

**ЗАКЛАД ВИЩОЇ ОСВІТИ
«УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА»**

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

ЗАТВЕРДЖУЮ

Проректор з навчально-методичної роботи

Ярослав ШТАНЬКО

“31” серпня 2026 р.

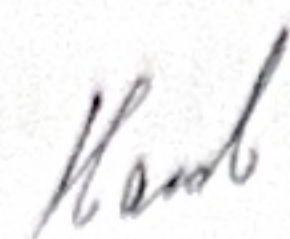
ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

СИЛАБУС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Галузь знань:	F Інформаційні технології
Спеціальність:	F2 Інженерія програмного забезпечення
Освітньо-професійна (освітньо-наукова) програма:	Розробка та тестування програмного забезпечення
Освітній рівень:	перший (бакалаврський)
Статус дисципліни:	обов’язкова
Мова викладання, навчання та оцінювання:	українська
Семестр викладання	3 семестр (2 курс)

РОЗРОБНИК:

PhD, доцент

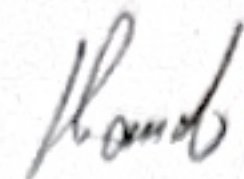


Олександр ІВАНОВ

ЗАТВЕРДЖЕНО:

на засіданні кафедри інформаційних технологій, протокол № 1 від 27 серпня 2026

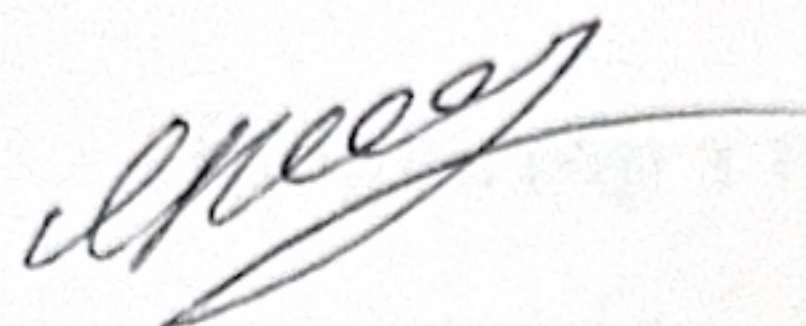
Завідувач кафедри



Олександр ІВАНОВ

УЗГОДЖЕНО:

Гарант ОПП/ОНП, к.ф-м.н.



Володимир МАКОВИШИН

СХВАЛЕНО:

на засіданні Науково-методичної ради, протокол № 1 від 31 серпня 2026 р.

e-mail	oleksandr.o.ivanov@ukd.edu.ua
Номер аудиторії чи кафедри	206
Посилання на сайт	https://ukd.edu.ua/dovidnyk/kafedra-informatsiynykh-tekhnologiy
Сторінка курсу в СДО	https://online.ukd.edu.ua/

ВСТУП

Анотація навчальної дисципліни. Курс знайомить здобувачів вищої освіти з інженерними принципами, методологіями та практичним інструментарієм проектування, розробки, верифікації та підтримки якісного програмного забезпечення. Дисципліна охоплює повний спектр життєвого циклу ПЗ: від формування бізнес-вимог і системного моделювання до конфігураційного менеджменту, неперервної інтеграції (CI/CD), всебічного тестування та моніторингу релізів. Вивчення інженерії програмного забезпечення закладає базис для побудови відмовостійких, масштабованих і безпечних цифрових продуктів в умовах динамічної ІТ-індустрії.

Мета та завдання дисципліни:

Мета: формування у студентів цілісної системи теоретичних знань та комплексних практичних умінь у сфері програмної інженерії, необхідних для проектування, реалізації, верифікації та супроводу промислових програмних рішень відповідно до світових стандартів якості.

Завдання:

- опанування класичних та гнучких моделей життєвого циклу програмних систем;
- засвоєння інженерних методів збору, структурування, аналізу та документування вимог;
- набуття навичок архітектурного проектування із застосуванням UML-діаграм та структурних/поведінкових патернів;
- опанування методологій контролю якості, розробки тест-планів, модульного та інтеграційного тестування;
- практичне освоєння сучасних інструментів командної розробки, систем контролю версій (Git) та конвеєрів CI/CD.

В результаті вивчення дисципліни студент повинен **знати**:

- моделі, фази та процеси життєвого циклу програмного забезпечення;
- стандарти інженерії вимог та способи їх валідації;
- архітектурні стилі і базові патерни GoF;
- рівні, техніки («білий», «чорний» ящик) та метрики ефективності тестування ПЗ.

В результаті вивчення дисципліни студент повинен **вміти**:¹

¹ поняття вміти і знати повинні співвідноситися з програмними результатами навчання

- застосовувати сучасні CASE-засоби для об'єктно-орієнтованого моделювання систем;
- розробляти професійну інженерну документацію на ПЗ (специфікації вимог SRS, тест-кейси, баг-репорти);
- реалізовувати модульні та інтеграційні тести з оцінкою відсотка покриття коду;
- організовувати командну роботу через системи контролю версій (Git) та автоматизувати релізні процедури.

Компетентності та результати навчання, яких набувають здобувачі освіти внаслідок вивчення навчальної дисципліни (шифри та зміст компетентностей та програмних результатів навчання вказано відповідно до ОПП/ОНП «Розробка та тестування програмного забезпечення»).

Шифр та назва компетентності	Шифр та назва програмних результатів навчання
------------------------------	---

K03: Здатність спілкуватися державною мовою як усно, так і письмово. K04: Здатність спілкуватися іноземною мовою як усно, так і письмово. K05: Здатність вчитися і оволодівати сучасними знаннями. K06: Здатність до пошуку, оброблення та аналізу інформації з різних джерел. K10: Здатність діяти соціально відповідально та свідомо. K13: Здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення. K14: Здатність брати участь у проектуванні програмного забезпечення, включаючи моделювання структури, поведінки та процесів функціонування. K16: Здатність формулювати та забезпечувати вимоги щодо якості програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами. K17: Здатність дотримуватися специфікацій, стандартів, правил і рекомендацій у професійній галузі при реалізації процесів життєвого циклу. K18: Здатність аналізувати, вибирати і застосовувати методи і засоби для забезпечення інформаційної безпеки. K20: Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв'язання завдань інженерії програмного забезпечення. K22: Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження ПЗ та визнання важливості навчання протягом всього життя. K23: Здатність реалізовувати фази та ітерації життєвого циклу програмних систем на основі відповідних моделей і підходів розробки ПЗ.	ПР01: Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки. ПР03: Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення. ПР04: Знати і застосовувати професійні стандарти та інші нормативно-правові документи в галузі інженерії програмного забезпечення. ПР06: Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення. ПР07: Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення. ПР09: Знати та вміти використовувати методи та засоби збору, формулювання та аналізу вимог до програмного забезпечення. ПР10: Проводити передпроектне обстеження предметної області, системний аналіз об'єкта програмування. ПР11: Вибирати вихідні дані для проектування, керуючись формальними методами опису вимог та моделювання. ПР15: Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення.
--	--

K24: Здатність здійснювати процес інтеграції системи, застосовувати стандарти і процедури управління змінами для підтримки надійності ПЗ.	
K25: Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення.	

ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Курс	2		
Семестр	2		
Кількість кредитів ЄКТС	6		
Аудиторні навчальні заняття		денна форма	заочна форма
	лекції	32 (в годинах)	6 (в годинах)
	практичні	28 (в годинах)	6 (в годинах)
Самостійна робота		120 (в годинах)	168 (в годинах)
Форма підсумкового контролю	Екзамен 2		

Структурно-логічна схема вивчення навчальної дисципліни²:

Пререквізити	Постреквізити
Основи роботи з нейронними мережами, Основи програмування, Математичний аналіз, Інформаційно-комунікаційні технології	Конструювання програмного забезпечення, UI/UX дизайн, Документування програмних продуктів

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Перелік тем лекційного матеріалу

Змістовий модуль 1. Життєвий цикл, специфікація вимог та архітектурне моделювання ПЗ

Тема 1. Основи інженерії програмного забезпечення (денна – 6 год., заочна – 1 год.)

Питання теми. Вступ до дисципліни; інженерний підхід у створенні ПЗ; програмна криза та її подолання; структура життєвого циклу ПЗ; каскадна, ітеративна, спіральна моделі та

² тільки для обов'язкових дисциплін

V-подібна модель; порівняння предикативних та адаптивних методологій (Agile, Scrum, Kanban); ролі та обов'язки в команді розробки.

Самостійна робота (денна – 2 год., заочна – 4 год.): Еволюція стандартів інженерії ПЗ; професійна етика та кодекс інженера-програміста IEEE-CS/ACM.

Тема 2. Збір, системний аналіз та інженерія вимог (денна – 6 год., заочна – 1 год.)

Питання теми. Інженерія вимог як фундамент проекту; класифікація вимог: функціональні, нефункціональні, бізнес-вимоги та обмеження реалізації; техніки виявлення та збору вимог; методи вирішення конфліктів у вимогах; структура інженерної специфікації вимог (SRS) за стандартом ISO/IEC/IEEE 29148; верифікація та трасування вимог.

Самостійна робота (денна – 2 год., заочна – 4 год.): Нотації специфікації вимог; розробка користувацьких історій (User Stories) та критеріїв приймання (Acceptance Criteria).

Тема 3. Проєктування архітектури програмного забезпечення (денна – 6 год., заочна – 1 год.)

Питання теми. Архітектурні принципи: висока згуртованість (High Cohesion) та слабе зв'язування (Low Coupling); архітектурні стилі: монолітна архітектура, шарувата (N-tier), клієнт-серверна, мікросервісна та подійно-орієнтована (Event-Driven); базові шаблони проєктування GoF (породжуючі, структурні, поведінкові); мова візуального моделювання UML.

Самостійна робота (денна – 3 год., заочна – 5 год.): Проєктування діаграм класів та діаграм послідовностей (Sequence Diagrams) для розподілених систем.

Змістовий модуль 2. Верифікація, забезпечення якості, конфігураційне управління та перспективи галузі

Тема 4. Забезпечення якості та тестування програмного забезпечення (денна – 6 год., заочна – 1 год.)

Питання теми. Концепції Quality Assurance (QA) та Quality Control (QC); верифікація та валідація; рівні тестування: модульне (Unit), інтеграційне, системне та приймальне (UAT); стратегії «білої», «чорної» та «сірої» скриньок; тест-дизайн: еквівалентний поділ, аналіз граничних значень; автоматизація тестування, метрики покриття коду (Code Coverage).

Самостійна робота (денна – 3 год., заочна – 5 год.): Класифікація дефектів, життєвий цикл бага, системи баг-трекінгу (Jira, GitHub Issues).

Тема 5. Управління конфігураціями, версіями та релізами (денна – 4 год., заочна – 1 год.)

Питання теми. Завдання керування конфігурацією програмного забезпечення (SCM); розподілені системи контролю версій (DVCS Git); стратегії розгалуження (GitFlow, Trunk-Based Development); управління залежностями проекту; філософія DevOps та концепція Continuous Integration & Continuous Delivery (CI/CD); основи контейнеризації.

Самостійна робота (денна – 2 год., заочна – 4 год.): Автоматизовані інструменти складання проєктів (Maven, Gradle, npm); семантичне версіонування програмних релізів (SemVer).

Тема 6. Перспективи розвитку інженерії програмного забезпечення (денна – 4 год., заочна – 1 год.)

Питання теми. Сучасні тенденції розробки: Cloud-Native архітектури, безсерверні обчислення (Serverless); когнітивна кібербезпека та DevSecOps; застосування генеративного штучного інтелекту та LLM-асистентів у процесах генерації, рефакторингу коду та виявлення вразливостей; етичні межі автоматизації програмування.

Самостійна робота (денна – 2 год., заочна – 5 год.): Дослідження сучасних інструментів автоматизованого аудиту коду та AI-асистентів розробника.

Зміст практичних занять

Практичне заняття 1. Інструментарій програмного інженера та розподілений контроль версій Git (денна – 4 год., заочна – 1 год.)

Налаштування робочого середовища та IDE. Ініціалізація Git-репозиторію, базові команди (commit, branch, checkout, merge, rebase). Моделювання конфліктів паралельної розробки та їх вирішення. Створення та оформлення Pull Request на платформі GitHub.

Самостійна робота (денна – 2 год., заочна – 4 год.): Налаштування SSH-ключів, робота з upstream-репозиторіями.

Практичне заняття 2. Практика збору, моделювання та документування вимог (денна – 6 год., заочна – 1 год.)

Формулювання технічного завдання для предметної області проєкту. Складання інженерної специфікації вимог (Software Requirements Specification, SRS).

Побудова моделі прецедентів (Use Case Diagram) за допомогою CASE-засобів.

Самостійна робота (денна – 2 год., заочна – 4 год.): Опис детальних сценаріїв використання (Use Case Specifications) та критеріїв приймання.

Практичне заняття 3. Архітектурне проектування системи та реалізація шаблонів GoF (денна – 6 год., заочна – 1 год.)

Розробка детальної структури системи: побудова діаграми класів (Class

Diagram) та діаграми взаємодії (Sequence Diagram). Програмна імплементація архітектурних рішень із застосуванням патернів проєктування (Singleton, Factory Method, Strategy, Observer).

Самостійна робота (денна – 2 год., заочна – 4 год.): Рефакторинг проєктного коду та документація архітектурних інтерфейсів.

Практичне заняття 4. Модульне та інтеграційне тестування компонентів ПЗ (денна – 6 год., заочна – 2 год.)

Розробка комплексу модульних тестів (Unit Tests) з використанням тестових бібліотек. Застосування підходів TDD (Test-Driven Development). Створення Mock-об'єктів для ізоляції тестових сценаріїв. Генерація звітів про покриття коду та фіксація виявлених дефектів у форматі Issue/Bug Report.

Самостійна робота (денна – 3 год., заочна – 5 год.): Написання тестів для граничних умов та інтеграційних сценаріїв.

Практичне заняття 5. Конфігураційне управління, налаштування CI/CD конвеєра та захист проєкту (денна – 6 год., заочна – 1 год.)

Налаштування скриптів автоматизованого складання та валідації коду.

Конфігурування пайплайну CI/CD (GitHub Actions) для автоматичного прогону тестів при кожному пуші. Фінальне збирання релізного артефакту та презентація командного проєкту.

Самостійна робота (денна – 3 год., заочна – 5 год.): Фіналізація інженерної документації репозиторію та підготовка до публічного захисту.

Зміст самостійної роботи здобувачів

Розподіл годин, виділених на вивчення дисципліни:

Найменування видів робіт	Розподіл годин за формами навчання	
	денна	заочна
Самостійна робота, год, у т.ч.:	120	128
Опрацювання матеріалу, викладеного на лекціях	19	27

Підготовка до практичних занять та контрольних заходів	30	42
Підготовка звітів з практичних робіт	19	27
Підготовка до поточного контролю	10	13
Опрацювання матеріалу, винесеного на самостійне вивчення	42	59

ПОЛІТИКА КУРСУ

Коротко, з покликанням на відповідну нормативну базу УКД, висвітлити питання.³

1) щодо системи поточного і підсумкового контролю

Організація поточного та підсумкового семестрового контролю знань студентів, проведення практик та атестації, переведення показників академічної успішності за 100-бальною шкалою в систему оцінок за національною шкалою здійснюється згідно з “Положенням про систему поточного і підсумкового контролю, оцінювання знань та визначення рейтингу здобувачів освіти”. Ознайомитись з документом можна за [посиланням](#).



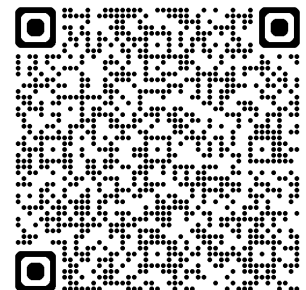
2) щодо оскарження результатів контрольних заходів

Здобувачі вищої освіти мають право на оскарження оцінки з дисципліни отриманої під час контрольних заходів. Апеляція здійснюється відповідно до «Положення про політику та врегулювання конфліктних ситуацій». Ознайомитись з документом можна за [посиланням](#).



3) щодо відпрацювання пропущених занять

Згідно “Положення про організацію освітнього процесу” здобувач допускається до семестрового контролю з **конкретної навчальної дисципліни (семестрового екзамену, диференційованого заліку)**, якщо він виконав усі види робіт, передбачені на семестр навчальним планом та **силабусом/робочою програмою навчальної дисципліни, підтвердив опанування на мінімальному рівні результатів навчання (отримав ≥ 35 бали)**, відпрацював визначені індивідуальним навчальним планом всі лекційні, практичні, семінарські та



³ зміст пунктів може редагуватись з огляду на особливості курсу

лабораторні заняття, на яких він був відсутній. Ознайомитись з документом можна за [покликанням](#).

4) щодо дотримання академічної доброчесності

“Положення про систему забезпечення академічної доброчесності” закріплює моральні принципи, норми та правила етичної поведінки, позитивного, сприятливого, доброчесного освітнього і наукового середовища, професійної діяльності та професійного спілкування спільноти Університету, викладання та провадження наукової (творчої) діяльності з метою забезпечення довіри до результатів навчання. Ознайомитись з документом можна за [покликанням](#).



5) щодо використання штучного інтелекту

“Політика використання штучного інтелекту” визначає політику щодо використання технічних засобів на основі штучного інтелекту в освітньому процесі. Ознайомитись з документом можна за [покликанням](#).⁴ “Положення про систему забезпечення академічної доброчесності” містить рекомендації щодо використання в академічних текстах генераторів на основі штучного інтелекту. Ознайомитись з документом можна за [покликанням](#).



6) щодо використання технічних засобів в аудиторії та правила комунікації

Використання мобільних телефонів, планшетів та інших гаджетів під час лекційних та практичних занять дозволяється виключно у навчальних цілях (для уточнення певних даних, перевірки правопису, отримання довідкової інформації тощо). На гаджетах повинен бути активований режим «без звуку» до початку заняття. Під час занять заборонено надсилання текстових повідомлень, прослуховування музики, перевірка електронної пошти, соціальних мереж тощо. Під час виконання заходів контролю використання гаджетів заборонено (за винятком, коли це передбачено умовами його проведення). У разі порушення цієї заборони результат анулюється без права перескладання. Ознайомитись з відповідним документом можна за [посиланням](#).



Комунікація відбувається через електронну пошту і сторінку дисципліни в Moodle.

7) щодо зарахування результатів навчання, здобутих шляхом формальної/інформальної освіти

Процедури визнання результатів навчання, здобутих шляхом формальної/інформальної освіти визначаються «Положенням про порядок визнання результатів навчання, здобутих шляхом неформальної



⁴ визначається політика використання ШІ в навчальній дисципліні - дозволене/заборонене, правила використання

та / або інформальної освіти». Ознайомитись з документом можна за [посиланням](#).⁵ Окремі теми можуть бути зараховані як вивчені за рахунок публікації одноосібних або у співавторстві статей, тез, виступів на конференціях. Можливість зарахування попередньо узгоджується з викладачем з позицій актуальності теми, журналу, конференції, тощо.

МЕТОДИ НАВЧАННЯ

При вивченні дисципліни застосовується комплекс методів для організації навчання студентів з метою розвитку їх логічного та абстрактного мислення, творчих здібностей, підвищення мотивації до навчання та формування особистості майбутнього фахівця.

Програмний результат навчання ⁶	<u>Метод навчання</u>	Метод оцінювання
ПР01, ПР04	Проблемна лекція, дослідницький метод, опрацювання нормативної бази та стандартів ISO/IEEE, критичний аналіз джерел.	Експрес-тестування знань стандартів, перевірка аналітичних звітів, письмові відповіді на теоретичні питання.
ПР03, ПР06	Інтерактивні лекції, кейс-метод, порівняльний аналіз методологій розробки (Agile vs Waterfall), дискусії.	Оцінювання командних презентацій життєвого циклу, колоквиуми, модульні контрольні роботи.
ПР09, ПР10, ПР11	Метод проектів, робота в малих групах, інтерв'ювання замовника, моделювання прецедентів у CASE-середовищах.	Захист специфікації вимог (SRS), оцінювання повноти Use Case моделей та структури вихідних даних.
ПР07, ПР15	Практикуми з розробки, код-рев'ю, налагодження ПЗ, самостійне написання модульних і інтеграційних тестів.	Захист практичних робіт у Git-репозиторії, автоматизовані тести на CI-сервері, підсумковий екзамен.

МАТЕРІАЛЬНО-ТЕХНІЧНІ РЕСУРСИ

Апаратне забезпечення: спеціалізовані комп'ютерні аудиторії кафедри ІТ з підключенням до локальної гігабітної мережі та стабільним доступом до мережі Інтернет; проекційне мультимедійне обладнання.

Програмні засоби: сучасні середовища розробки (PyCharm, Visual Studio Code); CASE-системи візуального UML-проектування (PlantUML, Draw.io, Visual Paradigm); система контролю версій Git; хмарні сервери коду GitHub та сервіси GitHub Actions; інституційна система дистанційного навчання Moodle.

ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

⁵ визначається перелік електронних та інших ресурсів та умови перезарахування

⁶ для вибіркового навчальних дисциплін вказується результат навчання

Контрольні заходи⁷

Вид контрольного заходу	Зміст заходу (етапи виконання)	% від оцінки	Min бал	Max бал
Поточні контрольні заходи (5 практичних робіт + модульний контроль):				
Практична робота 1	1.1. Налаштування середовища розробки та конфігурація робочого простору	5%	2	5
	1.2. Розподілений контроль версій Git, робота з гілками та Pull Requests	5%	2	5
Практична робота 2	2.1. Збір та класифікація функціональних/нефункціональних вимог, Use Cases	5%	2	5
	2.2. Розробка та верифікація інженерної специфікації вимог до ПЗ (SRS)	5%	2	5
Практична робота 3	3.1. Архітектурне UML-моделювання структури та динаміки системи	5%	2	5
	3.2. Програмна імплементація шаблонів проєктування GoF	5%	2	5
Практична робота 4	4.1. Розробка комплексного тест-плану та сценаріїв чорного ящика	5%	2	5
	4.2. Написання Unit-тестів, розрахунок покриття коду та Bug Tracking	5%	2	5
Практична робота 5	5.1. Налаштування CI/CD конвеєра та автоматизованої збірки проєкту	5%	2	5
	5.2. Формування релізного артефакту та публічний захист командного проєкту	5%	2	5
Модульний контроль	Модульна контрольна робота 1 (Змістовий модуль 1)	5%	2	5
	Модульна контрольна робота 2 (Змістовий модуль 2)	5%	2	5
Всього за поточний контроль:		60%	12	60
Підсумковий контрольний захід:				
Семестровий екзамен	Теоретичні завдання, інженерні кейси з архітектури та тестування ПЗ	40%	25	40
РАЗОМ ПО КУРСУ:		100%	60	100

7.2. Процедура та регламент проведення контрольних заходів⁷ зміст редагується залежно від наповнення дисципліни

Процедура проведення контрольних заходів регулюється «Положенням про систему поточного та підсумкового контролю, оцінювання знань та визначення рейтингу студентів». Фіксація поточного контролю здійснюється в «Електронному журналі обліку успішності академічної групи» за чотирибальною шкалою («2», «3», «4», «5»). У разі відсутності студента на занятті виставляється індекс «н». За результатами семестрової роботи в Журналі автоматично визначається кумулятивна підсумкова оцінка поточної роботи та ведеться облік пропусків.

Правило трьох тижнів: Усі пропущені заняття (лекційні, практичні) та отримані негативні оцінки («2») студенти зобов'язані відпрацювати **впродовж трьох наступних тижнів** з моменту виникнення заборгованості. У разі недотримання зазначеного терміну замість «н» виставляється оцінка **«0» (нуль балів) без права подальшого перездавання**. Відпрацьоване лекційне заняття фіксується літерою «в».

Умови допуску до екзамену: До складання семестрового екзамену допускаються здобувачі, які за підсумками поточного контролю набрали **не менше 35 балів** та відпрацювали всі пропущені аудиторні заняття. Студенти, які набрали **34 бали і менше**, не допускаються до складання підсумкового контролю та на підставі укладання додаткового договору зобов'язані пройти повторний курс вивчення дисципліни.

За підсумками складання екзамену здобувач максимально може набрати 40 балів. Мінімальний поріг успішного складання екзамену становить **25 балів**. Здобувачі, які під час екзамену отримали 24 бали і менше, вважаються такими, що не склали екзамен (отримують підсумковий статус FX), та повинні з'явитися на академічне перескладання згідно з графіком деканату.

7.3. Критерії оцінювання

«Відмінно» (90–100 балів, А): Здобувач демонструє всебічні, глибокі та системні знання теоретичного матеріалу; вільно володіє архітектурними концепціями, стандартами життєвого циклу ПЗ та сучасним інструментарієм; бездоганно створює UML-діаграми, пише ефективний тестовий код; оперативно та аргументовано розв'язує нестандартні інженерні кейси під час усного або комп'ютерного захисту робіт.

«Добре» (75–89 балів, В–С): Здобувач твердо й упевнено володіє програмним матеріалом; успішно застосовує його для вирішення стандартних інженерних та алгоритмічних задач; вміє самостійно аналізувати специфікації вимог та

виправляти виявлені дефекти коду; припускається незначних неточностей або дрібних синтаксичних/оформлювальних похибок, які самостійно аргументує та виправляє.

«Задовільно» (60–74 бали, D–E): Здобувач володіє навчальним матеріалом на мінімально достатньому рівні; відтворює базові концепції програмної інженерії з допомогою навідних запитань викладача; зазнає труднощів під час самостійної розробки складної архітектури або написання автоматизованих тестів, однак володіє ключовими термінами й техніками для розв'язання типових завдань.

«Незадовільно» з можливістю повторного складання (35–59 балів, FX): Здобувач володіє окремими розрізненими фрагментами навчального матеріалу; не в змозі виконати базові практичні завдання без суттєвих критичних помилок; має академічну заборгованість, але виконав необхідний мінімум поточної роботи для отримання права на додаткову заліково-екзаменаційну сесію.

«Незадовільно» з обов'язковим повторним вивченням (0–34 бали, F): Здобувач не засвоїв основи дисципліни; практичні завдання не виконані або виконані незадовільно; не з'явився на обов'язкові заходи або має систематичні пропуски; не допущений до семестрового контролю і підлягає повторному вивченню навчальної дисципліни.

7.4 Шкала оцінювання за ЄКТС

Шкала оцінювання знань за ЄКТС:

Оцінка за національною шкалою	Рівень досягнень, %	Шкала ECTS
Національна диференційована шкала		
Відмінно	90 – 100	A
Добре	83 – 89	B
	75 – 82	C
Задовільно	67 – 74	D
	60 – 66	E
Незадовільно	35 – 59	FX
	0 – 34	F
Національна недиференційована шкала		

Зараховано	60 – 100	-
Не зараховано	0 – 59	-

Студенти, які не з'явилися на заліки/екзамени без поважних причин, вважаються такими, що одержали незадовільну оцінку.

РЕКОМЕНДОВАНІ ДЖЕРЕЛА ІНФОРМАЦІЇ⁸

Основна література

1. Грицюк Ю. І. Інженерія програмного забезпечення : підручник. Львів : Сполом, 2021. 468 с.
2. Лаврищева К. М. Програмна інженерія : підручник. 2-ге вид., допов. і виправ. Київ : Академперіодика, 2018. 384 с.
3. Баран Б. А. Тестування програмного забезпечення : навч. посіб. Львів : Видавництво Львівської політехніки, 2021. 248 с.
4. Каленюк А. А. Архітектура програмних систем : навч. посіб. Київ : КПІ ім. Ігоря Сікорського, 2022. 182 с.
5. Литвинов В. В., Голуб М. Ю. Системний аналіз та проєктування програмних систем : навч. посіб. Чернігів : НУ «Чернігівська політехніка», 2021. 216 с.

Додаткова література

1. Sommerville I. Software Engineering. 10th ed. Harlow : Pearson Education, 2016. 816 p.
2. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner's Approach. 9th ed. New York : McGraw-Hill Education, 2020. 944 p.
3. Winters T., Manshreck T., Wright H. Software Engineering at Google: Lessons Learned from Programming Over Time. Sebastopol : O'Reilly Media, 2020. 599 p.
4. Martin R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Boston : Prentice Hall, 2017. 432 p.
5. Crispin L., Gregory J. Agile Testing: A Practical Guide for Testers and Agile Teams. Boston : Addison-Wesley Professional, 2019. 576 p.
6. Шаховська Н. Б., Гощинський Г. І., Пасічник В. В. Об'єктно-орієнтоване програмування : навч. посіб. Львів : Магнолія-2006, 2016. 424 с.

Інформаційні ресурси в інтернеті та нормативні документи

1. Інженерія систем і програмних засобів. Процеси життєвого циклу. Інженерія вимог (ISO/IEC/IEEE 29148:2018, IDT) : ДСТУ ISO/IEC/IEEE 29148:2025. [Чинний від 2025-12-01]. Київ : ДП «УкрНДНЦ», 2025. 104 с.
2. Guide to the Software Engineering Body of Knowledge (SWEBOK Guide). Version 3.0 / ed. by P. Bourque, R. E. Fairley. IEEE Computer Society. URL: <https://www.computer.org/education/bodies-of-knowledge/software-engineering> (дата звернення: 11.09.2026).
3. Manifesto for Agile Software Development. URL: <https://agilemanifesto.org/> (дата звернення: 11.09.2026).
4. Git Documentation : official manual. URL: <https://git-scm.com/doc> (дата звернення: 11.09.2026).
5. GitHub Documentation. URL: <https://docs.github.com/> (дата звернення: 11.09.2026).

⁸ обов'язково: враховувати вимоги [ДСТУ 8302:2015](#) (відповідно до [Наказу № 65, від 4.03. 2016](#)), [рекомендації](#) Національного агентства з забезпечення якості вищої освіти, використовувати літературу за останні 5-7 років, наводити власні публікації за змістом навчальної дисципліни.

