасно з повним переліком артефактів → 1.0; 1 етап із затримкою ≤24 год → 0.8; 2 з 3 вчасно → 0.6; 1 з 3 → 0.4; жодного або постфактум → 0.
A2	Ведення репозиторію логбука /	5	+0.2 за кожен виконаний пункт (перевірка Git-історії): (1) ≥1 змістовний коміт/тиждень; (2) коміти прив'язані до issue; (3) README з інструкцією запуску; (4) лог рішень ≥5 записів «проблема→рішення»; (5) гілкова модель (не все в main).
A3	Опрацювання зворотного зв'язку	4	Рівень = частка зауважень із чекпоінтів, оформлених як issue і закритих до наступного етапу. 1.0 ≥90%; 0.8 = 70–89%; 0.6 = 50–69%; 0.4 = 30–49%; 0 = <30%.
В. Внесок і надійність (індивід.)			

B1	Виконання індивідуальної зони відповідальності	12	Зона фіксується на Зрізі 0. Рівень оцінюється за сукупністю артефактів (ланцюг issue → branch → PR → коміти → review → merge; код, архітектурні рішення, тести, документація, debugging); story points – лише один із доказів, не самоціль. Повний обсяг якісно → 1.0; переважно (незначні прогалини) → 0.8; частково → 0.6; фрагментарно → 0.4; мінімально → 0.2; не виконано → 0. Авторство визначається за виконаною роботою, а не git blame.
B2	Виконання зобов'язань і комунікація ризиків	8	Рівень = частка взятих задач, які АБО виконані в узгоджений строк, АБО за якими ризик/затримку завчасно (до дедлайну) комуніковано з оновленням issue та новою датою. $1.0 \geq 95\%$; $0.8 = 80-94\%$; $0.6 = 60-79\%$; $0.4 = 40-59\%$; $0.2 = 20-39\%$; $0 = <20\%$. Мовчазний зрив дедлайну як виконане не зараховується.
B3	Результативність code-review	5	Зараховуються лише результативні рев'ю (провів рев'ю чужого PR → знайшов суттєву проблему або запропонував обґрунтоване покращення → автор відреагував → перевірено виправлення). ≥ 3 результативних → 1.0; 2 → 0.8; 1 → 0.5; 0 → 0. Кількість дрібних косметичних коментарів не враховується.

С. Технічна якість продукту (КОМАНДНЕ – однакове всім)

C1	Функціональна коректність (приймальні кейси)	12	Рівень = пройдені приймальні тест-кейси / усього узгоджених кейсів (фіксований список кейсів). Прямий розрахунок частки.
C2	Обробка крайніх випадків і помилок	5	+0.2 за кожен: (1) валідація введення; (2) обробка порожніх/некоректних даних; (3) обробка винятків без аварійного завершення; (4) межові значення; (5) відсутність необроблених винятків у лог-файлі прогону.
C3	Автотести і покриття	6	$1.0 = \text{автотести} + \text{покриття} \geq 80\%$ (звіт інструмента) + CI ганяє тести; $0.8 = \text{автотести} + \text{покриття} \geq 60\%$; $0.6 = \text{автотести} + \text{покриття} \geq 40\%$; $0.4 = \text{автотести} \text{ є, покриття} < 40\%$; $0 = \text{автотестів немає}$.
C4	Якість коду за статичним аналізом	4	Critical/major зауважень лінера-аналізатора на 1000

			рядків: $0 \rightarrow 1.0$; $\leq 2 \rightarrow 0.8$; $\leq 5 \rightarrow 0.6$; $\leq 10 \rightarrow 0.4$; $> 10 \rightarrow 0$. Інструментальна перевірка (pylint/ESLint/SonarQube).
C5	Цілісність і безпека роботи з даними / схеми БД	3	+0.2 за кожен застосовний пункт: (1) нормалізація ≥ 3 НФ або обґрунтована денормалізація; (2) РК/ФК та обмеження цілісності; (3) індекси на полях пошуку/JOIN; (4) параметризовані запити (захист від ін'єкцій); (5) транзакції де потрібно. Для проєктів без БД – еквівалент для файлових даних (валідація, транзакційність, узгодженість).
D. Складність зони відповідальності (індивід.)			
D1	Складність і повнота індивідуальної зони відповідальності	10	Зона фіксується на Зрізі 0 (для індивідуальних проєктів – весь проєкт). Рівень = складність зони $\times$ повнота якісного виконання: експертна зона повністю виконана (event-driven backend, складна авторизація, розподілені воркери тощо) $\rightarrow 1.0$; просунута $\rightarrow 0.9$; базова, повністю та якісно виконана $\rightarrow 0.8$; зона виконана частково/спрощено $\rightarrow 0.4\text{--}0.5$; тривіальний внесок (напр. README + шаблонний CRUD) $\rightarrow 0.2$; не виконано $\rightarrow 0$. Добре виконаний базовий рівень НЕ карається.
E. Захист і розуміння (індивід.)			
E1	Відповіді за відкритим банком питань	12	Рівень = сума за відповіді / максимум. Здобувач вищої освіти отримує К питань з оприлюдненого банку; кожна відповідь: повна = 1, часткова = 0.5, немає = 0.
E2	Модифікація умови «на льоту»	5	2 заздалегідь заготовлені мікрозміни умови; кожна коректно пояснена/реалізована = 0.5. Рівень = 0 / 0.5 / 1.0.
E3	Демонстрація запуску за README	3	Запускається з нуля за README без втручання автора у визначений ліміт часу $\rightarrow 1.0$; із дрібним втручанням $\rightarrow 0.5$; не запускається $\rightarrow 0$.
РАЗОМ (сума ваг)		100	Підсумок = $\text{MIN}(100; M \times \Sigma(\text{вага} \times \text{рівень}) + \text{Бонус})$

5. Рубрика Б · Математичні дисципліни

Блок С – командна складова. Блоки А, В, D, E – індивідуальні. Джерела даних: робочий журнал і проміжні викладки, еталонний набір задач/кейсів для звірки результатів, відтворений скрипт розрахунку, захист.

Код	Критерій	Вага	Об'єктивне правило нарахування рівня (у %, 0–100)
А. Процес і дисципліна (індивід.)			
A1	Дотримання проміжних дедлайнів (3 чекпоінти)	6	Рівень за дискретною шкалою (не частка): усі 3 чекпоінти вчасно з повним переліком артефактів → 1.0; 1 етап із затримкою ≤24 год → 0.8; 2 з 3 вчасно → 0.6; 1 з 3 → 0.4; жодного або постфактум → 0.
A2	Ведення робочого журналу / оформлення проміжних результатів	5	+0.2 за кожен пункт: (1) щотижневі записи ходу роботи; (2) кожен результат датований і з посиланням на метод/джерело; (3) проміжні викладки збережені й відтворювані; (4) лог помилок/виправлень ≥5 записів; (5) фінальні матеріали структуровані (LaTeX/ноутбук).
A3	Опрацювання зворотного зв'язку	4	Рівень = частка зауважень із чекпоінтів, урахованих і закритих до наступного етапу. 1.0 ≥90%; 0.8 = 70–89%; 0.6 = 50–69%; 0.4 = 30–49%; 0 = <30%.
В. Внесок і надійність (індивід.)			
B1	Виконання індивідуальної зони відповідальності (виведення / обчислення / розділи)	12	Зона фіксується на Зрізі 0. Рівень оцінюється за сукупністю: виконані виведення, обчислювальні експерименти, розділи, перевірки; підтверджено робочим журналом/репозиторієм. Повний обсяг якісно → 1.0; переважно (незначні прогалини) → 0.8; частково → 0.6; фрагментарно → 0.4; мінімально → 0.2; не виконано → 0.

B2	Виконання зобов'язань і комунікація ризиків	8	Рівень = частка взятих підзадач, які АБО виконані в узгоджений строк, АБО за якими ризик/затримку завчасно (до дедлайну) комуніковано з оновленням запису та новою датою. 1.0 $\geq$ 95%; 0.8 = 80–94%; 0.6 = 60–79%; 0.4 = 40–59%; 0.2 = 20–39%; 0 = < 20%. Мовчазний зрив дедлайну як виконане не зараховується.
B3	Результативність взаємної перевірки розв'язків	5	Зараховуються лише результативні рецензії (перевірив розв'язок колеги $\rightarrow$ знайшов помилку або запропонував суттєве покращення $\rightarrow$ автор відреагував $\rightarrow$ перевірено виправлення). ≥ 3 результативних $\rightarrow$ 1.0; 2 $\rightarrow$ 0.8; 1 $\rightarrow$ 0.5; 0 $\rightarrow$ 0.
С. Математична якість (КОМАНДНЕ – однакове всім)			
C1	Коректність результатів (еталонний набір)	12	Рівень = задачі/кейси з результатом, що збігається з еталоном/аналітичним розв'язком у межах заданої похибки / усього кейсів. Прямий розрахунок частки.
C2	Строгість обґрунтування	6	+0.2 за кожен: 1) усі кроки виведення наведені; 2) названі теореми/методи з умовами; 3) перевірено умови застосовності методу; 4) наведено оцінку похибки/збіжності; 5) перевірка розмірностей/граничних випадків.
C3	Адекватність обраних математичних методів і моделей	5	+0.25 за кожен: 1) припущення явно виписані; 2) верифікація на відомому окремому випадку; 3) аналіз чутливості до параметрів;

			4) порівняння з альтернативним методом/джерелом.
C4	Чисельна коректність і стійкість	4	+0.25 за кожен: 1) обробка крайніх (ділення на нуль, погана обумовленість); 2) контроль похибки округлення; 3) чисельно підтверджена збіжність; 4) відтворюваність (фіксовані дані/seed).
C5	Нотація й відтворюваність оформлення	3	+0.333 за кожен: 1) коректна й послідовна математична нотація; 2) усі позначення визначені; 3) наданий відтворюваний скрипт/викладки розрахунку.
D. Складність зони відповідальності (індивід.)			
D1	Складність і повнота індивідуальної зони відповідальності	10	Зона фіксується на Зрізі 0 (для індивідуальних проєктів – весь проєкт). Рівень = складність зони × повнота якісного виконання: 1) експертна зона повністю виконана (нестандартне доведення, дослідницька/оптимізаційна задача) → 1.0; 2) просунута → 0.9; 3) базова, повністю та якісно виконана → 0.8; 4) зона виконана частково/спрощено → 0.4–0.5; 5) тривіальний внесок → 0.2; 6) не виконано → 0. 7) Добре виконаний базовий рівень НЕ карається.
E. Захист і розуміння (індивід.)			
E1	Відповіді за відкритим банком теоретичних питань	12	Рівень = сума за відповіді / максимум. К питань з оприлюдненого банку; кожна: повна = 1, часткова = 0.5, немає = 0.
E2	Розв'язання видозміненої задачі «на льоту»	5	2 заготовлені мікрозміни параметрів/умов;

			кожна коректно розв'язана/пояснена = 0.5. Рівень = 0 / 0.5 / 1.0.
ЕЗ	Демонстрація відтворюваності розрахунку	3	Перерахунок: з нуля відтворює результат → 1.0; частково → 0.5; ні → 0.
РАЗОМ (сума ваг)		100	Підсумок = MIN(100; M × Σ(вага×рівень) + Бонус)

6. Академічна доброчесність і політика щодо ШІ

Доброчесність застосовується як зовнішній множник М до підсумкового бала – Оцінка = $M \times \text{MIN}(100; \Sigma + \text{Бонус})$:

М = 1.0 – порушень не виявлено.

М = 0.5 – встановлене порушення: плагіат, запозичення коду/викладок без атрибуції, недеклароване використання згенерованого коду. Застосовується за рішенням згідно з політикою кафедри.

М = 0.0 – груба фальсифікація результатів або видавання чужої роботи за власну; підсумок = 0 (навіть із бонусом), робота передається на розгляд комісії.

Політика щодо генеративного ШІ та пов'язаних інструментів і технологій

Використання інструментів метапрограмування в широкому сенсі, зокрема ШІ-інструментів генерації коду та інших інструментів, пов'язаних з генерацією і автоматичною обробкою текстів формальними і неформальними мовами в контексті процесу розробки ПЗ у процесі виконання проєктної роботи є дозволеним (і, залежно від завдання роботи, може бути рекомендованим) за таких обов'язкових умов:

1) у файлі META-USAGE здобувач вищої освіти декларує застосовані принципи (мета)програмування, їх переваги і обмеження у контексті проєктної роботи, а також конкретні застосовані інструменти, ключові запити до них, згенеровані фрагменти коду / інші артефакти;

2) на захисті здобувач вищої освіти демонструє володіння матеріалом файлу META-USAGE, зокрема, здатний відповісти на додаткові питання щодо обраної стратегії застосування згаданих інструментів та її порівняння з альтернативними стратегіями, що впливають з раніше пройденого матеріалу даної дисципліни та з попередніх дисциплін;

3) на захисті здобувач вищої освіти демонструє розуміння змісту згенерованих фрагментів коду / артефактів, способів перевірки їх якостей (наприклад, коректності, ефективності і т.д.), можливих шляхів їх подальшого використання.

Недекларований згенерований результат кваліфікується як порушення доброчесності (див. вище). Нездатність пояснити власний результат безпосередньо знижує бали блоку Е (захист) за відповідними правилами.

7. Механіка збору даних

Щоб критерії спиралися на факти, а не на пам'ять наприкінці семестру, впроваджуються чотири інструменти:

1. Зріз 0 (старт).

Здобувач вищої освіти фіксує план і – головне для команд – свою індивідуальну зону відповідальності та її рівень складності (базовий / просунутий / експертний).

Наприклад:

Петренко – API + авторизація;

Іваненко – рівень персистентності + міграції;

Сидоренко – асинхронні воркери + Redis.

Це вхід для блоків В і D.

2. Три чекпоінти (напр. 2-й, 5-й, 8-й тижні).

Здаються «сирі» напрацювання; оцінюються блоки А і частково В (процес). Кожне зауваження оформлюється як issue/запис і має бути закрито до наступного етапу – це прямо вимірює критерій А3.

3. Відкритий банк питань.

Заздалегідь оприлюднений перелік питань за категоріями. Захист у командному проєкті проводиться персонально для кожного (за свою частину + за проєкт загалом); кожна відповідь оцінюється за чек-листом повна/часткова/немає. Саме захист найнадійніше виявляє реальний внесок.

4. Анонімна взаємооцінка (peer-review) – датчик аномалії.

Індекс не змінює бал автоматично; за істотної розбіжності з об'єктивними артефактами (Git/issue/PR) викладач проводить додаткову індивідуальну перевірку. Прямий коефіцієнт «бал × думка команди» не застосовується. Готова форма – на аркуші «Peer-review» калькулятора.

8. Приклад: одна команда – різні семестрові оцінки

Команда зробила хороший продукт, тож командна складова однакова для всіх: $C = 27/30$. Спочатку рахуємо підсумок проєкту (0–100), потім семестрову оцінку = проєкт × 0.4 + лабораторні (0–20):

Здобувач вищої освіти А (активно працював):

$A = 14, B = 23, D = 9, E = 18 \rightarrow \text{проєкт } 27 + 64 = 91/100 \rightarrow 36.4/40$. Лабораторні 18/20.

Семестр 54.4/60 (90.7 %) $\rightarrow$ ECTS A / 5.

Здобувач вищої освіти Б (працював менше):

$A = 12, B = 14, D = 7, E = 16 \rightarrow \text{проєкт } 27 + 49 = 76/100 \rightarrow 30.4/40$. Лабораторні 15/20.

Семестр 45.4/60 (75.7 %) $\rightarrow$ ECTS C / 4.

Здобувач вищої освіти В (майже не працював):

$A = 5, B = 4, D = 3, E = 8 \rightarrow \text{проєкт } 27 + 20 = 47/100 \rightarrow 18.8/40$.

Це НИЖЧЕ порога 20/40, тож проєкт не складено.

Попри лабораторні 12/20, семестрова не виставляється: здобувач вищої освіти іде на перескладання проєкту.

Один продукт, одна команда – але А, С і незарахування. Проєкт не рятують ні спільний продукт, ні лабораторні, якщо особиста участь у проєкті нижча за поріг. Увесь цей розрахунок робиться в груповому журналі автоматично.

9. Лабораторні та домашні роботи (20 балів)

Лабораторні й домашні роботи – обов'язковий ІНДИВІДУАЛЬНИЙ компонент семестрової оцінки (20 балів). Їхнє призначення – перевірити ШИРИНУ компетентностей кожного здобувача вищої освіти. У командному проєкті людина неминуче спеціалізується (один робить API, інший – схему БД, третій – тести), а лабораторні гарантують, що кожен особисто виконав базові результати навчання: JOIN, транзакції, індекси, нормалізацію, розв'язання типових задач тощо.

Як оцінюється кожна робота

Робота оцінюється не за принципом «здав / не здав», а сукупно:

75 % – правильність виконання

25 % – короткий індивідуальний захист (буквально 2–3 хвилини усних запитань).

Захист успадковує політику щодо ШІ: якщо здобувач вищої освіти не може пояснити власний результат, частина «правильність» не зараховується. Захист можна проводити вибірково (частина здобувачів вищої освіти за пару).

Підрахунок підсумку L (0–20)

Кожна робота має «сиру» вагу за складністю (простий ≈ 5 , середній ≈ 8 , складний ≈ 12), а її виконання оцінюється у відсотках 0–100. Підсумок нормалізується, тому кількість робіт (до 12) і їхня складність можуть бути будь-якими:

$$L = 20 \times \Sigma(\text{вага} \times \text{оцінка}) / \Sigma(\text{вага} \times 100).$$

Роботи, які здобувач вищої освіти ще не складав, лишаються порожніми й у підрахунок не входять – тож журнал коректний і посеред семестру. Усі ці оцінки викладач веде в журналі групи (аркуш «Завдання_ДЗ»), а підсумок L рахується автоматично для кожного здобувача вищої освіти.

10. Семестрова оцінка: 60 = 40 (проєкт) + 20 (лабораторні / ДЗ)

У семестрову оцінку проєкт входить масштабованим ($\times 0.4 = 40$ балів), а решту 20 балів дають індивідуальні лабораторні / домашні роботи. Це зберігає всю логіку побудованих критеріїв – треба лише помножити підсумок проєкту на 0.4.

Модель, яку легко пояснити здобувачеві вищої освіти:

20 - «Я вмію» (індивідуальні лабораторні / ДЗ)

+ 28 - «Я зробив» (особистий внесок у проєкт: $A + B + D + E, \times 0.4$)

+ 12 - «Ми зробили» (спільний продукт С, $\times 0.4$)

= 60 балів поточної семестрової оцінки

Отже лише 20 % загальної оцінки залежить безпосередньо від спільного результату команди (12 із 60), а 80 % підтверджуються власною діяльністю здобувача вищої освіти.

Масштабування проєкту ($\times 0.4$)

1. Процес і дисципліна: 15 $\rightarrow$ 6.

2. Особистий внесок і надійність: 25 $\rightarrow$ 10.

3. Якість спільного продукту: 30 $\rightarrow$ 12.

4. Складність індивідуальної зони: 10 $\rightarrow$ 4.

5. Індивідуальний захист: 20 $\rightarrow$ 8.

Разом 40. Ваги й правила всередині проєкту не змінюються.

Поріг проєкту (щоб лабораторні не компенсували неучасть)

Проектна складова вважається складеною лише за умови ≥ 20 із 40 балів. Інакше семестрова оцінка не виставляється до перескладання проєкту. Це прибирає абсурдну ситуацію, коли здобувач вищої освіти має високі бали за лабораторні, але фактично не працював над центральною для дисципліни проєктною діяльністю. За $M = 0$ (груба фальсифікація) проєкт = 0, тож поріг не проходить автоматично.

Чому 40 + 20, а не 45 + 15 чи 30 + 30

40 + 20 водночас робить проєкт головним (дві третини балів) і лишає достатню вагу, щоб перевірити, що кожен здобувач вищої освіти особисто опанував базові результати навчання, а не лише свою роль у команді. 45 + 15 надто звужує цю індивідуальну «страховку ширини»; 30 + 30 розмиває акцент на проєкті, який є основною освітньою технологією дисципліни.

Місце у 100-бальному курсі

Ці 60 – поточний контроль; решта 40 – екзамен або залік. Для дисципліни із заліком без екзамену 60 балів нормалізуються до 100 ($\times 100/60$). Усе це реалізовано в груповому журналі-калькуляторі: на аркушах «Проект» і «Завдання_ДЗ» викладач ставить рівні у %, а аркуш «Відомість» сам збирає для кожного здобувача вищої освіти проєкт (40) + лабораторні (20) = семестрову оцінку, перевіряє поріг і видає ECTS.

11. Примітки щодо застосування

Індивідуальні (не командні) проєкти: блок С стає теж індивідуальним (весь продукт – робота одного здобувача вищої освіти), решта критеріїв застосовується без змін; для соло-проєкту не використовуються В3 (code-review/взаємоперевірка) і peer-review – рекомендовано перенести 5 балів В3 у С1. Сума ваг залишається 100.

Готовий груповий журнал-калькулятор (окремі файли для програмування/БД і для математичних дисциплін) веде цілу групу.

Аркуш:

«Група» (список і команди),

«Спільна_якість / Продукт_команд» (продукт С на команду),

«Проект» (індивідуальні А/В/Д/Е кожному),

«Завдання_ДЗ» (лабораторні/ДЗ, до 12 робіт)

«Відомість», яка сама збирає семестрову оцінку 60 з ECTS.

Заповнюються лише жовті клітинки; рівні вводяться у відсотках (0–100). Правила виставлення% спливають підказкою при кліку на клітинку і продубльовані на аркуші «Критерії».