Міжнародний гуманітарний університет
Факультет кібербезпеки, програмної інженерії
та комп'ютерних наук
Кафедра комп'ютерної інженерії та інноваційних технологій

«ЗАТВЕРДЖУЮ»
Ректор Міжнародного
гуманітарного університету
д.ю.н., професор
_____ Костянтин ГРОМОВЕНКО
«29» серпня 2025 р.

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Галузь знань 12 Інформаційні технології

Спеціальність 121 Інженерія програмного забезпечення

Назва освітньої програми Інженерія програмного забезпечення

Рівень вищої освіти перший (бакалаврський)

Одеса – 2025 рік

Робоча програма затверджена на засіданні кафедри комп'ютерної інженерії та інноваційних технологій.

Протокол № 1 від 26 серпня 2025 року.

Розробники і викладачі	Контактний тел.	E-mail
доцент кафедри комп'ютерної інженерії та інноваційних технологій, кандидат технічних наук, доцент, **Педяш Володимир Віталійович**	+380673787003	vpedyash@gmail.com

Завідувачка кафедри комп'ютерної інженерії
та інноваційних технологій _____ Лариса ЙОНА

Гарант освітньої програми
к.т.н., доцент _____ Олександр РУСУ

Узгоджено
Начальник навчального відділу _____ Лілія САЄНКО

1. ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

<table>
<tr><td rowspan="2">Найменування показників</td><td rowspan="2">Галузь знань, напрям підготовки, освітньо-кваліфікаційний рівень</td><td colspan="2">Характеристика навчальної дисципліни</td></tr>
<tr><td rowspan="2">Кількість кредитів – 6,
загальна кількість годин – 180</td><td rowspan="2">Галузь –
12 Інформаційні технології

Спеціальність –
121 Інженерія програмного забезпечення</td><td>Денна форма</td><td>Заочна форма</td></tr>
<tr><td colspan="2">обов’язкова</td></tr>
<tr><td rowspan="11">Мова навчання – українська</td><td rowspan="11">Рівень вищої освіти – перший (бакалаврський)</td><td colspan="2">Рік підготовки:</td></tr>
<tr><td colspan="2">2</td></tr>
<tr><td colspan="2">Семестр:</td></tr>
<tr><td colspan="2">3</td></tr>
<tr><td colspan="2">Лекційні заняття:</td></tr>
<tr><td>40</td><td>12</td></tr>
<tr><td colspan="2">Практичні заняття:</td></tr>
<tr><td>38</td><td>12</td></tr>
<tr><td colspan="2">Лабораторні роботи</td></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td colspan="2">Самостійна робота</td></tr>
<tr><td>102</td><td>156</td></tr>
<tr><td colspan="2">Вид контролю:</td></tr>
<tr><td colspan="2">екзамен</td></tr>
</table>

Дисципліна «Об’єктно-орієнтоване програмування» спрямована на формування системного розуміння принципів та методів розробки програмного забезпечення на основі об’єктно-орієнтованої парадигми програмування. Курс охоплює основні концепції об’єктно-орієнтованого підходу, зокрема поняття класу та об’єкта, інкапсуляцію, успадкування, поліморфізм та абстракцію. Значна увага приділяється моделюванню предметної області за допомогою класів і об’єктів, побудові ієрархій класів, організації взаємодії між програмними компонентами та повторному використанню програмного коду. У межах дисципліни розглядаються підходи до структуризації програмних систем, організації модулів і пакетів, а також принципи компонентної побудови програмного забезпечення.

У межах дисципліни вивчаються механізми створення класів і об’єктів, методи та конструктори класів, організація доступу до атрибутів об’єктів і взаємодія між об’єктами у програмній системі. Значна увага приділяється використанню механізмів успадкування для розширення функціональності програмних систем, реалізації поліморфної поведінки об’єктів, а також застосуванню абстрактних класів і інтерфейсів для побудови розширюваних програмних систем. Практична складова дисципліни орієнтована на набуття здобувачами навичок проєктування класів, реалізації об’єктно-орієнтованих програмних структур, використання модулів і пакетів мови програмування Python, а також розробки програмних систем із використанням принципів об’єктно-орієнтованого програмування.

МЕТА ДИСЦИПЛІНИ. Метою викладання навчальної дисципліни є формування у здобувачів здатності застосовувати об’єктно-орієнтований підхід до розробки програмного забезпечення, проєктувати структуру програмних систем на основі класів і об’єктів, використовувати механізми інкапсуляції, успадкування, поліморфізму та абстракції, а також реалізовувати програмні системи засобами мови програмування Python.

Опанування дисципліни забезпечує розвиток умінь моделювати предметну область у вигляді системи взаємодіючих об’єктів, проєктувати ієрархії класів, реалізовувати програмні

компоненти та організовувати їх взаємодію у програмній системі. Здобувачі набувають навичок розробки модульних програм, повторного використання програмного коду, використання механізмів успадкування та поліморфізму, а також застосування сучасних підходів до побудови об'єктно-орієнтованих програмних систем.

ПРЕРЕКВІЗИТИ. Необхідною умовою для опанування дисципліни є базові знання з алгоритмізації та програмування, а також основ алгоритмів і структур даних, що формуються під час вивчення дисциплін «Алгоритмізація та програмування» та «Алгоритми та структури даних». Здобувач повинен розуміти принципи побудови алгоритмів, основи програмування та структури програмного коду.

ПОСТРЕКВІЗИТИ. Знання та вміння, отримані під час вивчення дисципліни, використовуються під час опанування навчальних компонент «Організація баз даних та знань», «Програмування мікроконтролерів та пристроїв Інтернету речей», «Вебтехнології та вебдизайн», «Патерни та фреймворки», «Програмування інфокомунікаційних сервісів та систем», а також у процесі виконання кваліфікаційної роботи, де необхідно проєктувати архітектуру програмного забезпечення та реалізовувати програмні системи із використанням об'єктно-орієнтованих технологій.

2. ОЧІКУВАНІ КОМПЕТЕНТНОСТІ, ЯКІ ПЛАНУЄТЬСЯ СФОРМУВАТИ, ТА ДОСЯГНЕННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

У процесі реалізації програми дисципліни формуються наступні компетентності із передбачених освітньою програмою:

Інтегральна компетентність

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

Загальні компетентності

К01. Здатність до абстрактного мислення, аналізу та синтезу.

К07. Здатність працювати в команді.

Спеціальні (фахові, предметні) компетентності

К14. Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування.

К15. Здатність розробляти архітектури, модулі та компоненти програмних систем.

К23. Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення.

К26. Здатність до алгоритмічного та логічного мислення.

Результати навчання

ПР05. Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об'єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення.

ПР12. Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення.

ПР13. Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.

ПР15. Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення.

ПР17. Вміти застосовувати методи компонентної розробки програмного забезпечення.

Заплановані результати навчання за навчальною дисципліною

Знання:

– основні поняття, категорії та принципи об'єктно-орієнтованого програмування;

– методи об'єктно-орієнтованого аналізу та моделювання предметної області під час розробки програмного забезпечення;

– принципи проєктування програмних систем на основі класів і об'єктів;

– підходи до розробки алгоритмів, програмних модулів та структур даних у об'єктно-орієнтованих програмних системах;

– механізми інкапсуляції, успадкування, поліморфізму та абстракції;

– принципи побудови архітектури програмних систем та організації взаємодії програмних компонентів;

– особливості застосування мови програмування Python для реалізації об'єктно-орієнтованих програмних систем.

Уміння:

– застосовувати методи об'єктно-орієнтованого аналізу для моделювання предметної області;

– проєктувати структуру програмних систем на основі класів, модулів та програмних компонентів;

– розробляти алгоритми та реалізовувати програмні рішення з використанням об'єктно-орієнтованого підходу;

– використовувати механізми інкапсуляції, успадкування, поліморфізму та абстракції під час розробки програмного забезпечення;

– реалізовувати програмні компоненти та забезпечувати їх взаємодію у програмній системі;

– обґрунтовано використовувати можливості мови програмування Python для розв'язання задач розробки програмного забезпечення.

Навички:

– проєктування структури програмних систем із використанням об'єктно-орієнтованого підходу;

– розробки програмних модулів, класів і компонентів мовою програмування Python;

– побудови ієрархій класів та організації взаємодії між програмними компонентами;

– застосування алгоритмічного та логічного мислення під час розробки програмного забезпечення;

– використання методів компонентної розробки програмного забезпечення;

– реалізації програмних систем відповідно до сучасних підходів проєктування програмного забезпечення.

3. ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Тема 1. Основи об’єктно-орієнтованого програмування

Парадигми програмування. Процедурний, модульний та об’єктно-орієнтований підходи до розробки програмного забезпечення. Переваги використання об’єктно-орієнтованої парадигми у сучасних програмних системах. Основні принципи об’єктно-орієнтованого програмування.

Основні поняття об’єктно-орієнтованого програмування. Клас і об’єкт. Стан та поведінка об’єкта. Атрибути та методи класу. Представлення предметної області у вигляді системи об’єктів.

Мова програмування Python як інструмент реалізації об’єктно-орієнтованого підходу. Особливості синтаксису Python для визначення класів і створення об’єктів. Структура програмного коду та правила іменування програмних елементів.

Створення та використання об’єктів. Ініціалізація об’єктів та керування їх станом. Взаємодія між об’єктами у програмній системі.

Тема 2. Класи, методи та конструювання об’єктів

Структура класу в об’єктно-орієнтованій програмі. Поля класу та методи класу. Призначення методів у програмній системі. Виклик методів та передавання параметрів.

Методи класів у Python. Методи екземпляра. Класові методи та статичні методи. Особливості їх використання у програмних системах.

Конструктори об’єктів. Призначення конструктора та механізм ініціалізації об’єктів. Конструктор за замовчуванням і конструктор із параметрами.

Спеціальні механізми доступу до елементів класу. Посилання на поточний об’єкт. Організація програмного коду. Модулі та пакети Python.

Тема 3. Інкапсуляція та взаємодія об’єктів

Принцип інкапсуляції в об’єктно-орієнтованому програмуванні. Приховування даних та забезпечення цілісності об’єктів. Організація доступу до атрибутів класу.

Властивості та методи доступу до даних. Методи отримання і зміни значень атрибутів. Контроль доступу до внутрішнього стану об’єкта.

Взаємодія об’єктів у програмній системі. Передавання повідомлень між об’єктами. Використання об’єктів як атрибутів інших об’єктів.

Відносини між класами. Асоціація, агрегація та композиція. Моделювання складних систем за допомогою об’єктно-орієнтованих конструкцій.

Тема 4. Успадкування та поліморфізм

Успадкування як механізм повторного використання програмного коду. Ієрархія класів. Базові та похідні класи.

Особливості реалізації успадкування у Python. Розширення функціональності класів. Використання успадкованих методів і атрибутів.

Перевизначення методів у похідних класах. Динамічне зв’язування методів. Використання механізму виклику методів базового класу.

Поліморфізм у об’єктно-орієнтованому програмуванні. Поліморфізм часу виконання та реалізація поліморфної поведінки об’єктів у Python.

Тема 5. Абстракція та компонентна організація програмних систем

Поняття абстракції у програмуванні. Абстрактні класи та їх призначення у процесі розробки програмного забезпечення.

Абстрактні методи та визначення інтерфейсів взаємодії між компонентами програмної системи. Реалізація абстрактних класів у Python.

Інтерфейсний підхід до проєктування програмних систем. Роль інтерфейсів у забезпеченні розширюваності програмного забезпечення.

Компонентна організація програмних систем. Модульність програмного забезпечення та повторне використання програмних компонентів.

4. СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Назви змістових модулів і тем	Кількість годин									
	Денна форма					Заочна форма				
	Заг.	у тому числі				Заг.	у тому числі			
		Лек.	Прак.	Лаб.	Сам. роб.		Лек.	Прак.	Лаб.	Сам. роб.
Тема 1. Основи об’єктно-орієнтованого програмування	32	8	8	0	16	32	2	2	0	28
Тема 2. Класи, методи та конструювання об’єктів	36	8	10	0	18	36	2	2	0	32
Тема 3. Інкапсуляція та взаємодія об’єктів	36	8	8	0	20	36	2	2	0	32
Тема 4. Успадкування та поліморфізм	38	8	8	0	22	38	4	4	0	30
Тема 5. Абстракція та компонентна організація програмних систем	38	8	4	0	26	38	2	2	0	34
Загалом	**180**	**40**	**38**	**0**	**102**	**180**	**12**	**12**	**0**	**156**
Підсумковий контроль – екзамен										

5. ПИТАННЯ ДО ПРАКТИЧНИХ РОБІТ

Назва теми	Кількість годин	
	Денна форма	Заочна форма
Тема 1. Основи об’єктно-орієнтованого програмування 1. Аналіз основних парадигм програмування та визначення особливостей об’єктно-орієнтованого підходу до розробки програмного забезпечення. 2. Розробка простої моделі предметної області у вигляді системи класів та об’єктів. 3. Створення класів у Python та реалізація атрибутів і методів класу. 4. Створення об’єктів класу та дослідження взаємодії об’єктів у програмі.	8	2
Тема 2. Класи, методи та конструювання об’єктів 1. Реалізація методів класу та дослідження механізму передавання параметрів і повернення результатів. 2. Реалізація різних типів методів у Python: методів екземпляра, класових і статичних методів.	10	2

3. Створення конструкторів класу та дослідження механізму ініціалізації об’єктів. 4. Реалізація класів із використанням конструкторів із параметрами. 5. Організація програмного коду з використанням модулів і пакетів Python.		
Тема 3. Інкапсуляція та взаємодія об’єктів 1. Реалізація інкапсуляції у класах Python та організація доступу до атрибутів об’єкта. 2. Створення методів доступу до даних об’єкта та контроль зміни стану об’єкта. 3. Реалізація взаємодії між об’єктами різних класів у програмній системі. 4. Моделювання відносин між класами на основі асоціації, агрегації та композиції.	8	2
Тема 4. Успадкування та поліморфізм 1. Реалізація успадкування класів та побудова ієрархії класів у Python. 2. Дослідження механізму розширення функціональності похідних класів. 3. Реалізація перевизначення методів у похідних класах. 4. Дослідження механізму поліморфізму під час виконання програм.	8	4
Тема 5. Абстракція та компонентна організація програмних систем 1. Реалізація абстрактних класів та абстрактних методів у Python. 2. Реалізація інтерфейсного підходу до проєктування програмних систем.	4	2
Загалом	**38**	**12**

6. САМОСТІЙНА РОБОТА

До самостійної роботи здобувачів відносяться наступні види діяльності:

1. Опрацювання лекційного матеріалу.
2. Підготовка до практичних занять.
3. Консультації з викладачем впродовж семестру.
4. Ознайомлення з додатковою літературою відповідно до зазначених у програмі тем.
5. Самостійне опрацювання окремих питань навчальної дисципліни.
6. Підготовка індивідуальних завдань у вигляді презентацій або рефератів.
7. Підготовка до підсумкового контролю.

Назва теми	**Кількість годин**	
	Денна форма	**Заочна форма**
Тема 1. Основи об’єктно-орієнтованого програмування 1. Основні парадигми програмування. Порівняльний аналіз процедурного, модульного та об’єктно-орієнтованого підходів до розробки програмного забезпечення. 2. Основні принципи об’єктно-орієнтованого програмування: абстракція, інкапсуляція, успадкування та поліморфізм. 3. Поняття класу та об’єкта. Моделювання предметної області за допомогою об’єктів і класів. 4. Структура класу у Python. Атрибути, методи та інтерфейс класу. 5. Створення об’єктів та керування їх станом у Python. Життєвий цикл об’єкта у програмній системі.	16	28
Тема 2. Класи, методи та конструювання об’єктів 1. Методи класу та їх роль у програмній системі. Передавання параметрів та повернення значень. 2. Методи екземпляра, класові та статичні методи у Python. Особливості їх використання. 3. Конструктори класів у Python. Механізм ініціалізації об’єктів. 4. Використання параметризованих конструкторів для ініціалізації об’єктів. 5. Посилання на поточний об’єкт та взаємодія методів одного класу. 6. Організація програмного коду у Python. Модулі та пакети. Структурування	18	32

програмних систем.		
Тема 3. Інкапсуляція та взаємодія об'єктів 1. Інкапсуляція у програмних системах. Приховування даних і контроль доступу до атрибутів класу. 2. Методи доступу до атрибутів класу та контроль зміни стану об'єкта. 3. Властивості у Python та механізм керування доступом до даних об'єкта. 4. Взаємодія об'єктів у програмній системі. Передавання повідомлень між об'єктами. 5. Використання об'єктів як атрибутів інших об'єктів. 6. Асоціація, агрегація та композиція як відносини між класами. 7. Моделювання складних систем за допомогою об'єктно-орієнтованих конструкцій.	20	32
Тема 4. Успадкування та поліморфізм 1. Успадкування класів у об'єктно-орієнтованому програмуванні. Ієрархії класів. 2. Особливості реалізації успадкування у Python. Базові та похідні класи. 3. Розширення функціональності класів за допомогою успадкування. 4. Перевизначення методів у похідних класах. 5. Використання механізму виклику методів базового класу. 6. Поліморфізм у об'єктно-орієнтованому програмуванні. 7. Реалізація поліморфізму у Python.	22	30
Тема 5. Абстракція та компонентна організація програмних систем 1. Абстракція у програмуванні. Абстрактні класи та їх роль у проєктуванні програмного забезпечення. 2. Абстрактні методи та інтерфейси взаємодії між компонентами програмної системи. 3. Реалізація абстрактних класів у Python. 4. Інтерфейсний підхід до проєктування програмних систем. 5. Компонентна організація програмних систем. 6. Модульність програмного забезпечення та повторне використання програмних компонентів. 7. Архітектурні підходи до побудови об'єктно-орієнтованих програмних систем.	26	34
Загалом	**102**	**156**

7. ВИДИ ТА МЕТОДИ КОНТРОЛЮ

Види контролю	Складові оцінювання
Поточний контроль, який здійснюється під час проведення практичних (лабораторних) занять, виконання самостійної роботи, індивідуальних завдань, дослідної роботи здобувача тощо	50%
Підсумковий контроль, який здійснюється у ході проведення екзамену	50%

Методи діагностики знань (контролю)	Фронтальне опитування, розв'язання практичних завдань, тестування здобувачів, наукові доповіді, реферати, усні повідомлення, екзамен

Питання для підсумкового контролю

1. Парадигми програмування. Процедурний, модульний та об'єктно-орієнтований підходи до розробки програмного забезпечення.
2. Основні принципи об'єктно-орієнтованого програмування: абстракція, інкапсуляція, успадкування та поліморфізм.
3. Поняття класу та об'єкта у об'єктно-орієнтованому програмуванні.
4. Стан та поведінка об'єкта. Атрибути та методи класу.

5. Моделювання предметної області за допомогою класів і об’єктів.
6. Структура класу у Python. Атрибути класу та методи класу.
7. Створення об’єктів класу та керування їх станом.
8. Методи класів у Python. Методи екземпляра, класові та статичні методи.
9. Передавання параметрів у методах та повернення результатів.
10. Конструктори класів у Python. Призначення та механізм ініціалізації об’єктів.
11. Конструктор за замовчуванням та конструктор із параметрами.
12. Посилання на поточний об’єкт та взаємодія методів одного класу.
13. Організація програмного коду у Python. Модулі та пакети.
14. Інкапсуляція у об’єктно-орієнтованому програмуванні. Приховування даних та контроль доступу до атрибутів.
15. Методи доступу до атрибутів класу та керування станом об’єкта.
16. Властивості у Python та їх використання для керування доступом до даних.
17. Взаємодія об’єктів у програмній системі. Передавання повідомлень між об’єктами.
18. Використання об’єктів як атрибутів інших об’єктів.
19. Відносини між класами: асоціація, агрегація та композиція.
20. Моделювання складних програмних систем за допомогою об’єктно-орієнтованих конструкцій.
21. Успадкування у об’єктно-орієнтованому програмуванні. Базові та похідні класи.
22. Ієрархії класів та повторне використання програмного коду.
23. Особливості реалізації успадкування у Python.
24. Розширення функціональності класів за допомогою успадкування.
25. Перевизначення методів у похідних класах.
26. Виклик методів базового класу у похідному класі.
27. Поліморфізм у об’єктно-орієнтованому програмуванні.
28. Поліморфізм часу виконання у Python.
29. Поліморфна поведінка об’єктів у програмній системі.
30. Абстракція у програмуванні та її роль у проєктуванні програмного забезпечення.
31. Абстрактні класи та їх призначення.
32. Абстрактні методи та механізм їх реалізації у Python.
33. Інтерфейсний підхід до проєктування програмних систем.
34. Контракти взаємодії між компонентами програмної системи.
35. Компонентна організація програмного забезпечення.
36. Модульність програмного забезпечення та повторне використання програмних компонентів.
37. Архітектурні підходи до побудови об’єктно-орієнтованих програмних систем.
38. Використання об’єктно-орієнтованого підходу у сучасній розробці програмного забезпечення.
39. Переваги та обмеження об’єктно-орієнтованого програмування.
40. Практичні аспекти застосування об’єктно-орієнтованого підходу під час розробки програмних систем.

8. ОЦІНЮВАННЯ ПОТОЧНОЇ, САМОСТІЙНОЇ ТА ІНДИВІДУАЛЬНОЇ РОБОТИ ЗДОБУВАЧІВ

Види роботи	Планові терміни виконання	Форми контролю та звітності	Максимальний відсоток оцінювання
Систематичність і активність роботи на практичних заняттях			
1.1. Підготовка до практичних занять	Відповідно до робочої програми та розкладу занять	Перевірка обсягу та якості засвоєного матеріалу під час практичних занять	**30**
Виконання завдань для самостійного опрацювання			
1.2. Індивідуальне тестування	Відповідно до робочої програми та розкладу занять	Перевірка результатів індивідуального тестування, перевірка конспектів лекцій, перевірка рефератів	**10**
Виконання індивідуальних завдань (дослідна робота здобувача)			
1.3. Інші види індивідуальних завдань, в т.ч. підготовка наукових публікацій, участь у роботі круглих столів, конференцій тощо.	Відповідно до робочої програми та розкладу занять	Обговорення результатів проведеної роботи під час аудиторних занять, наукових конференцій та круглих столів.	**10**
Разом балів за поточний контроль			**50**
Підсумковий контроль – екзамен			**50**
Загалом балів			**100**

Поточний контроль знань здобувачів при підсумковому контролі у формі екзамену (максимум 50 балів) оцінюється за шкалою («2», «3», «4», «5»), з обов'язковим переведенням балів за кожний вид роботи таким чином:

– за підготовку до аудиторних занять (максимум 30 балів) здобувачу освіти необхідно мати не менше 1/2 оцінок від загальної кількості практичних (лабораторних) занять, для переводу балів за підготовку до аудиторних занять середньоарифметична оцінка множиться на коефіцієнт 6;

– за виконання завдань для самостійного опрацювання (тестування, завдання, підготовка реферату за заданою тематикою) (максимум 10 балів) здобувачу освіти необхідно мати не менше 1/2 оцінок від загальної кількості програмного матеріалу, що виноситься на самостійне вивчення, для переводу зазначених балів середньоарифметична оцінка множиться на коефіцієнт 2;

– за інші види індивідуальних завдань, в т.ч. підготовку наукових публікацій, участь у роботі круглих столів, конференцій тощо (максимум 10 балів) здобувачу освіти необхідно мати не менше 1/2 оцінок від загальної кількості інших видів індивідуальних завдань, для переводу балів за підготовку до аудиторних занять середньоарифметична оцінка множиться на коефіцієнт 2.

Критерії оцінювання поточного контролю

– **«2»** – виставляється за неправильну відповідь (письмову роботу), яка не відповідає змісту теми та свідчить про нерозуміння її основних положень;

– **«3»** – виставляється за відповідь (письмову роботу), яка відповідає змісту теми, але є неповною і неточною у визначенні понять, свідчить про неповне розуміння основних положень навчального матеріалу, свідчить про те, що знання здобувача мають фрагментарний, поверховий характер;

– **«4»** – оцінюється правильна і обґрунтована відповідь (письмова робота), з якої видно, що здобувач знає й розуміє теоретичний матеріал в обсязі навчальної теми, володіє необхідними навичками, не допускає суттєвих помилок.

– **«5»** – оцінюється повна, правильна, ґрунтовна відповідь (письмова робота) на запитання теми, що передбачає логічність і послідовність викладу матеріалу, володіння термінологією, показує вміння повно й глибоко використовувати теоретичні знання в практичній площині.

Критерії оцінювання підсумкового контролю у формі екзамену

Максимальна оцінка підсумкового контролю у формі екзамену не може перевищувати 50 балів. При цьому рівень знань оцінюється:

– **від 40 до 50 балів** – здобувач виявляє особливі творчі здібності, вміє самостійно знаходити та опрацьовувати необхідну інформацію, демонструє знання матеріалу, проводить узагальнення і висновки;

– **від 30 до 40 балів** – здобувач володіє знаннями матеріалу, але допускає незначні помилки у формуванні термінів, категорій, проте за допомогою викладача швидко орієнтується і знаходить правильні відповіді;

– **від 20 до 30 балів** – здобувач відтворює значну частину теоретичного матеріалу, виявляє знання і розуміння основних положень, з допомогою викладача може аналізувати навчальний матеріал, але дає недостатньо обґрунтовані, невичерпні відповіді, допускає помилки.

– **від 10 до 20 балів** – здобувач володіє навчальним матеріалом на середньому рівні, допускає помилки, серед яких є значна кількість суттєвих;

– від 0 до 10 балів – здобувач володіє навчальним матеріалом на рівні, вищому за початковий, значну частину його відтворює на репродуктивному рівні, на всі запитання дає необґрунтовані, невичерпні відповіді, допускає помилки.

9. ОЦІНЮВАННЯ ЗАГАЛЬНИХ РЕЗУЛЬТАТІВ ЗНАНЬ ЗДОБУВАЧІВ

Загальні результати знань здобувачів по даній дисципліні оцінюються наступним чином:

– **«Відмінно»/«зараховано» (А)** – від 90 до 100 балів. Здобувач виявляє особливі творчі здібності, вміє самостійно знаходити та опрацьовувати необхідну інформацію, демонструє знання матеріалу, проводить узагальнення і висновки. Був присутній на лекціях та практичних заняттях, під час яких давав вичерпні, обґрунтовані, теоретично і практично правильні відповіді, має конспект з виконаними завданнями до самостійної роботи, презентував реферат за заданою тематикою, проявляє активність і творчість у науково-дослідній роботі;

– **«Добре»/«зараховано» (В)** – від 82 до 89 балів. Здобувач володіє знаннями матеріалу, але допускає незначні помилки у формуванні термінів, категорій, проте за допомогою викладача швидко орієнтується і знаходить правильні відповіді. Був присутній на лекціях та практичних заняттях, має конспект з виконаними завданнями до самостійної роботи, презентував реферат за заданою тематикою, проявляє активність і творчість у науково-дослідній роботі;

– **«Добре»/«зараховано» (C)** – від 74 до 81 балів. Здобувач відтворює значну частину теоретичного матеріалу, виявляє знання і розуміння основних положень, з допомогою викладача може аналізувати навчальний матеріал, але дає недостатньо обґрунтовані, невичерпні відповіді, допускає помилки. При цьому враховується наявність конспекту з виконаними завданнями до самостійної роботи, реферату та активність у науково-дослідній роботі;

– **«Задовільно»/«зараховано» (D)** – від 64 до 73 балів. Здобувач був присутній не на всіх лекціях та практичних заняттях, володіє навчальним матеріалом на середньому рівні, допускає помилки, серед яких є значна кількість суттєвих. При цьому враховується наявність конспекту з виконаними завданнями до самостійної роботи та рефератів;

– **«Задовільно»/«зараховано» (E)** – від 60 до 63 балів. Здобувач був присутній не на всіх лекціях та практичних заняттях, володіє навчальним матеріалом на рівні, вищому за початковий, значну частину його відтворює на репродуктивному рівні, на всі запитання дає необґрунтовані, невичерпні відповіді, допускає помилки, має неповний конспект з завданнями до самостійної роботи.

– **«Незадовільно з можливістю повторного складання»/«не зараховано» (FX)** – від 35 до 59 балів. Здобувач володіє матеріалом на рівні окремих фрагментів, що становлять незначну частину навчального матеріалу.

– **«Незадовільно з обов'язковим повторним вивченням дисципліни»/«не зараховано» (F)** – від 0 до 34 балів. Здобувач не володіє навчальним матеріалом.

Таблиця відповідності результатів контролю знань за різними шкалами

100-бальна шкала	Шкала за ECTS	Національна шкала	
		Екзамен	Залік
90-100	A	Відмінно	Зараховано
82-89	B	Добре	Зараховано
74-81	C		
64-73	D	Задовільно	Зараховано
60-63	E		
35-59	FX	Незадовільно	Не зараховано
0-34	F		

10. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна

1. Phillips, Dusty. Python 3 Object-Oriented Programming: Build Robust and Maintainable Software with Object-Oriented Design Patterns in Python 3.8. 3rd ed. Packt Publishing, 2018. 466p. ISBN: 9781789617078, 1789617073.
2. Lott, Steven F. Mastering Object-Oriented Python. 2nd ed. Packt Publishing, 2019. 770p. ISBN: 9781789531404, 1789531403.
3. Lott, Steven F., Phillips, Dusty. Python Object-Oriented Programming. 4th ed. Packt Publishing, 2021. 714p. ISBN: 9781801075237, 1801075239.
4. Ramalho, L. Fluent Python. O'Reilly Media. 2022. 1014p. ISBN: 9781492056324, 1492056324.
5. Thomas, D., Hunt, A. The Pragmatic Programmer: Your Journey to Mastery, 20th Anniversary Edition. Pearson Education. 2019. 352. ISBN: 9780135956915, 0135956919.

Допоміжна

6. Martin, R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Pearson Education. 2018. 432p. ISBN 9780134494166.

7. Fowler, M. Refactoring: Improving the Design of Existing Code. Pearson Education. 2018. 448p. ISBN: 9780134757704, 013475770X.

8. Robert Johnson Object-Oriented Programming with Python: Best Practices and Patterns. 2024. HiTeX Press. 2024. 519p.

9. Sina, L. Algorithms and Data Structures. BoD - Books on Demand. 2025. 132p. ISBN: 9783819205330, 3819205330.

10. Goodrich, Michael T., Tamassia, Roberto, Goldwasser, Michael H. Data Structures and Algorithms in Python. Wiley, 2013. 768p. ISBN: 9781118290279, 1118290275.

Інформаційні ресурси

11. Python Software Foundation. Python Documentation. URL: https://docs.python.org
12. Python Enhancement Proposals (PEP). URL: https://peps.python.org
13. Real Python Tutorials. URL: https://realpython.com
14. Packt Programming Resources. URL: https://www.packtpub.com
15. Stack Overflow Developer Community. URL: https://stackoverflow.com