ЗАКЛАД ВИЩОЇ ОСВІТИ
«УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА»

Факультет суспільних і прикладних наук

Кафедра інформаційних технологій

ЗАТВЕРДЖУЮ
Проректор з методичної роботи
_____ Ярослав ШТАНЬКО
"28" 08 2025 р.

Основи програмування

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

Івано-Франківськ
2025

РОЗРОБНИК:
викладач кафедри ІТ Богдан ПИЛИПІВ

ЗАТВЕРДЖЕНО:

на засіданні кафедри ІТ
протокол № 1 від 28.08 2025 р.
PhD, заступник завідувача кафедри Олександр ІВАНОВ

УЗГОДЖЕНО:

Гарант ОП
к.ф-м.н., доцент кафедри Володимир МАКОВИШИН

СХВАЛЕНО:

на засіданні Науково-методичної ради, протокол № 1 від 29.08 2025 р.

e-mail	bohdan.r.pylypiv@ukd.edu.ua
Номер аудиторії чи кафедри	206
Посилання на сайт	https://ukd.edu.ua
Сторінка курсу в СДО	https://online.ukd.edu.ua

ВСТУП

Дисципліна: Основи програмування.

Анотація дисципліни:
Курс охоплює фундаментальні концепції та методи програмування, зосереджуючись на розробці алгоритмів, структурах даних та їх реалізації мовою програмування Python. Значна увага приділяється практичному застосуванню отриманих знань для розв'язання різноманітних задач. Окрім базових аспектів програмування, дисципліна також включає ознайомлення з основами веброзробки (HTML, CSS, JavaScript), що дозволяє студентам побачити практичне поєднання бекенд- та фронтенд-складових. Дисципліна є базовою для подальшого вивчення більш складних предметів у галузі інформаційних технологій та забезпечує необхідні навички для розробки ефективного та якісного програмного забезпечення. Актуальність курсу зумовлена зростаючою потребою у фахівцях, які володіють знаннями та вміннями в області програмування для успішної реалізації інноваційних проєктів.

Мета та завдання навчальної дисципліни:
Сформувати у студентів базові знання та практичні навички в області програмування та веброзробки, необхідні для розробки та аналізу алгоритмів, використання основних структур даних, написання та тестування програмного коду, а також створення простих інтерактивних вебсторінок.

В результаті вивчення дисципліни студент повинен знати:
- Основні парадигми програмування;
- Базові типи даних та структури даних;
- Синтаксис та семантику мови програмування Python;
- Принципи об’єктно-орієнтованого програмування;
- Основи веброзробки (HTML, CSS, JavaScript);
- Принципи розробки та тестування програмного забезпечення.

В результаті вивчення дисципліни студент повинен вміти:
- Розробляти та реалізовувати алгоритми для вирішення поставлених задач;
- Використовувати базові структури даних для організації інформації;

- Писати та налагоджувати програми на мові Python;
- Створювати прості вебсторінки та надавати їм інтерактивність за допомогою HTML, CSS і JavaScript;
- Тестувати та документувати програмний код.

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

Шифр та назва компетентності	**Шифр та назва програмних результатів навчання**
К03 Здатність спілкуватися державною мовою як усно, так і письмово	ПР07 Знати і застосовувати на практиці фундаментальні концепції,

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, К16 Здатність формулювати та забезпечувати вимоги щодо якості програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами, К18 Здатність аналізувати, вибирати і застосовувати методи і засоби для забезпечення інформаційної безпеки (в тому числі кібербезпеки), К19 Володіння знаннями про інформаційні моделі даних та системи, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних, К20 Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв'язання завдань інженерії програмного забезпечення, К22 Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження та визнання важливості навчання протягом всього життя, К23 Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення, К24 Здатність здійснювати процес інтеграції системи, застосовувати стандарти і процедури управління змінами для підтримки цілісності, загальної функціональності і надійності програмного забезпечення, К25 Здатність обгрунтовано обирати та	парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії 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

освоювати інструментарій з розробки та супроводження програмного забезпечення, К26 Здатність до алгоритмічного та логічного мислення	

ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Курс	**1**		
Семестр	**1, 2**		
Кількість кредитів ECTS	**9**		
Аудиторні навчальні заняття		**денна форма**	**заочна форма**
	лекції	**0**	
	практичні	**126**	**24**
Самостійна робота (в годинах)		**144**	**246**
Форма підсумкового контролю	**Залік, Екзамен**		

Структурно-логічна схема вивчення навчальної дисципліни:

Пререквізити	**Постреквізити**
	Інженерія програмного забезпечення

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Зміст навчальної дисципліни

1 Семестр
Тема 1. Вступ до програмування (2 год) 1. Поняття мови програмування. Компіляція та інтерпретація. Огляд популярних мов програмування. 2. Огляд мови програмування Python. Історія мови програмування. Галузі її використання. 3. Встановлення мови програмування на комп'ютер. Ознайомлення з робочим середовищем розробки (PyCharm, Visual Studio Code). **Самостійне вивчення (1 год):** - Поняття алгоритму та його властивості; - Основні парадигми програмування; - Встановлення та налаштування середовища розробки.
Тема 2. Базові структури даних та оператори (4 год) 1. Поняття типізації. Види типізації в різних мовах програмування. Типізація в мові Python. 2. Вивчення основних типів даних: цілі числа, числа з плаваючою комою, рядки, булеві значення. 3. Розгляд операторів: арифметичні, логічні, порівняння, присвоєння. 4. Кастинг змінних. Методи визначення типу змінних 5. Зчитування клавіатури **Самостійне вивчення (2 год):** - Типи даних та їхні особливості; - Пріоритет операторів; - Перетворення типів даних.
Тема 3. Особливості роботи з рядками (4 год) 1. Види форматування виводу 2. Тип змінних рядок (str). Особливості рядків 3. Індексація рядків в python 4. Методи властиві над рядками

Самостійне вивчення (2 год):
- Методи, доступні для роботи з рядками;
- Використання f-strings для форматування;
- Практика роботи з індексацією та зрізами.

Тема 4. Умовні оператори (6 год)

1. Поняття умови в мовах програмування. Види умовних операторів
2. Умовний оператор **if-elif-else**
3. Вкладені умови та множинний вибір
4. Тернарні оператори
5. Умовний оператор **case.** Різниця між if та case

Самостійне вивчення (2 год):
- Синтаксис та семантика умовних операторів;
- Приклади вкладених умов;
- Використання умов для перевірки користувацького вводу.

Тема 5. Колекційні типи даних. Масиви (6 год)

1. Ознайомлення з основними колекційними структурами в мовах програмування.
2. Списки (list): створення, індексація, зрізи, базові операції.
3. Кортежі (tuple): особливості незмінності, використання у програмах.
4. Множини (set): унікальність елементів, операції над множинами.
5. Словники (dict): ключі, значення, методи роботи.
6. Масиви в середині інших масивів. Вкладені масиви

Самостійне вивчення (3 год):
- Методи списків та приклади їх використання;
- Відмінності між списками та кортежами;
- Методи словників і множин;
- Приклади використання генераторів списків.

Тема 6. Цикли(6 год)

1. Повторення дій в мовах програмування
2. Оператор циклу **while**. Особливості та сфери застосування
3. Оператор циклу **for**. Особливості та сфери застосування
4. Керування виконання циклів. Методи примусової зупинки циклів.
5. Вкладені цикли

Самостійне вивчення (3 год):
- Порівняння циклів for та while;
- Приклади використання вкладених циклів;
- Практичні завдання з обробки послідовностей.

Тема 7. Імпорти та робота з модулями (2 год)

1. Імпорт вбудованих модулів.
2. Імпорт користувацьких модулів. Використання ключових слів при імпорті модулів
3. Правильна організація директорій в середовищі програмування. Структура проєкту
4. Відносні та прямі імпорти

Тема 8. Введення в поняття функцій (8 год)

1. Визначення та виклик функцій.
2. Аргументи та параметри функцій.
3. Значення за замовчуванням.
4. Повернення результатів.
5. Область видимості змінних.
6. Рекурсія та її застосування.

Самостійне вивчення (4 год):
- Способи передачі параметрів у функцію;
- Рекурсивні функції;
- Глобальні та локальні змінні.

Тема 9. Робота з файлами (4 год)

1. Вивчення файлових операцій
2. Запис у файл
3. Режими роботи з файлами (r, w, a, b).
4. Робота з текстовими та бінарними файлами.
5. Контекстний менеджер
6. Обробка помилок при роботі з файлами.

Самостійне вивчення (2 год):
- Порівняння режимів відкриття файлів;
- Практика роботи з бінарними файлами;
- Читання великих файлів частинами;

2 Семестр

Тема 10. Обробка винятків. try-catch (4 год)

1. Призначення вийнятків в програмуванні. Необхідність їх обробки
2. Механізм обробки винятків try-except.
3. Типи винятків.
4. Використання **else** та **finally**
5. Генерація власних винятків.

Самостійне вивчення (2 год):

- Ієрархія винятків;
- Конструкція try-finally;
- Створення власних класів винятків.

Тема 11. Розширена робота над функціями. Декоратори (6 год)

1. Анонімна функція **lambda**
2. Функції як об’єкти: передача у змінні та параметри.
3. Вкладені функції.
4. Замикання (closures).
5. Принцип роботи декораторів.
6. Створення власних декораторів та використання вбудованих.

Самостійне вивчення (3 год):

- Практика створення вкладених функцій;
- Написання власних декораторів;
- Ознайомлення з готовими декораторами Python

Тема 12. Система контролю версій та її використання. (12 год)

1. Поняття контролю версій.
2. Огляд популярних систем (Git, SVN).
3. Встановлення та налаштування Git.
4. Основні команди Git. Використання **.gitigrone**
5. Робота з відгалуженнями.
6. Вирішення конфліктів злиття.
7. Використання віддалених репозиторіїв.
8. Робота в команді над одним репозиторієм. Клонування та fork.

Самостійне вивчення (6 год):

- Створення та налаштування власного репозиторію на GitHub;
- Використання .gitignore;
- Практика роботи з гілками;
- Розв’язання конфліктів при злитті;
- Використання pull request у командній розробці.

Тема 13. Ознайомлення із поняттям класів (8 год)

1. Вступ до об’єктно орієнтованого програмування
2. Поняття класу та об’єкта.
3. Визначення класів у Python.
4. Конструктор __init__.
5. Атрибути та методи класу.
6. Створення об’єктів і робота з ними.
7. Приклади практичного застосування класів.

Самостійне вивчення (4 год):

- Відмінність між класом і об’єктом;
- Використання методу __str__ для представлення об’єктів;
- Практика написання простих класів;
- Створення кількох об’єктів одного класу.

Тема 14. Поглиблення в розуміння класів (8 год)

1. Розширений погляд на класи
2. Інкапсуляція: публічні та приватні атрибути.
3. Наслідування: базові та похідні класи.
4. Перевизначення методів.
5. Використання super()
6. Поліморфізм.
7. Спеціальні методи (__len__, __getitem__, __iter__ тощо).

Самостійне вивчення (4 год):

- Практика наслідування та перевизначення методів;
- Створення ієрархії класів;
- Використання super() для доступу до методів батьківського класу;
- Створення власних спеціальних методів.

Тема 15. Ітератори та генератори (4 год)

1. Ознайомлення з механізмами роботи послідовностей у Python

2. Поняття ітератора та протокол ітерації.
3. Функції **iter()** та **next()**.
4. Створення власних ітераторів.
5. Генератори: принцип роботи та створення за допомогою ключового слова **yield**.
6. Генераторні вирази та їх відмінність від спискових включень.

Самостійне вивчення (2 год):
- Створення власного ітератора для обробки даних;
- Практичні приклади генераторів;
- Використання генераторних виразів;
- Застосування генераторів для роботи з великими даними.

Тема 16. Регулярні вирази (6 год)

1. Покращений пошук та фільтрація. Вивчення синтаксису та можливостей регулярних виразів
2. Бібліотека **re.** Основні функції та способи використання
3. Квантифікатори, групи та класи символів.
4. Використання метасимволів
5. Приклади використання регулярних виразів у реальних випадках

Самостійне вивчення (2 год):
- Створення регулярних виразів для перевірки email та телефонів;
- Використання груп у регулярних виразах;
- Практика заміни підрядків за допомогою sub;
- Ознайомлення з "жадібними" та "нежадібними" виразами.

Тема 17. Вступ у Frontend розробку. HTML (8 год)

1. Ознайомлення з основами веброзробки
2. Поняття вебсторінки та браузера.
3. Основи HTML: структура документа.
4. Основні теги: заголовки, параграфи, списки, посилання, зображення.
5. Таблиці та форми.
6. Семантичні теги

Самостійне вивчення (4 год):
- Практика створення простої HTML-сторінки;

- Використання таблиць і форм;
- Створення навігаційного меню;
- Ознайомлення з атрибутами тегів (id, class).

Тема 18. Вступ у Frontend розробку. CSS (8 год)

1. Вивчення основ каскадних таблиць стилів
2. Поняття селекторів і властивостей.
3. Підключення CSS до HTML (inline, internal, external).
4. Основні властивості: шрифти, кольори, відступи, рамки.
5. Робота з блоками та контейнерами.
6. Flexbox та Grid — базове ознайомлення.

Самостійне вивчення (4 год):

- Практика застосування різних видів селекторів;
- Створення стилізованої HTML-сторінки;
- Використання Flexbox для створення адаптивного макета;
- Ознайомлення з CSS-анімаціями.

Тема 19. Основи інтегрованої фронтенд-розробки: HTML, CSS та JavaScript (14 год)

1. Роль **JavaScript** у створенні динамічних вебсторінок.
2. Підключення **JavaScript** до **HTML** (inline, external, script tag).
3. Змінні та типи даних у **JavaScript**.
4. Оператори та вирази.
5. Функції та події (**events**).
6. Робота з **DOM**: отримання елементів, зміна вмісту та стилів.
7. Маніпуляції з атрибутами та класами елементів.
8. Створення та видалення елементів у **DOM**.
9. Робота з циклами та умовами в **JS** для управління логікою сторінки.
10. Взаємодія між **CSS** і **JavaScript** (динамічна зміна стилів).
11. Обробка подій: кліки, наведення, введення з клавіатури.
12. Валідація форм за допомогою **JavaScript**.
13. Основи роботи з локальним сховищем (localStorage, sessionStorage).

Самостійне вивчення (6 год):

- Створення інтерактивної вебсторінки з HTML, CSS та JS;

- Реалізація валідації форм;
- Практика використання DOM для зміни контенту;
- Робота з локальним сховищем;

Тема 20. Тестування програм (6 год)

1. Ознайомлення з способами перевірки правильності роботи програмного забезпечення.
2. Поняття тестування та його види.
3. Тестування через **print().**
4. Використання **debug** функцій середовища.
5. **Unit** тестування.
6. Основи тестування функцій та класів.
7. Практика створення простих тестових кейсів.

Самостійне вивчення (3 год):

- Написання власних модульних тестів;
- Використання pytest для розширених можливостей тестування;
- Ознайомлення з поняттям тестового покриття (coverage);
- Практика побудови тестових сценаріїв для невеликих програм.

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.:	144	246
Опрацювання матеріалу, викладеного на лекціях	23	39
Підготовка до практичних занять та контрольних заходів	36	62
Підготовка звітів з практичних робіт	23	39
Підготовка до поточного контролю	12	20
Опрацювання матеріалу, винесеного на самостійне вивчення	50	86

ПОЛІТИКА КУРСУ

Коротко, з покликанням на відповідну нормативну базу УКД, висвітлити питання:[1]

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 покликанням.

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 покликанням.

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,***

[1] зміст пунктів може редагуватись з огляду на особливості курсу

семінарські та лабораторні заняття, на яких він був відсутній. Ознайомитись з документом можна за покликанням.

4) щодо дотримання академічної доброчесності

"Положення про академічну доброчесність" 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

"Положення про академічну доброчесність" визначає політику щодо використання технічних засобів на основі штучного інтелекту в освітньому процесі. Ознайомитись з документом можна за покликанням.[2] *"Положення про систему запобігання та виявлення академічного плагіату, самоплагіату, фабрикації та фальсифікації академічних творів" 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, окрім виробничої необхідності. Під час виконання заходів контролю використання гаджетів заборонено (за винятком, коли це передбачено умовами його проведення). У разі порушення цієї заборони результат анульовується без права перескладання.

Комунікація відбувається через електронну пошту і сторінку дисципліни в курсі СДО.

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

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

[2] визначається політика використання ШІ в навчальній дисципліні - дозволене/заборонене, правила використання

результатів навчання, здобутих шляхом неформальної та / або інформальної освіти». Ознайомитись з документом можна за покликанням.[3]

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[4]	**Метод навчання**	**Метод оцінювання**
ПР07 Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії 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	лекція, практичні роботи, практичні роботи, кейс-метод, робота під керівництвом викладача лекція, практичні роботи, порівняння, узагальнення, кейс-метод лекція, практичні роботи, порівняння, узагальнення, кейс-метод	іспит, письмовий контроль, тестовий контроль іспит, письмовий контроль, тестовий контроль іспит, письмовий контроль, тестовий контроль

[3] визначається перелік електронних та інших ресурсів та умови перезарахування

[4] для вибіркових навчальних дисциплін вказується результат навчання

ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

Вид	Зміст	% від загальної оцінки	Бал	
			min	max
Поточні контрольні заходи	всього	60	35	60
Підсумкові контрольні заходи	екзамен	40	24	40
Всього:	-	100	60	100

Процедура проведення контрольних заходів, а саме поточного контролю знань протягом семестру та підсумкового семестрового контролю, 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” на підставі чотирибальної шкали – “2”; “3”; “4”; “5”. У разі відсутності студента на занятті виставляється “н”. За результатами поточного контролю у Журналі, автоматично визначається підсумкова оцінка, здійснюється підрахунок пропущених занять.

Усі пропущені заняття, а також негативні оцінки студенти зобов'язані відпрацювати впродовж трьох наступних тижнів. У випадку недотримання цієї норми, замість “н” в журналі буде виставлено “0” (нуль балів), без права перездачі. Відпрацьоване лекційне заняття в електронному журналі позначається літерою «в».[5]

[5] можна вказати теми чи завдання, які є обов’язковими до виконання, а також особисті підходи до оцінювання рівня знань здобувачів під час аудиторної роботи

Критерії оцінювання:

«незадовільно»	Студент володіє матеріалом лише на рівні розпізнавання і відтворення окремих фактів, елементів та об'єктів, що виражаються окремими словами чи реченнями; володіння матеріалом обмежується елементарним рівнем засвоєння, викладення уривається речення; здатний висловити думку на елементарному рівні; володіє матеріалом на рівні окремих фрагментів, що становлять незначну частину навчального матеріалу; не може розпізнати або відтворити матеріал практичних.
«задовільно»	студент володіє матеріалом на початковому рівні, значну частину матеріалу відтворює на репродуктивному рівні; володіє матеріалом на рівні вищому за початковий, за допомогою викладача може логічно відтворити значну його частину; може відтворити значну частину теоретичного матеріалу, виявляючи знання і розуміння основних положень; за допомогою викладача може аналізувати навчальний матеріал, порівнювати, робити висновки та виправляти помилки; може розпізнати або відтворити за прикладом матеріал практичних занять.
«добре»	Студент здатний застосовувати вивчений матеріал у стандартних ситуаціях, частково контролювати власні навчальні дії та наводити окремі власні приклади на підтвердження певних тверджень; вміє порівнювати, узагальненювати та систематизувати інформацію під керівництвом викладача, в чілому самостійно застосовувати її на практиці; контролює власну діяльність, виправляє помилки та добирає аргументи на підтвердження певних думок під керівництвом викладача; вільно володіє вивченим обсягом матеріалу та вміє застосовувати його на практиці; вільно розв'язує задачі в стандартних ситуаціях, самостійно виправляє помилки та добирає переконливі аргументи на підтвердження вивченого матеріалу.
«відмінно»	студент виявляє початкові творчі здібності, самостійно визначає окремі цілі власної навчальної діяльності та оцінює нові факти, явища і ідеї; знаходить джерела інформації та самостійно використовує їх відповідно до цілей, поставлених викладачем; здатний вільно дискутувати на теми, пов'язані з матеріалом навчальної дисципліни, висловлювати власні думки та визначати програму особистої діяльності; самостійно оцінює різноманітні явища і факти, виявляючи особисту позицію щодо них; без

	допомоги викладача знаходить джерела інформації та використовує одержані відомості відповідно до мети й завдань власної пізнавальної діяльності; використовує набуті знання і вміння в нестандартних ситуаціях, виявляє вміння знаходити альтернативні шляхи для вирішення завдань та здобути нові знання самостійно.

До підсумкового контролю допускаються студенти які за результатами поточного контролю отримали не менше 35 балів. Усі студенти, що отримали 34 балів і менше, не допускаються до складання підсумкового контролю і на підставі укладання додаткового договору, здійснюють повторне вивчення дисципліни впродовж наступного навчального семестру. За результатами підсумкового контролю (екзамен) студент може отримати 40 балів. Студенти, які під час підсумкового контролю отримали 24 бали і менше, вважаються такими, що не здали екзамен і повинні йти на перездачу.

Загальна семестрова оцінка з дисципліни, яка виставляється в екзаменаційних відомостях оцінюється в балах (згідно з **Шкалою оцінювання знань за ЄКТС**) і є сумою балів отриманих під час поточного та підсумкового контролю.

Підсумковий контроль з дисципліни Основи програмування проводиться у вигляді Залік, Екзамен.

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 екзамени без поважних причин, вважаються такими, що одержали незадовільну оцінку.

РЕКОМЕНДОВАНІ ДЖЕРЕЛА ІНФОРМАЦІЇ

Основна література:

1. Бублик В. В. *Об’єктно-орієнтоване програмування*. Київ: АртЕк, 2022. 320 с.
2. Васильєв О. Програмування мовою Python. Київ: НаУКМА, 2019. 504 с.
3. Руденко В. Д., Жугастров О. О. *Основи алгоритмізації і програмування мовою Python*. Харків: Ранок, 2021. 288 с
3. Sweigart A. *Automate the Boring Stuff with Python*. 2nd ed. No Starch Press, 2019. 592 p.
4. Zelle J. *Python Programming: An Introduction to Computer Science*. 3rd ed. Franklin, Beedle & Associates, 2016. 552 p.
5. McKinney W. *Python for Data Analysis*. 3rd ed. O’Reilly Media, 2022. 544 p.
6. Beazley D., Jones B. *Python Cookbook*. 3rd ed. O’Reilly Media, 2013. 706 p.
7. Freeman E. *Head First HTML and CSS.* O’Reilly Media, 2012. *736 p.*
6. Duckett J. *HTML & CSS: Design and Build Websites*. Wiley, 2014. 512 p.

Додаткова література:

1. Pilgrim M. *Dive Into Python 3*. Apress, 2009.
2. Downey A. B. *Think Python: How to Think Like a Computer Scientist*. 2nd ed. O’Reilly Media, 2015. 300 p.
3. M. Haverbeke. *Eloquent JavaScript*. 4th ed. No Starch Press, 2024. 480 p.
4. Sedgewick R., Wayne K. *Algorithms in Python*. Addison-Wesley, 2023. 970 p.
5. Grinberg M. *Flask Web Development: Developing Web Applications with Python*. 2nd ed. O’Reilly Media, 2018. 320 p.

Електронні джерела:

1. GeeksforGeeks. Your All-in-One Learning Portal. URL: https://www.geeksforgeeks.org/ (дата звернення: 19.09.2025).
2. The Python Tutorial. URL: https://docs.python.org/3/tutorial/ (дата звернення: 19.09.2025).
3. Real Python. URL: https://realpython.com/.
4. MDN Web Docs (HTML, CSS, JavaScript). URL: https://developer.mozilla.org/ (дата звернення: 19.09.2025).
5. Курс «Основи програмування» на платформі Prometheus. URL: https://prometheus.org.ua/courses/programming-basics/ (дата звернення: 19.09.2025).
6. FreeCodeCamp. URL: https://www.freecodecamp.org/ (дата звернення: 19.09.2025).
7. W3Schools Tutorials. URL: https://www.w3schools.com/ (дата звернення: 19.09.2025).
8. W3Schools Python Tutorial. URL: https://www.w3schools.com/python/ (дата звернення: 19.09.2025).