

**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

«ПРОГРАМУВАННЯ МУЛЬТИАГЕНТНИХ СИСТЕМ»

для здобувачів вищої освіти

галузь знань
спеціальність
освітній рівень
освітня програма
вид дисципліни

F Інформаційні технології
F2 Інженерія програмного забезпечення
магістр
Інженерія програмного забезпечення
вибіркова

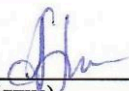
Форма навчання	денна
Навчальний рік	2026/2027
Семестр	4
Кількість кредитів ECTS	3
Мова викладання, навчання та оцінювання	українська
Форма заключного контролю	іспит

Викладачі: **Бичков Олексій Сергійович**, *доктор технічних наук, професор*

Пролонговано: на 2027/2028 н.р. _____ (підпис) _____ (ПІБ) «____» _____ 2027 р.
(дата)
на 2028/2029 н.р. _____ (підпис) _____ (ПІБ) «____» _____ 2028 р.
(дата)
на 2029/2030 н.р. _____ (підпис) _____ (ПІБ) «____» _____ 2029 р.
(дата)

Розробник: **Бичков Олексій Сергійович**, доктор технічних наук, професор,
професор кафедри програмних систем і технологій

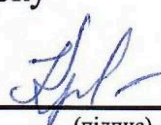
Завідувач кафедри
програмних систем і технологій

 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)

© Олексій БИЧКОВ, 2026 рік
© _____, 20__ рік
© _____, 20__ рік

1. Мета дисципліни – формування у студентів теоретичних знань та практичних навичок програмування мультиагентних систем (МАС) з використанням мови програмування Rust. Курс охоплює основи мови Rust, включаючи систему ownership, borrowing та lifetimes, а також концепції паралельного та асинхронного програмування для створення автономних агентів та їх координації у рої. Особлива увага приділяється безпечному програмуванню без data races, що забезпечується унікальними властивостями компілятора Rust.

2. Попередні вимоги до опанування або вибору навчальної дисципліни:

Дисципліна базується на знаннях, вміннях і практичних навичках, отриманих студентами при вивченні дисциплін «Основи програмування», «Об'єктно-орієнтоване програмування», «Операційні системи».

3. Анотація навчальної дисципліни

Навчальна дисципліна «Програмування мультиагентних систем» присвячена отриманню студентом компетенцій для проектування та реалізації автономних агентів і мультиагентних систем. Курс використовує мову програмування Rust як основний інструмент, що забезпечує безпеку пам'яті та потокобезпечність на етапі компіляції. Студенти вивчають концепції автономних агентів, патерни комунікації між агентами, багатопотокове та асинхронне програмування, Actor Model та практичні аспекти координації рою агентів.

4. Завдання (навчальні цілі)

- формування у здобувачів вищої освіти розуміння концепцій мультиагентних систем та автономних агентів;
- оволодіння основами мови програмування Rust: синтаксис, система типів, ownership, borrowing;
- вивчення механізмів багатопотокового програмування: threads, Arc, Mutex, channels;
- ознайомлення з асинхронним програмуванням: async/await, Tokio runtime;
- практичне застосування патернів Actor Model для побудови МАС;
- розробка наскрізного проєкту симуляції автономного рою БПЛА.

Фахові компетентності спеціальності (СК)

СК12 (для вибіркових блоків освітньо-наукової програми). Здатність застосовувати методи і алгоритми інтелектуального аналізу даних.

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	Відсоток у підсумковій оцінці з дисципліни
Код	Результат навчання			
1.1	Знати синтаксис та семантику мови Rust, систему типів, ownership та borrowing	Лекція, лабораторне заняття	Захист лабораторних робіт, МКР	20%

1.2	Знати концепції мультиагентних систем, архітектуру автономних агентів	Лекція, лабораторне заняття	Захист лабораторних робіт, МКР	20%
2	Вміти розробляти безпечний код на Rust з використанням ownership та borrowing	Лабораторне заняття	Захист лабораторних робіт	40%
3	Здатність працювати в команді над проектом MAC	Лабораторне заняття	Захист проекту	10%
4	Демонструвати відповідальність за якість та безпеку коду	Лабораторне заняття	Захист лабораторних робіт	10%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Результати навчання дисципліни	1.1	1.2	1.3	2	3	4
Програмні результати навчання						
ПРН21.1. (Для вибіркового блоку 1 <i>Програмування інтелектуальних систем</i>). Вміти обирати і застосовувати методи та алгоритми інтелектуального аналізу даних та машинного навчання у задачах класифікації, прогнозування, кластерного аналізу з використанням програмних інструментів підтримки багатовимірного аналізу даних на основі технологій DataMining, TextMining, WebMining.	+	+	+	+	+	+

7. Схема формування оцінки

7.1 Форми оцінювання здобувачів освіти:

Контроль знань студентів здійснюється за модульно-рейтинговою системою. Результати навчальної діяльності студентів оцінюються за 100-бальною шкалою.

Підсумковий контроль (іспит) проводиться у формі іспиту. Максимально можлива кількість балів, що може бути отримана протягом семестру 60. Максимальна можлива кількість балів за іспит – 40.

Мінімальний пороговий рівень оцінки, що може бути отримана протягом семестру, складає 50% від максимально можливої кількості балів, тобто - 30.

Якщо студент не здав лабораторні роботи, то він не допускається до іспиту.

Для студентів, які набрали сумарно меншу кількість балів ніж пороговий мінімум, для одержання допуску до іспиту необхідно перездати модульну контрольну роботу, виконати та захистити лабораторні роботи.

Здобувачі вищої освіти, які упродовж семестру не досягли мінімального порогового рівня проходять оцінювання результатів навчання, яку не були оцінені протягом семестру.

У випадку відсутності студента з поважних причин відпрацювання та перездача МКР здійснюються у відповідності до «Положення про організацію освітнього процесу» від 11 квітня 2022 року.

Семестрове оцінювання:

Поточне оцінювання проводиться для:

- лабораторних робіт 5 шт. – до 20 балів;

- проєктної частини STEM завдання – до 40 балів. Оцінювання проводиться відповідно до «Положення про оцінювання освітніх STEM-проєктів», затвердженого на засіданні кафедри програмних систем і технологій (протокол № 16 від «01» травня 2026 р.): <https://pst.knu.ua/academics/bachelor/evaluation-stem-projects/>.

Підсумкове оцінювання (у формі іспиту):

Екзаменаційний білет складається із тестових завдань і практичних задач. Максимальна можлива кількість балів за іспит – 40.

При результаті підсумкового оцінювання менше 24 балів виставляється оцінка незадовільно.

Оцінювання за формами контролю:

Вид контролю	Оцінювання
Проектна частина STEM завдання	40
Лабораторні роботи	20

7.2. Організація оцінювання:

Упродовж семестру, після завершення лекційного матеріалу проводиться захист лабораторних робіт та оцінювання модульної контрольної роботи.

Екзаменаційна робота проводиться письмово із співбесідою (за необхідності). Співбесіда проводиться та темами лекцій, самостійної роботи та теоретичною частиною STEM завдання.

На модульний контроль виносяться теми навчальних занять (аудиторної роботи).

На іспит виносяться додатково теми самостійної роботи.

Рівень досягнення всіх запланованих результатів навчання визначається за результатами:

- захистів виконаних лабораторних робіт,
- захисту теоретичної частини STEM завдання,
- оцінювання екзаменаційної роботи.

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

№ п/п	Назва лекції	Кількість годин		
		Лекції	Лабораторні	СР
Змістовий модуль. Основи Rust та мультиагентних систем				
1.	Вступ. Концепція агента та MAC. Налаштування середовища Rust	2	2	6
2.	Змінні, типи даних, функції в Rust. Моделювання стану агента	2	2	8
3.	Ownership та Borrowing: унікальна система керування пам'яттю Rust	2	2	10
4.	Struct та Enum. Машина станів агента. Pattern Matching	2	2	8
5.	Traits та Generics. Абстракції поведінки агента	2	2	8
6.	Багатопотоковість: threads, Arc, Mutex, channels	2	2	11
7.	Async/Await та Tokio. Actor Model для MAC	2	2	10

	ВСЬОГО	14	14	61
--	---------------	-----------	-----------	-----------

Загальний обсяг 90 год., в тому числі:

- Лекцій – 14 год.;
- Консультації – 1 год.;
- Лабораторні заняття – 14 год.;
- Самостійна робота – 61 год.

Теми лабораторних робіт:

1. Налаштування середовища Rust. Перший проєкт «Hello, Agent!»
2. Моделювання стану агента: змінні, функції, структури керування
3. Ownership та Borrowing на прикладі ресурсів агента
4. Структури та перелічення. Машина станів БПЛА
5. Traits та абстракції агента. Поліморфізм поведінки
6. Локальний рій агентів: threads, Arc, Mutex, channels
7. Асинхронний рій агентів: Tokio, Actor Model

Індивідуальне завдання (проєкт):

Розробка симуляції автономного рою БПЛА з використанням Actor Model. Проєкт включає: реалізацію агентів-БПЛА з машиною станів, комунікацію через message passing, координацію рою для виконання спільного завдання (патрулювання, пошук, доставка).

9. Рекомендовані джерела

Основні:

1. Klabnik S., Nichols C. The Rust Programming Language. 2nd ed. No Starch Press, 2023. 560 p.
2. Blandy J., Orendorff J., Tindall L. Programming Rust: Fast, Safe Systems Development. 2nd ed. O'Reilly Media, 2021. 738 p.
3. Wooldridge M. An Introduction to MultiAgent Systems. 2nd ed. Wiley, 2009. 484 p.
4. Бичков О.С. Основи сучасного програмування. Київ: ВПЦ «Київський університет», 2007. 260 с.
5. C# - творчість програмування. Том 1. Об'єктно-орієнтоване програмування: підручник // Під ред. Бичкова О.С. Авторський колектив: Іванов Є.В., Жабська Є.О., Петрівський В.Я., Меркулова К.В., Шатирко А.В., Порєв Г.В.: Волин. обереги, 2024.–292 с.

Додаткові:

1. Gjengset J. Rust for Rustaceans: Idiomatic Programming for Experienced Developers. No Starch Press, 2021. 280 p.
2. Shoham Y., Leyton-Brown K. Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press, 2009. 532 p.
3. Tokio Documentation. <https://tokio.rs/>
4. The Rust Book (офіційна документація). <https://doc.rust-lang.org/book/>

ПОЛОЖЕННЯ ПРО ОЦІНЮВАННЯ STEM ПРОЄКТУ

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: частка пройдених тест-кейсів, покриття тестами, кількість зауважень linter (програма, яка перевіряє код: **cpplint** – 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. Доброчесність як зовнішній множник. Порушення діє як множник М до підсумку ($M \times \text{MIN}(100; \Sigma + \text{Бонус})$), а не як окремий 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 М – множник академічної доброчесності (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_i = Продукт_команди (спільне) + Індивідуальне_i.

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 / Бази 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 – однаково 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 ганняє 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) PK/FK 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).
Д. 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: експертна зона повністю виконана (event-driven backend, складна авторизація, розподілені воркери тощо) → 1.0; просунута → 0.9; базова, повністю та якісно виконана → 0.8; зона виконана частково/спрощено → 0.4–0.5; тривіальний внесок (напр. README + шаблонний CRUD) → 0.2; не виконано → 0. Добре виконаний базовий рівень НЕ карається.
Е. 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	5	2 заздалегідь заготовлені мікрозміни умови; кожна

	«на льоту»		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 → 1.0; із дрібним втручанням → 0.5; не запускається → 0.
РАЗОМ (сума ваг)		100	Підсумок = MIN(100; M × Σ(вага×рівень) + Бонус)

5. Рубрика Б · Математичні дисципліни

Блок С – командна складова. Блоки А, В, D, Е – індивідуальні. Джерела даних: робочий журнал і проміжні викладки, еталонний набір задач/кейсів для звірки результатів, відтворюваний скрипт розрахунку, 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 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 ≥95%; 0.8 = 80–94%; 0.6 = 60–79%; 0.4 = 40–59%;

			$0.2 = 20\text{--}39\%$; $0 = <20\%$. Мовчазний зрив дедлайну як виконане не зараховується.
В3	Результативність взаємної перевірки розв'язків	5	Зараховуються лише результативні рецензії (перевірів розв'язок колеги → знайшов помилку або запропонував суттєве покращення → автор відреагував → перевірено виправлення). ≥ 3 результативних → 1.0; 2 → 0.8; 1 → 0.5; 0 → 0.
С. Математична якість (КОМАНДНЕ – однакове всім)			
С1	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.
С2	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.
С3	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.
С4	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).
С5	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.
Д. 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.
Е. Захист і розуміння (індивід.)			
E1	Відповіді за відкритим банком теоретичних	12	Рівень = сума за відповіді / максимум. К питань з оприлюдненого банку; кожна: повна = 1,

	питань		часткова = 0.5, немає = 0.
Е2	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.
Е3	Демонстрація відтворюваності розрахунку	3	Перерахунок: з нуля відтворює результат → 1.0; частково → 0.5; ні → 0.
РАЗОМ (сума ваг)		100	Підсумок = MIN(100; М × Σ(вага×рівень) + Бонус)

6. Академічна доброчесність і політика щодо ШІ

Доброчесність застосовується як зовнішній множник М до підсумкового бала – Оцінка = М × MIN(100; Σ + 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 проектної 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 проектної 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 неминує 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 ІІІ: 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_Д3»), 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 ВЗ (code-review/взаємоперевірка) і peer-review – рекомендовано перенести 5 балів ВЗ 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».