КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

БУДІВНИЦТВА І АРХІТЕКТУРИ

БАКАЛАВР
(освітній ступінь)

Кафедра ***інформаційних технологій***

«Затверджую»
Гарант ОПП факультету автоматизації і інформаційних технологій
_____ /Олександр ТЕРЕНТЬЄВ/
«___» _____ 2024 року

РОБОЧА ПРОГРАМА ОСВІТНЬОЇ КОМПОНЕНТИ
ОК 12 «ПРОГРАМУВАННЯ МОВОЮ PYTHON»
(шифр та назва освітньої компоненти)

Шифр	назва спеціальності, освітньої програми
122	***Комп'ютерні науки, "Інформаційні управляючі системи і технології"***

Мова викладання: українська

Розробники:
Владислав ГОЦ, кандидат технічних наук, доцент _____
(ім'я та прізвище, науковий ступінь, звання) (підпис)

Сергій ДОЛГОПОЛОВ, асистент кафедри ІТ _____
(ім'я та прізвище, науковий ступінь, звання) (підпис)

Робоча програма затверджена на засіданні кафедри інформаційних технологій Протокол № ***13*** від ***«25» червня 2024 року***

Завідувачка кафедри ІТ _____ Тетяна ГОНЧАРЕНКО
(підпис)

Схвалено гарантом освітньої програми «***Інформаційні управляючі системи і технології***»

Гарант ОП _____ Олександр ПОПЛАВСЬКИЙ
(підпис)

Розглянуто на засіданні науково-методичної комісії спеціальності ***122 «Комп'ютерні науки».***
Протокол № ***3*** від ***«28» червня 2024*** року.

ВИТЯГ З РОБОЧОГО НАВЧАЛЬНОГО ПЛАНУ НА 2024-2025 НАВЧАЛЬНИЙ РІК

Шифр	Назва спеціальності, освітньої програми	Форма здобуття ВО: **денна**											Форма контролю	Семестр	Погодження заступником декана факультету
		Кількість кредитів	Кількість годин						Кількість індивідуальних робіт						
			Всього	Аудиторних				Самостійна робота	КП	КР	РГР	Конт. роб.			
				разом	у тому числі										
					Лекції	Лабораторні	Практичні								
122	Комп'ютерні науки, "Інформаційні управляючі системи і технології"	***3,0***	***90***	***50***	***20***	***30***		***40***			***1***		***Залік***	***3***	

Анотація. Мета та завдання освітньої компоненти

Пререквізити: Програмування та алгоритмічні мови; Офісні інформаційні технології; Інструментальні засоби програмування.

Посилання на сторінку електронного навчально-методичного комплексу освітньої компоненти: https://org2.knuba.edu.ua/course/view.php?id=3621

Мета освітньої компоненти – надання здобувачам фундаментальних знань та практичних навичок з програмування на мові Python, що є однією з найпопулярніших та універсальних мов програмування. Освітня компонента спрямована на розвиток у здобувачів вміння створювати ефективні алгоритми, писати чистий і структурований код, а також вирішувати складні задачі у різних галузях застосування Python.

Завдання освітньої компоненти:

1) Ознайомлення з основами програмування на Python: вивчення синтаксису та основних конструкцій мови; розуміння принципів роботи з різними типами даних (числа, рядки, списки, кортежі, словники, множини); вивчення основних операторів та управляючих конструкцій (умовні оператори, цикли).
2) Розвиток алгоритмічного мислення: формування вмінь розробки та реалізації алгоритмів для вирішення різних типів задач; ознайомлення з методами сортування та пошуку, роботою з рекурсією.
3) Опанування об’єктно-орієнтованого програмування (ООП): вивчення основних концепцій ООП (класи, об’єкти, спадкування, інкапсуляція, поліморфізм); практичне застосування ООП для розробки програмних систем.
4) Вивчення роботи з файлами та базами даних: ознайомлення з методами роботи з файлами (зчитування, запис, обробка); основи роботи з базами даних.
5) Засвоєння принципів роботи з популярними бібліотеками та фреймворками: використання бібліотек для обробки даних (pandas, numpy); ознайомлення з інструментами для візуалізації даних (matplotlib, seaborn); основи веб-розробки з використанням Flask або Django.
6) Розвиток навичок тестування та налагодження коду: вивчення методів тестування програм (unittest, pytest); ознайомлення з техніками налагодження та оптимізації коду.
7) Розробка повноцінних проєктів: застосування отриманих знань для розробки реальних програмних продуктів; робота над груповими проєктами для розвитку навичок командної роботи.
8) Підготовка до професійної діяльності: ознайомлення з практичними аспектами професії програміста; розвиток навичок написання технічної документації та підготовка до технічних інтерв’ю.

Вивчення освітньої компоненти «Програмування мовою Python» сприяє формуванню у здобувачів **наступних компетентностей**.

Компетентності здобувачів освітньої програми, що формуються в результаті засвоєння освітньої компоненти

Код	Зміст компетентності
Інтегральна компетентність	
ІК	Здатність розв'язувати складні спеціалізовані задачі та практичні проблеми у галузі комп'ютерних наук або у процесі навчання, що передбачає застосування теорій та методів інформаційних технологій і характеризується комплексністю та невизначеністю умов.
Загальні компетентності	
ЗК 1	Здатність до абстрактного мислення, аналізу та синтезу.
ЗК 2	Здатність застосовувати знання у практичних ситуаціях.
ЗК 3	Знання та розуміння предметної області та розуміння професійної діяльності.
ЗК 6	Здатність вчитися й оволодівати сучасними знаннями.
ЗК 7	Здатність до пошуку, оброблення та аналізу інформації з різних джерел.
ЗК 8	Здатність генерувати нові ідеї (креативність).
ЗК 9	Здатність працювати в команді.
ЗК 12	Здатність оцінювати та забезпечувати якість виконуваних робіт.
Спеціальні (фахові, предметні) компетентності	
СК 1	Здатність до математичного формулювання та досліджування неперервних та дискретних математичних моделей, обґрунтовування вибору методів і підходів для розв'язування теоретичних і прикладних задач у галузі комп'ютерних наук, аналізу та інтерпретування.
СК 3	Здатність до логічного мислення, побудови логічних висновків, використання формальних мов і моделей алгоритмічних обчислень, проектування, розроблення й аналізу алгоритмів, оцінювання їх ефективності та складності, розв'язності та нерозв'язності алгоритмічних проблем для адекватного моделювання предметних областей і створення програмних та інформаційних систем.
СК 5	Здатність здійснювати формалізований опис задач дослідження операцій в організаційно-технічних і соціально-економічних системах різного призначення, визначати їх оптимальні розв'язки, будувати моделі оптимального управління з урахуванням змін економічної ситуації, оптимізувати процеси управління в системах різного призначення та рівня ієрархії.
СК 6	Здатність до системного мислення, застосування методології системного аналізу для дослідження складних проблем різної природи, методів формалізації та розв'язування системних задач, що мають суперечливі цілі, невизначеності та ризики.
СК 8	Здатність проєктувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління.

СК 9	Здатність реалізувати багаторівневу обчислювальну модель на основі архітектури клієнт-сервер, включаючи бази даних, знань і сховища даних, виконувати розподілену обробку великих наборів даних на кластерах стандартних серверів для забезпечення обчислювальних потреб користувачів, у тому числі на хмарних сервісах.
СК 10	Здатність застосовувати методології, технології та інструментальні засоби для управління процесами життєвого циклу інформаційних і програмних систем, продуктів і сервісів інформаційних технологій відповідно до вимог замовника.
СК 11	Здатність до інтелектуального аналізу даних на основі методів обчислювального інтелекту включно з великими та погано структурованими даними, їхньої оперативної обробки та візуалізації результатів аналізу в процесі розв'язування прикладних задач.

Це забезпечує досягнення ***програмних результатів навчання***, згідно з якими **Здобувач повинен мати знання з питань**:

Програмні результати здобувачів освітньої програми, що формуються в результаті засвоєння освітньої компоненти

Код	**Програмні результати**
ПР 1	Застосовувати знання основних форм і законів абстрактно-логічного мислення, основ методології наукового пізнання, форм і методів вилучення, аналізу, обробки та синтезу інформації в предметній області комп'ютерних наук
ПР 5	Проектувати, розробляти та аналізувати алгоритми розв'язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей алгоритмів та обчислюваних функцій.
ПР 6	Використовувати методи чисельного диференціювання та інтегрування функцій, розв'язання звичайних диференціальних та інтегральних рівнянь, особливостей чисельних методів та можливостей їх адаптації до інженерних задач, мати навички програмної реалізації чисельних методів.
ПР 10	Використовувати інструментальні засоби розробки клієнт-серверних застосувань, проектувати концептуальні, логічні та фізичні моделі баз даних, розробляти та оптимізувати запити до них, створювати розподілені бази даних, сховища та вітрини даних, бази знань, у тому числі на хмарних сервісах, із застосуванням мов веб-програмування.
ПР 11	Володіти навичками управління життєвим циклом програмного забезпечення, продуктів і сервісів інформаційних технологій відповідно до вимог і обмежень замовника, вміти розробляти проєктну документацію (техніко-економічне обґрунтування, технічне завдання, бізнес-план, угоду, договір, контракт).
ПР 12	Застосовувати методи та алгоритми обчислювального інтелекту та інтелектуального аналізу даних в задачах класифікації, прогнозування, кластерного аналізу, пошуку асоціативних правил з використанням програмних інструментів підтримки багатовимірного аналізу даних на основі технологій Data Mining, Text Mining, Web Mining.
ПР 16	Розуміти концепцію інформаційної безпеки, принципи безпечного проєктування програмного забезпечення, забезпечувати безпеку комп'ютерних мереж в умовах неповноти та невизначеності вихідних даних.

ЗМІСТ КУРСУ

Змістовий модуль 1. Основи програмування на Python

Лекція 1. Вступ до програмування на Python. Розглядається історія розвитку мови програмування *Python*, її основні принципи та філософія. Окреслюються основні особливості мови, такі як простота синтаксису та висока читабельність коду. Встановлення та налаштування середовища розробки (*IDE*), а також введення в основні синтаксичні конструкції *Python*, типи даних та оператори. Детально вивчаються умовні оператори (*if*, *else*, *elif*) та цикли (*for*, *while*).

Лекція 2. Функції та структури даних. Оглядаються методи створення та використання функцій, передачі аргументів та повернення значень. Особлива увага приділяється рекурсії, її застосуванню та реалізації у вирішенні різноманітних задач. Охоплюються основні структури даних, такі як списки, кортежі, словники та множини. Розглядаються операції над цими структурами, методи їх обробки та використання в різних задачах. Аналізуються вбудовані функції та методи Python для ефективної роботи з цими структурами даних.

Лекція 3. Об'єктно-орієнтоване програмування (ООП) в Python. Вивчаються основи об'єктно-орієнтованого програмування в *Python*, включаючи створення та використання класів й об'єктів, принципи інкапсуляції, спадкування й поліморфізму. Практичне застосування ООП в *Python* демонструється на прикладах, що ілюструють розробку програм з використанням цих принципів.

Лекція 4. Модульне програмування та організація коду. Розглядається концепція модулів та пакетів в *Python*. Детально описується використання стандартних модулів *Python*, створення власних модулів та пакетів, а також управління залежностями. Окреслюються методи організації коду у великих проєктах з метою підвищення його читабельності та підтримки.

Змістовий модуль 2. Робота з даними та веб-розробка

Лекція 1. Основи роботи з даними та файлами. Розглядаються методи роботи з текстовими та бінарними файлами в *Python*, що включає операції читання та запису файлів, обробку даних у форматах *TXT*, *CSV* та *JSON*. Вивчаються методи парсингу файлів, обробка великих обсягів даних, а також методи обробки бінарних даних. Аналізуються практичні приклади обробки реальних даних. Розглядаються методи обробки та аналізу даних з використанням бібліотек *pandas* та *numpy*. Охоплюються операції маніпуляції табличними даними, фільтрація, групування, агрегація. Аналізуються методи обробки великих обсягів даних та побудова візуалізацій для їх аналізу.

Лекція 2. Бази даних та веб-розробка. Вивчаються основи роботи з базами даних за допомогою *SQL*. Розглядаються методи підключення до БД, створення та управління БД з використанням бібліотеки *PostgreSQL*. Охоплюються операції вставки, вибірки, оновлення та видалення даних. Вивчаються основи

веб-розробки з використанням фреймворків *Flask* та *Django*. Охоплюються методи створення веб-додатків, налаштування роутингу, обробка *HTTP*-запитів, робота з шаблонами та статичними файлами. Розглядаються методи взаємодії з БД у контексті веб-додатків та забезпечення безпеки.

Лекція 3. Тестування, налагодження та оптимізація коду. Вивчаються методи тестування та налагодження коду в *Python*. Охоплюються основи написання тестів з використанням бібліотек *unittest* та *pytest*. Розглядаються методи налагодження коду, виявлення та виправлення помилок, оптимізація продуктивності програм, що передбачає аналіз кращих практик тестування та налагодження програмного забезпечення. Розглядаються практичні приклади використання даних з реальних джерел та їх тестування.

Змістовий модуль 3. Розширені можливості та практичне застосування Python

Лекція 1. Візуалізація та машинне навчання в Python. В Вивчаються методи візуалізації даних з використанням бібліотек *matplotlib* та *seaborn*. Розглядаються основні типи графіків та діаграм, їх налаштування та стилізація. Охоплюються методи побудови інтерактивних графіків, аналізу даних через візуалізацію. Досліджуються приклади візуалізації великих обсягів даних для прийняття управлінських рішень. Вивчаються основи машинного навчання з використанням *Python*. Оглядаються основні алгоритми та методи машинного навчання, включаючи регресію, класифікацію та кластеризацію. Використовуються бібліотеки *scikit-learn* та *TensorFlow* для побудови моделей. Розглядаються приклади застосування машинного навчання в різних галузях, включаючи аналіз даних та прогнозування.

Лекція 2. Розширені можливості та безпека Python-додатків. Оглядаються розширені можливості *Python*, включаючи асинхронне програмування та багатопоточність. Вивчаються бібліотеки *asyncio* та *threading*, їх використання для створення високопродуктивних додатків. Розглядаються методи управління потоками, асинхронні виклики та обробка одночасних завдань. Аналізуються приклади реальних додатків з асинхронною обробкою даних. Розглядаються основи безпеки в Python-додатках. Вивчаються методи захисту веб-додатків від основних видів атак, таких як *SQL Injection*, *XSS* та *CSRF*. Оглядаються бібліотеки та інструменти для забезпечення безпеки коду та даних. Розглядаються кращі практики безпеки при розробці ПЗ на *Python*.

Лекція 3. DevOps та CI/CD з Python. Вивчаються основи *DevOps* та принципи безперервної інтеграції та розгортання (*CI/CD*) для *Python*-додатків. Розглядаються інструменти автоматизації, такі як *Jenkins*, *GitHub Actions* та *Docker*. Охоплюються методи налаштування *CI/CD* пайплайнів для автоматизації тестування, збірки та розгортання додатків. Аналізуються приклади впровадження *DevOps* у реальних проєктах.

Теми лабораторних робіт

1 модуль **2 модуль** **3 модуль**

№	Назва теми	Формат виконання	Кількість годин	Кількість балів
1.	Встановлення середовища розробки та написання першої програми	 Індивідуально	2	2
2.	Реалізація умовних операторів та циклів		2	2
3.	Створення та використання функцій, реалізація рекурсії		2	2
4.	Операції зі структурами даних		2	2
5.	Створення класів та об'єктів, реалізація ООП	 Команда	2	7
6.	Використання та створення модулів та пакетів	 Індивідуально	2	2
7.	Читання та запис даних у файли		2	2
8.	Підключення до бази даних та виконання основних операцій		2	2
9.	Створення простого веб-додатку на Flask/Django		2	2
10.	Розробка інтерактивних сторінок на Flask/Django	 Команда	2	7
11.	Використання pandas для обробки табличних даних	 Індивідуально	2	2
12.	Візуалізація даних з matplotlib та seaborn		2	2
13.	Написання тестів з використанням unittest або pytest		2	2
14.	Розробка початкової версії продукту	 Команда	2	4
15.	Удосконалення та презентація додатку		2	5
Разом			***30***	***45***

Розподіл годин самостійної роботи здобувачів

№	Назва теми	Кількість годин
	Змістовий модуль 1.	**7**
1	Порівняльний аналіз парадигм програмування в контексті *Python*. Вивчення особливостей імперативного програмування та його реалізації в *Python*. Дослідження функціонального програмування, його принципів та застосування через функції вищого порядку, лямбда-вирази та замикання. Аналіз об'єктно-орієнтованого та логічного програмування, їх основних концепцій та переваг. Порівняння ефективності різних парадигм для вирішення специфічних задач програмування.	2
2	Поглиблене вивчення структур даних *Python* та їх внутрішньої реалізації. Детальний аналіз внутрішньої будови списків, словників, множин та їх оптимізованих версій. Дослідження особливостей реалізації рядків та кортежів. Вивчення методів оптимізації роботи зі структурами даних, включаючи використання генераторів та ітераторів. Порівняльний аналіз швидкодії різних структур даних при виконанні типових операцій.	2,5
3	Розширені концепції об'єктно-орієнтованого програмування в *Python*. Вивчення метакласів та їх застосування для створення гнучких архітектурних рішень. Дослідження паттернів проєктування (*Factory*, *Singleton*, *Observer*, *Strategy*) та їх реалізації в *Python*. Аналіз принципів *SOLID* та їх застосування при розробці об'єктно-орієнтованих систем на *Python*.	2,5
	Змістовий модуль 2.	**7**
4	Вивчення сучасних підходів до роботи з базами даних в *Python*. Поглиблений аналіз *ORM*-систем *SQLAlchemy* та *Django ORM*, їх архітектури та особливостей використання. Дослідження методів оптимізації запитів, включаючи *lazy loading* та *eager loading*. Вивчення механізмів управління транзакціями та забезпечення цілісності даних. Аналіз методів міграції схем баз даних та версіонування структури бази даних.	2
5	Дослідження архітектурних патернів веб-додатків на *Python*. Детальний аналіз архітектури *MVC* та *MTV*, їх реалізації у *Flask* та *Django*. Вивчення механізмів маршрутизації, обробки запитів та відповідей. Дослідження систем шаблонізації та роботи з формами. Аналіз методів аутентифікації та авторизації у веб-додатках.	2,5
6	Вивчення методів оптимізації обробки великих об'ємів даних з використанням *pandas* та *numpy*. Дослідження векторизованих операцій та їх переваг над циклічними обчисленнями. Аналіз методів агрегації та групування даних. Вивчення оптимізації пам'яті при роботі з великими наборами даних. Дослідження методів паралельної обробки даних з використанням багатоядерних систем.	2,5
	Змістовий модуль 3.	**8**
7	Дослідження асинхронного програмування та багатопоточності в	2,5

	Python. Вивчення концепцій асинхронного програмування та *event loop*. Аналіз використання *asyncio* для створення високопродуктивних мережевих додатків. Дослідження багатопоточного програмування з використанням *threading* та *concurrent.futures*. Вивчення механізмів синхронізації потоків та уникнення *race conditions*.	
8	Вивчення принципів машинного навчання на *Python*. Дослідження основних алгоритмів класифікації, регресії та кластеризації. Аналіз методів попередньої обробки даних та вибору ознак. Вивчення методів оцінки якості моделей та їх оптимізації. Практичне застосування бібліотек *scikit-learn* та *TensorFlow* для розв'язання задач машинного навчання.	2,5
9	Дослідження сучасних практик *DevOps* та *CI/CD* для *Python*-проєктів. Вивчення методів автоматизації процесів тестування, збірки та розгортання *Python*-додатків. Аналіз інструментів *CI/CD*, таких як *Jenkins*, *GitHub Actions* та *GitLab CI*. Дослідження практик контейнеризації з використанням *Docker* та оркестрації з *Kubernetes*. Вивчення методів моніторингу та логування *Python*-додатків у промисловій експлуатації.	3
	Виконання РГР за варіантом	12
	Підготовка до заліку	6
	Всього	**40**

Індивідуальні завдання

Розрахунково-графічна робота

Розрахунково-графічна робота (РГР) є важливою складовою навчального процесу, що дозволяє здобувачам закріпити та поглибити теоретичні знання, отримані під час лекційних занять, шляхом практичного застосування їх для вирішення конкретних задач. РГР сприяє розвитку аналітичного мислення, навичок програмування, самостійного пошуку та обробки інформації, а також навичок ефективного представлення результатів своєї діяльності.

Основними цілями виконання РГР є:

- Засвоєння теоретичних знань.
- Розвиток практичних навичок.
- Формування навичок самостійної роботи.
- Оцінка знань та вмінь.

Успішне виконання РГР вимагає від здобувачів вміння аналізувати задачі, розробляти ефективні алгоритми, писати якісний код, проводити тестування та оптимізацію, а також представляти результати своєї роботи у вигляді звіту та презентації. Всі ці етапи є невід'ємною частиною процесу навчання і розвитку компетенцій, необхідних для майбутньої професійної діяльності.

Для забезпечення об'єктивності та прозорості оцінювання РГР вводиться шкала в 100 балів з можливістю отримання додаткових 10 балів за особливі досягнення (35 балів у загальній підсумковій оцінці). Оцінювання здійснюється за

чітко визначеними критеріями, що дозволяють врахувати всі аспекти виконаної роботи, від коректності вирішення завдання до якості коду та документації.

№	Критерій оцінювання	Максимальна кількість балів
1	**Коректність вирішення завдання**	**20**
1.1	Виконання усіх вимог завдання	10
1.2	Логічність і правильність реалізації алгоритму	10
2	**Якість коду**	**20**
2.1	Зрозумілий і читабельний код	7
2.2.	Використання відповідних структур даних та методів	3
2.3	Відсутність надлишкового коду	3
2.4	Коментування коду та дотримання стандартів стилю	4
2.5	Використання функцій та модульність	3
3	**Оптимізація та ефективність**	**12**
3.1	Ефективність обраних алгоритмів	7
3.2	Оптимальне використання ресурсів	5
4	**Документація та презентація результатів**	**13**
4.1	Наявність чіткої та детальної документації	5
4.2	Візуальне представлення результатів (графіки, таблиці, діаграми)	5
4.3	Якість та повнота звіту	3
5	**Інноваційність та креативність**	**10**
5.1	Використання нових або нестандартних підходів до вирішення задачі	5
5.2	Оригінальність презентації	5
6	**Складність завдання**	**10**
	Складність обраного завдання та його відповідність вимогам курсу	10
7	**Тестування та верифікація**	**10**
	Якість та повнота тестування коду	5
	Використання автоматизованих тестів	5
8	**Забезпечення гнучкості та масштабованості**	**5**
	Розширюваність та підтримка змін в коді	5
9	**Додаткові бали (Extra Points)**	**10**
	Використання передових методів або технологій	5
	Успішне впровадження проєкту у реальному середовищі	5
	Загалом	**100 (10 extra)**

Опис критеріїв оцінювання РГР:

Виконання усіх вимог завдання (10 балів) – ваш код повинен повністю відповідати поставленим завданням. Усі аспекти задачі, які були задані викладачем, повинні бути реалізовані у вашій програмі.

Логічність і правильність реалізації алгоритму (10 балів) – ваш алгоритм повинен бути логічним та коректно реалізованим, що передбачає правильний вибір методів і підходів для вирішення задачі, а також відсутність логічних помилок у коді.

Зрозумілий і читабельний код (7 балів) – код повинен бути написаний так, щоб його було легко читати і розуміти. Використовуйте зрозумілі назви змінних, функцій та інших елементів коду.

Використання відповідних структур даних та методів (3 балів) –використовуйте найвідповідніші структури даних та методи для вирішення поставленого завдання – це покращує ефективність та простоту розуміння коду.

Відсутність надлишкового коду (3 балів) – ваш код не повинен містити зайвих рядків чи блоків, які не мають практичного значення для вирішення задачі. Уникайте дублювання коду!

Коментування коду та дотримання стандартів стилю (4 балів) – код повинен бути добре прокоментований, щоб пояснити складні або важливі частини. Дотримуйтеся стандартів стилю програмування Python.

Використання функцій та модульність (3 балів) – розбивайте ваш код на функції та модулі, щоб він був більш організованим та легким для підтримки.

Ефективність обраних алгоритмів (7 балів) – використовуйте алгоритми, які забезпечують ефективне вирішення задачі. Пам’ятайте про складність алгоритмів та намагайтеся мінімізувати час виконання та використання пам’яті.

Оптимальне використання ресурсів (5 балів) – ваша програма повинна ефективно використовувати доступні ресурси комп’ютера, такі як процесорний час та пам’ять.

Наявність чіткої та детальної документації (5 балів) – у вашій документації повинні бути чітко викладені всі аспекти вашої роботи, включаючи опис проблеми, підходи до її вирішення, використані методи та алгоритми.

Візуальне представлення результатів (графіки, таблиці, діаграми) (5 балів) – представляйте результати вашої роботи у вигляді графіків, таблиць чи діаграм, щоб краще проілюструвати отримані дані та їх аналіз.

Якість та повнота звіту (3 балів) – звіт повинен бути повним та добре структурованим, включаючи всі необхідні розділи, що описують виконану роботу та отримані результати.

Використання нових або нестандартних підходів до вирішення задачі (5 балів) – демонструйте креативність у підході до вирішення задачі, використовуючи інноваційні методи або нестандартні підходи.

Оригінальність презентації (5 балів) – ваша презентація повинна бути оригінальною та привабливою, привертаючи увагу до ключових аспектів роботи.

Складність обраного завдання та його відповідність вимогам курсу (10 балів) – оцініть рівень складності вашого завдання та переконайтеся, що воно відповідає вимогам курсу. Складніші завдання заслуговують на вищу оцінку!

Якість та повнота тестування коду (5 балів) – забезпечте ретельне тестування вашого коду для виявлення можливих помилок та перевірки правильності його роботи.

Використання автоматизованих тестів (5 балів) – використовуйте автоматизовані тести для перевірки роботи вашої програми, що покращує її надійність та зручність тестування.

Розширюваність та підтримка змін в коді (5 балів) – ваш код повинен бути гнучким та легко розширюваним, щоб можна було без проблем додавати нові функції або змінювати існуючі.

Використання передових методів або технологій (5 балів) – використовуйте передові методи або технології для вирішення задачі, що демонструє ваше прагнення до новітніх досягнень у сфері програмування.

Успішне впровадження проєкту у реальному середовищі (5 балів) – якщо ваша програма успішно впроваджена та використовується у реальному середовищі, це буде оцінено додатковими балами.

Розподіл годин дозволяє ефективно організувати процес виконання розрахунково-графічної роботи, забезпечуючи належний рівень якості та відповідність встановленим критеріям оцінювання.

№	Етап	Кількість годин
1	**Аналіз завдання та планування**	**1**
1.1	Ознайомлення з вимогами завдання та планування роботи	1
2	**Розробка алгоритму та кодування**	**5**
2.1	Розробка алгоритму	3
2.2	Написання коду та його тестування	2
3	**Оптимізація коду та відладка**	**2**
3.1	Оптимізація алгоритмів та ефективності використання ресурсів	1
3.2	Відладка та усунення помилок	1
4	**Документація та оформлення звіту**	**2**
4.1	Написання технічної документації та звіту	2
5	**Розробка презентації**	**2**
5.1	Створення презентаційного матеріалу для захисту роботи	2
	Загалом	**12**

Здобувачі можуть обрати будь-яку з запропонованих тем для виконання своєї роботи. Якщо у здобувача є інша ідея для проєкту, яка здається більш цікавішою, будь ласка, обговоріть її з викладачем заздалегідь, щоб отримати дозвіл на її реалізацію.

Теми до розрахунково-графічної роботи:

1. Додаток для управління особистими фінансами.
2. Система рекомендацій для книжкового онлайн-магазину.
3. Автоматизована система управління домашніми задачам.
4. Додаток для аналізу даних соціальних мереж.
5. Чат-бот для підтримки клієнтів.
6. Додаток для планування подорожей.
7. Система розпізнавання облич для безпеки.
8. Аналізатор текстів для виявлення тональності.
9. Додаток для управління проєктами.
10. Додаток для моніторингу здоров’я.
11. Система управління інвентарем для малого бізнесу.
12. Додаток для автоматизації навчального процесу.
13. Інтерактивний довідник з Python.
14. Система управління IoT-пристроями.
15. Аналізатор продуктивності розробників.
16. Додаток для відстеження фізичної активності.
17. Система автоматизації розкладу занять.
18. Інтерактивний симулятор інвестицій.
19. Персональний асистент для вивчення мов.
20. Система моніторингу якості повітря.
21. Додаток для ведення нотаток та задач.
22. Платформа для онлайн-опитувань.
23. Система управління контактами.
24. Додаток для управління витратами на паливо.
25. Система управління ресторанними замовленнями.
26. Додаток для аналізу спортивних результатів.
27. Система моніторингу серверів та послуг.
28. Додаток для управління бібліотекою.
29. Система управління подіями та заходами.
30. Додаток для управління витратами на подорожі.
31. Система автоматизації повідомлень.
32. Додаток для створення та управління анкетами.
33. Система управління складом та запасами.
34. Додаток для аналізу фінансових ринків.
35. Система управління проєктами з Gantt діаграмою.
36. Додаток для управління спортзалом.
37. Система аналізу соціальних медіа.

38. Додаток для відстеження навичок та досягнень.
39. Система управління бізнес-процесами.
40. Додаток для аналізу даних клієнтів.
41. Система управління магазином.
42. Додаток для управління витратами на освіту.
43. Система моніторингу та аналізу погоди.
44. Додаток для планування сімейного бюджету.
45. Система управління паркуванням.
46. Додаток для управління рецептами та меню.
47. Система управління витратами на автомобіль.
48. Додаток для аналізу даних про здоров'я.
49. Система управління витратами на електроенергію.
50. Додаток для управління списками покупок.
51. Система управління логістикою.
52. Додаток для аналізу даних про споживання води.
53. Система моніторингу якості води.
54. Додаток для слідкування за домашніми улюбленцями.
55. Система управління витратами на зв'язок.
56. Додаток для аналізу даних про використання мобільного зв'язку.
57. Система управління туристичними подорожами.
58. Додаток для управління витратами на харчування.
59. Система моніторингу якості повітря в реальному часі.
60. Додаток для управління персональними цілями та завданнями.

Методи контролю та оцінювання знань

Загальне оцінювання здійснюється через вимірювання результатів навчання у формі проміжного (модульного) та підсумкового контролю (залік, захист розрахунково-графічної роботи тощо) відповідно до вимог зовнішньої та внутрішньої системи забезпечення якості вищої освіти.

Проміжний контроль проводиться під час навчального семестру для оцінки засвоєння здобувачами конкретних модулів або блоків робочої програми. Проміжний контроль може включати тести, опитування, лабораторні роботи та інші форми оцінювання, що дозволяють визначити рівень поточних знань і навичок здобувачів.

Підсумковий контроль здійснюється в кінці навчального семестру або курсу для підсумкової оцінки знань здобувачів та включає залік, захист розрахунково-графічної роботи, а також інші форми оцінювання, що відповідають вимогам освітньої програми.

Контрольні заходи поділяються на вхідний, поточний, модульний та семестровий контроль.

Вхідний контроль проводиться на початку навчального семестру або курсу для визначення початкового рівня знань здобувачів.

Поточний контроль здійснюється протягом семестру під час лабораторних та індивідуальних занять, забезпечуючи регулярну оцінку прогресу здобувачів.

Модульний контроль проводиться після завершення певного модуля або блоку навчальної програми для оцінки засвоєння конкретного матеріалу.

Семестровий контроль виконується за окремим графіком, складеним деканатом факультету, і включає підсумкову оцінку знань здобувачів за семестр.

Процедури контролю різняться залежно від типу контролю. Вхідний, поточний та модульний контроль проводяться під час лабораторних та індивідуальних занять з викладачем. Методи включають тести, опитування, практичні завдання та інші форми оцінювання. Семестровий контроль здійснюється відповідно до затвердженого графіку і може включати письмові іспити, заліки, усні опитування та інші форми підсумкового оцінювання.

Засоби контролю розрахунково-графічної чи курсової робіт включають їх представлення та захист перед комісією або викладачем. Здобувачі повинні продемонструвати розуміння теми, методології дослідження та вміння застосовувати отримані знання на практиці. Можуть вимагатися детальні письмові звіти про виконану роботу, що містять аналіз результатів, висновки та рекомендації. Також можливе візуальне представлення роботи за допомогою презентаційного матеріалу для демонстрації основних результатів та досягнень.

Додаткові механізми контролю включають використання онлайн-платформ для проведення тестів, завдань та обговорень, оцінку самостійних завдань, які виконуються здобувачами поза аудиторією, включаючи дослідницькі роботи, реферати та інші види діяльності. Регулярні консультації з викладачами сприяють обговоренню прогресу, отриманню зворотного зв’язку та уточненню незрозумілих питань.

Означені методи контролю спрямовані на забезпечення систематичного та об’єктивного оцінювання знань і навичок здобувачів, сприяючи їхньому успішному навчанню та професійному розвитку.

Політика щодо академічної доброчесності

Всі письмові роботи, включаючи індивідуальні завдання, розрахунково-графічну роботу, курсову роботу та презентації, повинні бути оригінальними. Роботи можуть бути перевірені на плагіат, і їх оригінальність повинна складати не менше 70%. Виключення становлять наукові публікації, що вже були перевірені на плагіат і прийняті до публікації у наукових виданнях або конференціях.

Списування (використання сторонніх джерел інформації, мобільних пристроїв або інших технічних засобів без дозволу викладача) під час тестів, іспитів та інших форм оцінювання заборонене. Порушники можуть бути позбавлені можливості продовжувати тестування та підлягати дисциплінарним заходам.

У разі виявлення порушень академічної доброчесності (плагіат, списування тощо), здобувачу буде надано повторне завдання або призначено додаткове заняття

для проходження оцінювання. Повторні порушення можуть призвести до більш серйозних наслідків – не зарахування проходження курсу освітньої компоненти.

Політика щодо відвідування

Навчальний процес з курсу «Програмування мовою Python» організовано з використанням платформи Microsoft Teams, що забезпечує гнучкість у форматі навчання.

Особливості організації навчального процесу:

- Усі лекційні заняття записуються та зберігаються у відповідному каналі Teams протягом семестру.
- Лабораторні роботи представлені в електронному вигляді з докладними інструкціями та прикладами виконання.
- Матеріали курсу (презентації, приклади коду, додаткові ресурси) доступні в Teams.
- Консультації можливі як в очному форматі, так і через Teams.

Виконання та захист лабораторних робіт:

- Лабораторні роботи можуть виконуватися дистанційно.
- Захист робіт можливий протягом усього семестру.
- Передбачено покрокові інструкції та шаблони для виконання робіт.
- Командні проєкти можуть виконуватися розподілено з використанням систем контролю версій.

У разі пропуску занять здобувач має:

- Надати до деканату та продемонструвати викладачу документи, що підтверджують поважність причини пропуску (медичні довідки, документи про участь у конференціях, стажуваннях тощо).
- Переглянути відеозапис пропущеної лекції в Teams.
- Виконати всі практичні завдання, передбачені за темою пропущеного заняття.

Можливість онлайн-навчання надається за таких умов:

- Хвороба (за наявності медичної довідки).
- Участь у міжнародному стажуванні.
- Участь у наукових конференціях.
- Інші об’єктивні обставини за погодженням з керівником курсу.

Визнання результатів неформальної та інформальної освіти

В межах курсу визнаються результати навчання, отримані у неформальній та інформальній освіті, зокрема:

- Сертифікати онлайн-курсів з Python на платформах Coursera, edX, Udacity тощо.

- Сертифікати з програмування від технологічних компаній (наприклад, Python Institute: PCEP-30-0x, PCAP-31-0x, PCPP-32-10x, PCPP-32-20x, PCET-30-0x, PCAT-31-0x, PCED-30-0x, PCAD-31-0x тощо).
- Завершені курси на платформі DataCamp (тощо) за відповідною тематикою.
- Участь у хакатонах та проєктах з використанням Python (за наявності підтверджуючих матеріалів).

Процедура визнання передбачає:
- Подання заяви та підтверджуючих документів (сертифікати, код проєктів, звіти тощо).
- Співбесіду для підтвердження набутих компетентностей.
- Зарахування відповідних тем чи лабораторних робіт за результатами розгляду.

Максимальний обсяг визнаних результатів не може перевищувати 25% від загального обсягу курсу.

Усі навчальні матеріали, включаючи презентації, додаткові ресурси та завдання, доступні здобувачам через систему Teams, що забезпечує безперервність навчального процесу незалежно від форми участі в заняттях.

Методи контролю

Основні форми участі здобувачів у навчальному процесі з курсу «Програмування мовою Python», що підлягають поточному контролю:
- Виконання та захист лабораторних робіт.
- Участь у розробці командних проєктів.
- Написання та відлагодження програмного коду.
- Презентація розроблених програмних рішень.
- Участь у code review та обговореннях архітектурних рішень.
- Виконання самостійних завдань.
- Робота з документацією та технічною літературою.

Кожна тема курсу відпрацьовується здобувачами через практичну реалізацію програмних завдань та захист виконаних робіт. Передбачається регулярна робота з кодом та його оптимізація протягом семестру.

Під час оцінювання рівня знань здобувача аналізу підлягають:
- Характеристики програмного коду: функціональність, ефективність, читабельність, документованість, відповідність стандартам PEP 8.
- Якість засвоєння матеріалу: розуміння принципів роботи Python, вміння застосовувати різні підходи до вирішення задач.
- Здатність поєднувати теоретичні знання з практичною реалізацією.

- Рівень володіння інструментарієм розробника: IDE, системи контролю версій, налагоджувачі.
- Навички проєктування та розробки програмних рішень.
- Самостійна робота з документацією, API та навчальними ресурсами.
- Вміння працювати в команді та комунікувати технічні рішення.

Тестове опитування проводиться за змістовими модулями та охоплює як теоретичні аспекти Python, так і практичні завдання з програмування.

Індивідуальне завдання передбачає виконання розрахунково-графічної роботи (РГР) за індивідуальним варіантом. РГР спрямована на практичне застосування принципів об'єктно-орієнтованого програмування та розробку закінченого програмного продукту.

Поточний контроль включає оцінювання:

- Виконаних лабораторних робіт.
- Якості програмного коду.
- Активності на заняттях.
- Участі в командних проєктах.
- Своєчасності виконання завдань.

Позитивна оцінка поточної успішності за відсутності пропущених та невідпрацьованих лабораторних робіт є підставою для допуску до підсумкового контролю.

Підсумковий контроль здійснюється під час залікової сесії та враховує:

- Результати виконання лабораторних робіт.
- Якість виконання РГР.
- Результати поточного контролю.
- Активність роботи протягом семестру.

Оцінювання проводиться за 100-бальною шкалою.

Розподіл балів для освітньої компоненти

Поточне оцінювання				Залік	Сума балів
Модуль № 1	Модуль № 2	Модуль № 3	Індивідуальне завдання		
15	15	15	35	20	100

Шкала оцінювання: національна та ECTS

Сума балів за всі види навчальної діяльності	Оцінка ECTS	Оцінка за національною шкалою
90-100	**A**	відмінно
82-89	**B**	добре
74-81	**C**	
64-73	**D**	задовільно
60-63	**E**	
35-59	**FX**	Не зараховано з можливістю повторного складання
0-34	**F**	Не зараховано з обов'язковим повторним вивченням дисципліни

Умови допуску до підсумкового контролю

Здобувачу, який має підсумкову оцінку за освітню компоненту від 35 до 59 балів, призначається додаткова залікова сесія. В цьому разі він повинен виконати додаткові завдання, визначені викладачем.

Здобувач, який не виконав вимог робочої програми за змістовними модулями, не допускається до складання підсумкового контролю. В цьому разі він повинен виконати визначене викладачем додаткове завдання за темами відповідних змістових модулів в період між основною та додатковою сесіями.

Здобувач має право на опротестування результатів контроля (апеляцію). Правила подання та розгляду апеляції визначені внутрішніми документами КНУБА, які розміщені на сайті КНУБА та зміст яких доводиться Здобувачам до початку вивчення освітньої компоненти.

Методичне забезпечення освітньої компоненти

Програмне забезпечення:

1. Python (*PSF License Agreement*): посилання для встановлення.
2. Google Colab (*безкоштовна версія*): посилання для використання.
3. PyCharm Community Edition (*Apache License 2.0*): посилання для встановлення; PyCharm Professional (*Trialware 30 днів*): посилання для встановлення.
4. Github (*безкоштовна версія*): посилання для використання.
5. Gitlab (*Trialware 60 днів*): інструкція з встановлення.
6. Docker (*Docker Personal, безкоштовна версія*): посилання для встановлення.
7. Jenkins (*MIT License*): інструкція з встановлення.
8. Notion (*безкоштовна версія*): посилання для використання.
9. Django (*Three-clause BSD license*): інструкція з встановлення.
10. Flask (*Three-clause BSD license*): інструкція з встановлення.
11. PostgreSQL (*The PostgreSQL License*): посилання для використання.
12. SQLAlchemy (*MIT License*): інструкція з встановлення.

Навчальні посібники:

1. А. О. Костюченко. Основи програмування мовою Python : навч. посіб. Чернігів : Нац. унів. «Чернігівський колегіум» ім. Т. Г. Шевченка, 2020. 180 с.
2. Г. Ю. Цибко Г.Ю., Ю. В. Горошко, А. О. Костюченко А. О. Програмування у Python. Практичний курс : навч. посібник. Ч. : ФОП Баликіна С. М., 2022. 183 с.
3. Селіверстов Р., Мельничин А. Основи програмування мовою Python: навч. посібник. – Львів : ЛНУ імені Івана Франка, 2020. – 190 с.

Методичні роботи:

1. Методичні вказівки до виконання лабораторних робіт з освітньої компоненти «Програмування мовою Python» (електронний варіант). Укладач: Долгополов С.Ю., КНУБА, 2024. – 100 с.
2. Методичні вказівки до виконання розрахунково-графічної роботи з освітньої компоненти «Програмування мовою Python» (електронний варіант). Укладач: Долгополов С.Ю., КНУБА, 2024. – 50 с.

Додаткові джерела:

1. Dolhopolov, S., Honcharenko, T., Dolhopolova, S.A., Riabchun, O., Delembovskyi, M., & Omelianenko, O. (2022). Use of Artificial Intelligence Systems for Determining the Career Guidance of Future University Student. 2022 International Conference on Smart Information Systems and Technologies (SIST), 1-6.

2. Dolhopolov, S., Honcharenko, T., Fedusenko, O., Khrolenko, V., Hots, V., & Golenkov, V. (2024). Neural Network Threat Detection Systems for Data Breach Protection. 2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST), 415-421.
3. T. Simon. Python for Beginners. Master Python Programming from Basics to Advanced Level, 2024. 133 p.
4. B. Lubanovic. FastAPI. Modern Python Web Development. O'Reilly Media, INC. 2024. 280 p.
5. A. Chapagain. Hands-On Web Scraping with Python. Extract quality data from the web using effective Python techniques, Packt Publishing Ltd. 2023, 324 p.
6. T. Shake. Python for Advance: 3 Days with Python. 2023, 263 p.
7. Chernyshev, D., Dolhopolov, S., Honcharenko, T., Haman, H., Ivanova, T., & Zinchenko, M. (2022). Integration of Building Information Modeling and Artificial Intelligence Systems to Create a Digital Twin of the Construction Site. 2022 IEEE 17th International Conference on Computer Sciences and Information Technologies (CSIT), 36-39.
8. Fluent Python : Clear, Concise, and Effective Programming. O'Reilly Media, INC. 2022. 1011 p.

Інформаційні ресурси:

1. https://leetcode.com/
2. https://pythoninstitute.org/
3. https://codecombat.com/
4. https://www.hackerrank.com/
5. https://docs.python.org/3/
6. https://pymbook.readthedocs.io/en/latest/
7. https://colab.google/
8. https://www.kaggle.com/
9. https://github.com/
10. http://library.knuba.edu.ua/
11. http://org2.knuba.edu.ua/