МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

Факультет фінансів та цифрових технологій
Кафедра кібернетики та прикладної математики

Затверджено
Науково-методична рада ДПУ
від «19» 04 2024 № 5
Голова НМР Іван ШЕМЕЛИНЕЦЬ

Робоча програма навчальної дисципліни
«Об'єктно-орієнтоване програмування»
для підготовки здобувачів вищої освіти першого (бакалаврського) рівня
(денної форми навчання)
галузь знань 05 «Соціальні та поведінкові науки»
спеціальність 051 «Економіка»

ОПП: «Економічна кібернетика»

Статус дисципліни: обов'язкова

Ірпінь – 2024

Робоча програма навчальної дисципліни «Об'єктно-орієнтоване програмування» складена на основі освітньо-професійної програми «Економічна кібернетика» першого (бакалаврського) освітнього рівня, спеціальності 051 «Економіка», затвердженої Вченою радою Університету 26.04.2021 року, протокол №5

Укладачі: _____ В.В. Лаговський, к.е.н., доцент, зав. кафедри кібернетики та прикладної математики

Гаранти освітніх програм _____ В.В. Лаговський, к.е.н., доцент, зав. кафедри кібернетики та прикладної математики

Робочу програму навчальної дисципліни розглянуто та схвалено кафедрою кібернетики та прикладної математики, протокол від «27» 11 2023 р. № 5

Завідувач кафедри _____ В.В. Лаговський, к.е.н., доцент

Розглянуто і схвалено Вченою радою Факультету фінансів та цифрових технологій, протокол від «12» 12 2023 р. № 5

Голова Вченої ради
факультету фінансів та цифрових технологій _____ В.В. Корнєєв, д.е.н., професор

Завідувач навчально-методичного відділу _____ І.В. Качур, к.біол.н, доцент

Реєстраційний №_____

Зміст

1. ПЕРЕДМОВА

Дисципліна «Об'єктно-орієнтоване програмування» є однією з фундаментальних дисциплін підготовки бакалаврів за освітньо-професійною програмою «Економічна кібернетика». Набуття вмінь та навичок з основ об'єктно-орієнтованого програмування є базою, що забезпечує подальше вивчення спеціальних дисциплін, пов'язаних з фаховою діяльністю.

Матеріал курсу допоможе при підготовці, наукових статей, доповідей на науково-практичних конференціях.

Мета навчальної дисципліни: вивчення студентами базових понять та принципів об'єктно-орієнтованого підходу до програмування і набутті навичок практичного застосування їх при рішенні типових задач програмування. Об'єктно-орієнтоване проектування програмного забезпечення та реалізація його основних принципів вивчається на базі засобів мови Python.

Завдання навчальної дисципліни: ознайомити студентів з ефективними методами об'єктно-орієнтованого програмування, а також існуючим інструментарієм створення складних програмних систем для різноманітних предметних областей, сформувати у студентів знання та практичні навички об'єктно-орієнтованого програмування на основі мови Python та вміння застосовувати стандартні бібліотеки.

Методи навчання:

1. Методи організації та здійснення навчально-пізнавальної діяльності:
- словесні, наочні, практичні методи;
- індуктивні методи і дедуктивний метод;
- творчі, проблемно-пошукові методи;
- навчальна робота під керівництвом, самостійна робота.

2. Методи стимулювання й мотивації навчально-пізнавальної діяльності:
- методи стимулювання інтересу до навчання (створення ситуації інтересу при викладанні того чи іншого матеріалу, навчальні дискусії, аналіз практичних ситуацій);
- методи стимулювання обов'язку й відповідальності (роз'яснення мети навчальної дисципліни, вимоги до вивчення навчальної дисципліни, заохочення).

3. Методи контролю за ефективністю навчально-пізнавальної діяльності:
- метод усного опитування;
- письмовий контроль;
- тестові методи;
- практична контрольна перевірка;
- екзамен.

Форми організації навчання: лекційні заняття, лабораторні роботи, самостійна робота здобувача вищої освіти, індивідуально-консультаційна робота під керівництвом викладача, тестові завдання.

Організація поточного та підсумкового контролю знань.

Поточний контроль здійснюється під час проведенні модульних контрольних робіт, оцінювання результатів виконання лабораторних робіт та завдань для самостійного вирішення, за допомогою тестів.

Підсумковий контроль – диф. залік**.** Підсумкове оцінювання знань здійснюється на основі оцінювання відповідей на теоретичні питання і вирішення практичних завдань.

2. ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Денна форма навчання
(група ЕКБ-21-1)
Опис навчальної дисципліни

<table>
<tr><td rowspan="2">Найменування показників</td><td rowspan="2">Рівень вищої освіти, галузь знань, спеціальність, освітня програма</td><td>Характеристика навчальної дисципліни</td></tr>
<tr><td>денна форма</td></tr>
<tr><td>Кількість кредитів - 5</td><td>Рівень вищої освіти: перший (бакалаврський)</td><td>Обов’язкова</td></tr>
<tr><td>Модулів - 1</td><td rowspan="2">Галузь знань:
05 Соціальні та поведінкові науки
Спеціальність: «Економіка»</td><td rowspan="2">Рік підготовки:
2-й</td></tr>
<tr><td>Змістових модулів - 2</td></tr>
<tr><td rowspan="5">Загальна кількість годин - 150</td><td rowspan="5">Освітні програми:
«Економічна кібернетика»</td><td>Семестр:
4-й</td></tr>
<tr><td rowspan="2">Лекції: 30 год.</td></tr>
<tr><td>Лабораторні роботи: 44 год.</td></tr>
<tr><td>Самостійна робота: 73 год.</td></tr>
<tr><td>Інд. консульт. робота: 3 год.</td></tr>
<tr><td></td><td></td><td>Вид контролю:
Диференційований залік</td></tr>
</table>

2.1. КОМПЕТЕНТНОСТІ І РЕЗУЛЬТАТИ НАВЧАННЯ
Освітня програма «Економічна кібернетика»
(ЕКБ-21-1)

ЗК6. Здатність спілкуватися іноземною мовою. ЗК7. Навички використання інформаційних і комунікаційних технологій. СК7. Здатність застосовувати комп'ютерні технології та програмне забезпечення з обробки даних для вирішення економічних завдань, аналізу інформації та підготовки аналітичних звітів. ФК 2 Здатність застосовувати мови програмування для роботи з базами даних, збору, представлення та аналізу інформації.	ПРН 22. Демонструвати гнучкість та адаптивність у нових ситуаціях, у роботі із новими об'єктами, та у невизначених умовах. ПРН 23. Показувати навички самостійної роботи, демонструвати критичне, креативне, самокритичне мислення.

2.2. ПРЕРЕКВІЗИТИ ТА ПОСТРЕКВІЗИТИ ВИВЧЕННЯ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Пререквізити вивчення дисципліни. Іноземна мова (за професійним спрямуванням, Економічна інформатика.

Постреквізити вивчення дисципліни. Знання, вміння і навички здобуті під час вивчення дисципліни використовуються в наступних дисциплінах: Web-програмування, Штучні нейронні мережі в моделюванні, прогнозуванні та аналізу даних, Прогнозування соціально-економічних процесів..

.

2.3. СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Денна форма навчання

№ п/п	Змістовні модулі	Кількість годин				
		Денна форма				
		Усього	У тому числі			
			Лекції	Лабораторні роботи	Інд.-конс. заняття	Самостійна роббота
Модуль І						
Змістовий модуль 1. Основи програмування на прикладі мови Python						
Т.1	Вступ до програмування мовою Python.	8	2	2	-	4
Т.2	Змінні в мові Python. Оператори в мові програмування Python.	8	2	2	-	4
Т.3	Оператори розгалуження в мові Python. Цикли.	12	2	4	-	6
Т 4	Базові типи для представлення даних.	12	2	4	-	6
Т 5	Рядки. Функції роботи з рядками.	8	2	2	-	4
Т.6	Функції в Python. Функції-генератори.	12	2	4	-	6
Т.7	Анонімні функції та декоратори.	10	2	2	-	6
Т.8	Pattern matching. Виключення та обробка помилок.	12	2	4	-	6
Т.9	Модулі і пакети в Python.	8	2	2	-	4
Форма модульного контролю - контрольна робота						
Разом змістовий модуль 1		**90**	**18**	**26**	**-**	**46**
Змістовий модуль 2. Основи та принципи об’єктно-орієнтованого програмування на прикладі мови Python.						
Т.10	Поняття про об'єктно-орієнтоване програмування.	8	2	2	-	4
Т.11	Інкапсуляція. Статичні методи, поля та методи класу	12	2	4	-	6
Т.12	Успадкування. Спеціальні методи.	12	2	4	-	6
Т.13	Поліморфізм та перезавантаження спеціальних методів. Абстрактні класи.	8	2	2	-	4
Т.14	Робота з файлами в Python. Робота з базами даних в Python на прикладі SQLite.	10	2	4	-	4
Т.15	Робота з датами та часом в Python.	10	2	2	3	3
Разом змістовий модуль 2		**60**	**12**	**18**	**3**	**27**
Разом за модулем І		**150**	**30**	**44**	**3**	**73**
Форма модульного контролю - контрольна робота						
Форма підсумкового контролю – диф. залік						

РЕЙТИНГ-ПЛАН

Денна форма навчання

Години	Тема	Форма заняття та діяльності	Результати навчання	Вага оцінки (кількість балів)
Змістовий модуль 1. Основи програмування на прикладі мови Python				
2	Т.1. Вступ до програмування мовою Python.	Лекція	ПРН 22, ПРН23	**0**
2		Лабораторна робота		**1**
2	Т.2. Змінні в мові Python. Оператори в мові програмування Python.	Лекція	ПРН 22, ПРН23	**0**
2		Лабораторна робота		**1**
2	Т.3. Оператори розгалуження в мові Python. Цикли.	Лекція	ПРН 22, ПРН23	**0**
4		Лабораторна робота		**2*1=2**
2	Т.4. Базові типи для представлення даних.	Лекція	ПРН 22, ПРН23	**0**
4		Лабораторна робота		**2*1=2**
2	Т.5. Рядки. Функції роботи з рядками.	Лекція	ПРН 22, ПРН23	**0**
2		Лабораторна робота		**1**
2	Т.6. Функції в Python. Функції-генератори.	Лекція	ПРН 22, ПРН23	**0**
4		Лабораторна робота		**2*1=2**
2	Т.7. Анонімні функції та декоратори.	Лекція	ПРН 22, ПРН23	**0**
2		Лабораторна робота		**1**
2	Т.8. Pattern matching. Виключення та обробка помилок.	Лекція	ПРН 22, ПРН23	**0**
4		Лабораторна робота		**2*1=2**
2	Т.9. Модулі і пакети в Python.	Лекція	ПРН 22, ПРН23	**0**
2		Лабораторна робота		**1**
Проміжний модульний контроль		Контрольна робота		**9**
Змістовий модуль 2. Основи та принципи об’єктно-орієнтованого програмування на прикладі мови Python.				
2	Т.10. Поняття про об'єктно-орієнтоване програмування.	Лекція	ПРН 22, ПРН23	**0**
2		Лабораторна робота		**1**
2	Т.11. Інкапсуляція. Статичні методи, поля та методи класу.	Лекція	ПРН 22, ПРН23	**0**
4		Лабораторна робота		**2*1=2**
2	Т.12. Успадкування. Спеціальні методи.	Лекція	ПРН 22, ПРН23	**0**
4		Лабораторна робота		**2*1=2**
2	Т.13. Поліморфізм та перезавантаження спеціальних методів. Абстрактні класи.	Лекція	ПРН 22, ПРН23	**0**
2		Лабораторна робота		**1**
2	Т.14. Робота з файлами. в Python. Робота з базами даних в Python на прикладі SQLite.	Лекція	ПРН 22, ПРН23	**0**
4		Лабораторна робота		**2*1=2**
2	Т.15. Робота з датами та часом в Python.	Лекція	ПРН 22, ПРН23	**0**
2		Лабораторна робота		**1**
Індивідуальна робота				**5**
Проміжний модульний контроль		Контрольна робота		**9**
Разом за модулем І				**45**
Комп'ютерне тестування на платформі дистанційного навчання ДПУ MOODLE				**5**
Підсумковий контроль		Диференційований залік		**50**

3. ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Змістовий модуль 1. Основи програмування на прикладі мови Python

Тема 1. Вступ до програмування мовою Python.

План лекційного заняття 1

1. Методи та задачі курсу, зв'язок з іншими дисциплінами.
2. Класифікація прикладного програмного забезпечення.
3. Парадигми програмування.
4. Інтерпретатор Python та його використання.
5. Середовища програмування мовою Python.

Лабораторна робота

Встановлення інтерпретатора і редактора кода Python.

План самостійної роботи здобувачів вищої освіти

1. Опис структури середовищ розробки програмного забезпечення.
2. Принципи створення архітектури сучасних комп'ютерів.
3. Класифікація службових програмних засобів.
4. Історія Python. Область застосування. Місце у сучасному світі.
5. Динаміка та перспективи розвитку Python.
6. Дзен Python.

Перелік питань для самоконтролю

1. Що таке програмне забезпечення?
2. Що таке програмне забезпечення системного рівня.?
3. Що таке прикладне програмне забезпечення?
4. Які є рівні алгоритмічних мов програмування?
5. Назвіть принципи створення архітектури сучасних комп'ютерів?
6. Для яких цілей використовується Python?
7. Як перейти в режим інтерактивного інтерпретатора?
8. Яке розширення мають файли із програмами, написаними на мові Python?
9. Як запустити програму, що міститься у файлі, на виконання у термінальному вікні?

Рекомендовані літературні джерела

Основна:1, 2, 3, 4,

Допоміжна:, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 2. Змінні в мові Python. Оператори в мові програмування Python.

План лекційного заняття 2

1. Іменування змінних, список ключових слів та інструкція для їх виводу.
2. Видалення змінної.
3. Різновидності математичних операторів та їх застосування.
4. Двійкові оператори.
5. Оператори належності.
6. Оператори присвоювання.
7. Пріоритет виконання операторів.
8. Оператори умови.
9. Оператори порівняння.

Лабораторна робота

Застосування операторів

План самостійної роботи здобувачів вищої освіти

1. Анотація змінних.
2. Консольне введення та виведення даних.
3. Коментарі.
4. Логічні операції.
5. Оператор in.

Перелік питань для самоконтролю

1. Який є список ключових слів та інструкція для їх виводу?
2. Які є способи присвоювання значення змінним?
3. Способи видалення змінної.
4. Анотація змінних.
5. Консольне введення та виведення даних.
6. Коментарі.
7. Математичні оператори в Python.
8. Двійкові оператори в Python.
9. Оператори належності в Python.
10. Пріоритет виконання операторів в Python.
11. Оператори умови в Python.
12. Оператори порівняння в Python.
13. Логічні операції в Python.
14. Логічні операції в Python.
15. Оператор in в Python.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 3. Оператори розгалуження в мові Python. Цикли.

План лекційного заняття 3

1. Умовна конструкція **if…else.**
2. Вкладені конструкції **if.**
3. Оператор циклу **for**.
4. Оператор циклу **while.**
5. Вихід із циклу: **break і continue.**
6. Функції **enumerate(), range()**, **len(), zip(), map().**

Лабораторна робота

Розгалуження та цикли в Python

План самостійної роботи здобувачів вищої освіти

1. Умовна конструкція **if…elif…else.**
2. Вкладені цикли в Python.

Перелік питань для самоконтролю

1. Умовна конструкція **if…else.**
2. Вкладені конструкції **if.**
3. Застосування умовної конструкції **if…elif…else.**
4. Вкладені цикли в Python.
5. Застосування циклу **for** в Python.

6. Застосування циклу **while** в Python.
7. Застосування **break і continue** в Python.
8. Функції **enumerate(), range()**, **len(), zip(), map()** в Python**.**

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 4. Базові типи для представлення даних.

План лекційного заняття 4

1. Представлення чисел. Вбудовані функції для роботи з числами.
2. Означення, властивості та застосування списків у мові Python. Способи створення списку. Способи створення копії списку. Операції над списками.
3. Особливості формування та застосування багатовимірних списків.
4. Методи перетворення списку. Сортування списку.
5. Означення, властивості та застосування кортежів, множин. Методи створення кортежів.
6. Створення множини за допомогою функції **set().** Методи для роботи з множинами.
7. Незмінювані множини типу **frozentset.** Оператори та методи, що підтримують перетворення даних типу **frozenset.**
8. Функції, що підтримують перетворення даних типу. Перевірка типу даних.
9. Означення, властивості та застосування словників.
10. Способи створення словників. Операції над словниками: доступ до елементів словника, перевірка існування ключа. Методи для роботи зі словниками.

Лабораторна робота

Робота з колекціями

План самостійної роботи здобувачів вищої освіти

1. Формат представлення та операції над комплексними числами.
2. Основні функції модуля math для роботи з числами.
3. Застосування модуля random для генерації випадкових чисел.
4. Заповнення списку числами. Функція **rang()** та її параметри.
5. Генератори списків та вирази генератори.
6. Властивості та параметри методу **index().**
7. Ітератори.
8. Основні властивості діапазонів.
9. Генератори множин та їх синтаксис.
10. Ітератори, що представлені функціями модуля **itertools.**
11. Перебір елементів словника, кортежа і множини за допомогою циклу **for**.
12. Сортування по ключах.
13. Числа, що задані з фіксованою точністю за підтримки модуля decimal.
14. Виконання операцій над дробами за підтримки методів модуля fractions.

Перелік питань для самоконтролю

1. Властивості списків у мові Python.
2. Способи створення списку. Способи створення копії списку.
3. Операції над списками.
4. Методи додавання та видалення елементів списку.
5. Сортування списку.

6. Властивості кортежів множин та діапазонів.
7. Основна властивість «множин».
8. Методи створення кортежів.
9. Перебір елементів множини в циклі **for.**
10. Методи для роботи з множинами.
11. Оператори та методи, що підтримують перетворення даних типу **frozenset.**
12. Означення, властивості та застосування словників.
13. Операції над словниками.
14. Методи для роботи зі словниками.
15. Перебір елементів словника за допомогою циклу **for**.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 5. Рядки. Функції роботи з рядками.

План лекційного заняття 5

1. Оголошення рядка. Операції над рядками.
2. Екрановані послідовності. Неформатовані рядки.
3. Доступ за індексами. Зрізи. Отримання фрагменту рядка.
4. Функції **len(), max(), min().** Функції для роботи з рядками, що визначають особливості рядка.
5. Функції пошуку та заміни підрядка в рядку**.** Функції що визначають та обробляють початок та кінець рядка.
6. Стилі форматування.
7. Функції вирівнювання рядків.
8. Функції які змінюють регістр символів у рядку.

Лабораторна робота.

Робота з даними типу str.

План самостійної роботи здобувачів вищої освіти

1. Багаторядкові блоки тексту
2. Доступ за індексами.
3. Зрізи.
4. Функції розбиття рядків на частини та утворення нових рядків з допомогою кортежів та списків: **join(), partition(), rpartition(), rsplit(), split(), splitlines()**
5. Операції форматування рядків. Вирази форматування рядків. Бінарний оператор **%.**
6. F-рядок.

Перелік питань для самоконтролю

1. Рядковий тип даних.
2. Операції над рядками.
3. Неформатовані рядки.
4. Багаторядкові блоки тексту.
5. Зрізи.
6. Засоби перетворення рядків та одиночних символів. Функції **int(), str(), repr(), float(), ord(), chr().**
7. Функції **len(), max(), min().**

8. Функції: **isalnum(), isalpha(), isascii(), isdecimal(), isdigit(), isidentifier(), islower(), isnumeric(), isprintable(), isspace(), istitle(), isupper().**
9. Функції: **count(), find(), index(), rfind(), rindex(), replace().**
10. Функції: **endswith(), startswith(), lstrip(), rstrip(), strip().**
11. Функції обробки рядка: **encode(), expandtabs(), format()**. Стилі форматування.
12. Функції вирівнювання рядків: **center(), ljust(), rjust(), zfill().**
13. Функції: **capitalize(), casefold(), lower(), swapcase(), title(), upper().**

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 6. Функції в Python. Функції-генератори.

План лекційного заняття 6

1. Визначення функції як фрагменту коду багаторазового використання. Схема створення функції за допомогою ключового слова **def.** Інструкція **return.**
2. Функції зворотного виклику.
3. Змінне число параметрів функції. Комбінування параметрів.
4. Поняття функції-генератора.
5. Виклик функції-генератора з функції-генератора за допомогою ключового слова **yield.**

Лабораторна робота.

Побудова та застосування функцій при написанні програм розв’язку задач.

План самостійної роботи здобувачів вищої освіти

1. Атрибути функції, вивід списку атрибутів за допомогою функції **dir().**
2. Строгий порядок слідування визначення та виклику функції.
3. Необов’язкові параметри функції та їх зіставлення по ключах.
4. Застосування методу **__next__** до функцій-генераторів.
5. Застосування функції генератора.

Перелік питань для самоконтролю

1. Визначення функції як фрагменту коду багаторазового використання.
2. Схема створення функції за допомогою ключового слова **def.**
3. Інструкція **return.**
4. Функції зворотного виклику.
5. Комбінування параметрів.
6. Атрибути функції, вивід списку атрибутів за допомогою функції **dir().**
7. Строгий порядок слідування визначення та виклику функції.
8. Необов’язкові параметри функції та їх зіставлення по ключах.
9. Поняття функції-генератора.
10. Виклик функції-генератора з функції-генератора за допомогою ключового слова **yield.**
11. Застосування методу **__next__** до функцій-генераторів.
12. Застосування функції генератора.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5
Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5
Міжнародні видання: 1, 2, 3, 4, 5

Тема 7. Анонімні функції та декоратори.

План лекційного заняття 7

1. Анонімні функції **lambda.**
2. Способи задавання та сфери використання анонімних функцій.
3. Декоратори функцій.
4. Видимість глобальних та локальних змінних.
5. Одержання словників глобальних та локальних ідентифікаторів за допомогою функцій **globals()** та **locals().**

Лабораторна робота.

Написання функцій, декораторів та анонімніх функцій.

План самостійної роботи здобувачів вищої освіти

1. Рекурсія, обчислення факторіала.
2. Вкладені функції. Анотації функцій. Атрибут об'єкта функції **__anotations__.**

Перелік питань для самоконтролю

1. Анонімні функції **lambda.**
2. Способи задавання та сфери використання анонімних функцій.
3. Декоратори функцій.
4. Видимість глобальних та локальних змінних.
5. Одержання словників глобальних та локальних ідентифікаторів за допомогою функцій **globals()** та **locals().**
6. Рекурсія.
7. Вкладені функції.
8. Анотації функцій.
9. Атрибут об'єкта функції **__anotations__.**

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6
Допоміжна: 1, 2, 3, 4, 5
Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5
Міжнародні видання: 1, 2, 3, 4, 5

Тема 8. Pattern matching. Виключення та обробка помилок.

План лекційного заняття 8

1. Конструкція match. Кортежі в pattern matching. Альтернативні значення, пропуск елементів, кортеж із невизначеною кількістю елементів.
2. Масиви в pattern matching. Масиви невизначеної довжини, альтернативні значення.
3. Помилки в програмі Python та методи обробки виключень. Інструкція **try...except...else...finally**. Формати інструкції **try**.
4. Опис типів помилок в програмі Python. Основне поняття про клас **Exception** та його атрибути.
5. Поняття про протокол менеджерів контексту. Інструкція **with...as.** Формати інструкції with...as. Конструкція **with open()** та її використання при обробці виключень.
6. Виключення користувача. Застосування інструкцій **raises** та **assert** для створення виключень користувача.

Лабораторна робота.

Застосування конструкції match при написанні програм.

Лабораторна робота.

Обробка помилок.

План самостійної роботи здобувачів вищої освіти

1. Словники в pattern matching.
2. Передача набору значень.
3. Отримання значень ключів, отримання всіх значень.
4. Метод **__exit__()** та його формат.
5. Застосування функції **exc_info()** для одержання інформації про виключення.

Перелік питань для самоконтролю

1. Конструкція match.
2. Кортежі в pattern matching.
3. Альтернативні значення, пропуск елементів, кортеж із невизначеною кількістю елементів.
4. Масиви в pattern matching.
5. Масиви невизначеної довжини, альтернативні значення.
6. Словники в pattern matching. Передача набору значень.
7. Отримання значень ключів, отримання всіх значень.
8. Помилки в програмі Python та методи обробки виключень.
9. Опис типів помилок в програмі Python.
10. Інструкція **try...except...else...finally**.
11. Поняття про протокол менеджерів контексту. Інструкція **with...as.**
12. Конструкція **with open()** та її використання при обробці виключень.
13. Виключення користувача.
14. Застосування інструкцій **raises** та **assert** для створення виключень користувача.
15. Основне поняття про клас **Exception** та його атрибути.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 9. Модулі і пакети в Python.

План лекційного заняття 9

1. Визначення модуля в Python.
2. Головний модуль **"__main__".**
3. Атрибут об’єкта модуля **__main__: __name__.**
4. Інструкція **import** та варіанти її застосування для імпортування модулів.
5. Означення пакета.
6. Файл ініціалізації пакета **__init__.py**.
7. Одержання доступу до ідентифікаторів імпортованого модуля.
8. Імпорт з головного модуля та з файлів пакета.

Лабораторна робота.

Побудова та імпорт модулів та пакетів.

План самостійної роботи здобувачів вищої освіти

1. Імпорт кількох модулів однією інструкцією.
2. Функція **getattr( )** для динамічного формування назви атрибута в ході виконання програми.
3. Перевірка існування атрибута за допомогою функції **hasattr( )**.

4. Способи одержання скомпільованого файла з розширенням ***.pyc**.
5. Шляхи пошуку модулів.
6. Повторне завантаження модулів.
7. Формат функції **reload( )**.

Перелік питань для самоконтролю

1. Визначення модуля в Python.
2. Головний модуль **"__main__".**
3. Атрибут об'єкта модуля **__main__: __name__.**
4. Інструкція **import** та варіанти її застосування для імпортування модулів.
5. Означення пакета.
6. Файл ініціалізації пакета **__init__.py**.
7. Одержання доступу до ідентифікаторів імпортованого модуля.
8. Імпорт з головного модуля та з файлів пакета.
9. Імпорт кількох модулів однією інструкцією.
10. Функція **getattr( )** для динамічного формування назви атрибута в ході виконання програми.
11. Перевірка існування атрибута за допомогою функції **hasattr( )**.
12. Способи одержання скомпільованого файла з розширенням ***.pyc**.
13. Шляхи пошуку модулів.
14. Повторне завантаження модулів.
15. Формат функції **reload( )**.

Індивідуально-консультативна робота

1. Написання модулів та пакетів.
2. Написання бібліотек.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4,
Допоміжна: 1, 2, 3
Інформаційні ресурси Інтернет: 1, 2
Міжнародні видання: 1, 2, 3, 4, 5

Змістовий модуль 2. Основи та принципи об'єктно-орієнтованого програмування на прикладі мови Python.

Тема 10. Поняття про об'єктно-орієнтоване програмування.

План лекційного заняття 19

1. Основні означення, що пов'язані з об'єктно-орієнтованим програмуванням.
2. Визначення класу й створення екземпляра класу.
3. Синтаксис створення атрибуту класу.
4. Спосіб створення методу класу за допомогою інструкції **def.**
5. Застосування змінної **self** для доступу до атрибутів та методів всередині класу.
6. Методи **__init__()** для задавання початкових значень атрибутам екземпляра класу.

Лабораторна робота.

Створення класів.

План самостійної роботи здобувачів вищої освіти

1. Метод **__del__()**, як деструктор, що викликається при видаленні екземпляра класу.
2. Гетери та сетери.
3. Декоратор **@property.**

Перелік питань для самоконтролю

1. Основні означення, що пов’язані з об'єктно-орієнтованим програмуванням.
2. Визначення класу й створення екземпляра класу.
3. Синтаксис створення атрибуту класу.
4. Спосіб створення методу класу за допомогою інструкції **def.**
5. Застосування змінної **self** для доступу до атрибутів та методів всередині класу.
6. Методи **__init__()** для задавання початкових значень атрибутам екземпляра класу.
7. Метод **__del__()**, як деструктор, що викликається при видаленні екземпляра класу.
8. Гетери та сетери.
9. Декоратор **@property.**

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 11. Інкапсуляція. Статичні методи, поля та методи класу

План лекційного заняття 11

1. Ідея інкапсуляції та її реалізація в Python за допомогою методів класу.
2. Інкапсуляція та приховування даних.
3. Статичні методи, поля та методи класу.

Лабораторна робота.

Побудова класів з використанням принципа інкапсуляції.

План самостійної роботи здобувачів вищої освіти

1. Відношення між сутностями.

Перелік питань для самоконтролю

1. Ідея інкапсуляції та її реалізація в Python за допомогою методів класу.
2. Інкапсуляція та приховування даних.
3. Статичні методи, поля та методи класу.
4. Відношення між сутностями.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 12. Успадкування. Спеціальні методи.

План лекційного заняття 12

1. Успадкування. Поняття базового класу (суперкласу) та похідного класу (підкласу).
2. Множинне успадкування.
3. Асоціація, агрегація, композиція.
4. Методи для всіх видів операцій.
5. Методи перевантаження операторів роботи з колекціями.
6. Основні методи для операцій над двійковими числами.
7. Методи для правосторонніх операцій над двійковими числами.

Лабораторна робота

Успадкування класів та перезавантаження методів.

План самостійної роботи здобувачів вищої освіти

1. Використання функції super()
2. Написання класів-домішок.

3. Комбіновані методи для операцій над двійковими числами.
4. Методи для інших операцій над числами.
5. Методи для операцій з дескрипторами.
6. Методи для операцій, що використовуються з диспетчерами контексту.

Перелік питань для самоконтролю

1. Успадкування. Поняття базового класу (суперкласу) та похідного класу (підкласу).
2. Множинне успадкування.
3. Асоціація,.
4. Агрегація.
5. Композиція.
6. Використання функції super()
7. Класи-домішки.
8. Методи для всіх видів операцій.
9. Методи перевантаження операторів роботи з колекціями.
10. Основні методи для операцій над двійковими числами.
11. Методи для правосторонніх операцій над двійковими числами.
12. Комбіновані методи для операцій над двійковими числами.
13. Методи для інших операцій над числами.
14. Методи для операцій з дескрипторами.
15. Методи для операцій, що використовуються з диспетчерами контексту.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 13. Поліморфізм та перезавантаження спеціальних методів. Абстрактні класи.

План лекційного заняття 13

1. Поліморфізм. Загальні відомості про перевантаження операторів. Принципи, що лежать в основі перевантаження операторів у класах.
2. Перелік методів, які можна перевантажувати. Перевантаження доступу за індексом []. Метод __getitem__().
3. Перевантаження доступу за індексом. Встановлення нового значення. Метод __setitem__().
4. Перевантаження бінарних арифметичних операторів +, –, *, /, //, %.
5. Поняття абстрактного класу. Абстрактні методи.
6. Поняття про метакласи.

Лабораторна робота.

Перевантаження арифметичних операторів

План самостійної роботи здобувачів вищої освіти

1. Віртуальні методи.
2. Метод __getattribute__(), який викликається при доступі до будь-якого атрибута класу.
3. Метод __setattr__(), який викликається при спробі присвоювання значення атрибуту екземпляра класу.
4. Метод __delattr__(), який викликається при видаленні атрибута за допомогою інструкції del .
5. Методи, що викликаються при виконанні арифметичних та логічних операцій.
6. Інтерфейси.

Перелік питань для самоконтролю

1. Поліморфізм.
2. Принципи, що лежать в основі перевантаження операторів у класах.
3. Перелік методів, які можна перевантажувати.
4. Перевантаження доступу за індексом. Метод __getitem__().
5. Перевантаження доступу за індексом. Встановлення нового значення. Метод __setitem__().
6. Перевантаження бінарних арифметичних операторів +, –, *, /, //, %.
7. Віртуальні методи.
8. Метод __getattribute__(), який викликається при доступі до будь-якого атрибута класу.
9. Метод __setattr__(), який викликається при спробі присвоювання значення атрибуту екземпляра класу.
10. Метод __delattr__(), який викликається при видаленні атрибута за допомогою інструкції del .
11. Абстрактний клас.
12. Абстрактні методи.
13. Поняття про метекласи.
14. Інтерфейси.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 14. Робота з файлами в Python.

План лекційного заняття 25

1. Основні поняття про файл та типи файлів. Відкриття файлу та формат функції **open()**.
2. Модифікатори відкриття файлу. Особливості роботи з механізмом буферизації файлів.
3. Методи для роботи з файлами. Функції для маніпулювання файлами.
4. Перевірка наявності файлу, розміру файлу, часу останнього доступу, часу створення, часу останньої зміни. Збереження об’єктів у файлі.
5. Функції модуля pickle для роботи з файлами. Використання модуля shelve для зберігання даних в файлі по ключу.
6. Підключення до SQLite. Отримання курсору.
7. Створення таблиці. Додавання даних.
8. Отримання даних. Оновлення даних. Видалення даних.

Лабораторна робота.

Робота з файлами в Python.

План самостійної роботи здобувачів вищої освіти

1. Перетворення відносного шляху у абсолютний.
2. Можливі задавання шляхів до файлу та їх модифікація.
3. Встановлення шляху до поточного каталогу.
4. Одержання шляху до файлу, що виконується, за допомогою атрибута **__file__**.
5. Одержання повного шляху до файлу.
6. Права доступу до файлів і каталогів.
7. Визначення прав доступу.
8. Функції для перетворення шляху до файлу.

9. Перенаправлення вводу/виводу. Функція **print( ).**
10. Функції для роботи з каталогами.
11. Виключення, які виникають при виконанні операцій з файлами.
12. Виконання запитів до бази даних.
13. Множинна вставка.
14. Основні операції з даними в SQLite.

Перелік питань для самоконтролю

1. Відкриття файлу та формат функції **open( )**. Абсолютний та відносний шлях до файлу.
2. Модифікатори відкриття файлу. Методи для роботи з файлами.
3. Функції для маніпулювання файлами.
4. Перевірка наявності файлу, розміру файлу, часу останнього доступу, часу створення, часу останньої зміни.
5. Функції модуля pickle для роботи з файлами. Використання модуля shelve для зберігання даних в файлі по ключу.
6. Перетворення відносного шляху у абсолютний. Можливі задавання шляхів до файлу та їх модифікація.
7. Одержання шляху до файлу, що виконується, за допомогою атрибута **__file__**. Права доступу до файлів і каталогів.
8. Підключення до SQLite. Отримання курсору.
9. Створення таблиці. Додавання даних.
10. Отримання даних.
11. Оновлення даних.
12. Видалення даних.
13. Множинна вставка.
14. Основні операції з даними в SQLite.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

Тема 15. Робота з датами та часом в Python.

План лекційного заняття 27

1. Модуль datetime.
2. Клас date.
3. Клас часу.
4. Клас datetime.
5. Перетворення з рядка на дату.
6. Отримання дат та часу.

Лабораторна робота.

Використання бібліотек по роботі з часом і датами при написанні програм.

План самостійної роботи здобувачів вищої освіти

1. Складання та віднімання дат і часу.
2. Властивості timedelta.
3. Порівняння дат.

Індивідуально-консультаційна робота

Написання додатку з використанням парадигми об’єктно-орієнтованого програмування.

Перелік питань для самоконтролю

1. Модуль datetime.
2. Клас date.
3. Клас часу.
4. Клас datetime.
5. Перетворення з рядка на дату.
6. Отримання дат та часу.
7. Складання та віднімання дат і часу.
8. Властивості timedelta.
9. Порівняння дат.

Рекомендовані літературні джерела

Основна:1, 2, 3, 4, 5, 6

Допоміжна: 1, 2, 3, 4, 5

Інформаційні ресурси Інтернет: 1, 2, 3, 4, 5

Міжнародні видання: 1, 2, 3, 4, 5

КРИТЕРІЇ ОЦІНЮВАННЯ РІВНЯ НАВЧАЛЬНИХ ДОСЯГНЕНЬ ЗДОБУВАЧІВ ВИЩОЇ ОСВІТИ

Загальний розподіл балів, які здобувач вищої освіти може отримати в межах 100-бальної системи оцінювання, представлено в таблиці.

Максимальна кількість балів отримана здобувачем вищої освіти на лабораторному занятті становить 2 бали для денної форми навчання.

Виконання самостійної роботи, як правило, оцінюється під час проведення лабораторного заняття у вигляді опитування в тому числі за питаннями, які виносяться на самостійну роботу.

Шкала оцінювання роботи здобувачів вищої освіти на лабораторних заняттях

Кількість балів	Критерії оцінювання
1	В повному обсязі володіє навчальним матеріалом, вільно самостійно та аргументовано його викладає під час усних відповідей, глибоко та всебічно розкриває зміст теоретичних питань та практичних завдань, використовуючи при цьому обов'язкову та додаткову літературу. Написані програми самостійно і працюють коректно.
0	Не володіє навчальним матеріалом та не в змозі його викласти, не розуміє змісту теоретичних питань та практичних завдань, не вирішив жодного практичного завдання.

Критерії оцінювання контрольних робіт.

Формою проміжного поточного контролю є контрольні роботи, які проводяться у письмовій формі та кожна з яких оцінюється від 0 до 1 бала.

Розподіл балів за різні види завдань в межах контрольної роботи

Вид завдання	Максимальна кількість балів за виконання
Теоретичні питання	6*1 = 6
Практичне завдання	3*1 = 3
Всього	9

Критерії оцінювання відповіді на теоретичне питання

Критерії оцінювання	Кількість балів
Оцінюється робота здобувача вищої освіти, який у повному обсязі дав відповіді на всі питання. При цьому використовував актуальну наукову термінологію, належним чином обґрунтовував свої думки та зробив узагальнені підсумки.	1
Оцінюється робота здобувача вищої освіти, який дав неправильну відповідь на всі теоретичні питання, допустив істотні помилки, оперував неактуальною застарілою інформацією або відповіді на питання відсутні взагалі.	0

Критерії оцінювання відповіді на практичне завдання

Критерії оцінювання	Кількість балів
Оцінюється робота здобувача вищої освіти, який у повному обсязі дав відповіді на всі практичні завдання. При цьому використовував актуальну наукову термінологію, належним чином обґрунтовував свої думки та зробив узагальнені підсумки.	1
Оцінюється робота здобувача вищої освіти, який дав неправильну відповідь на всі практичні завдання, допустив істотні помилки, оперував неактуальною застарілою інформацією або відповіді на питання відсутні взагалі.	0

Критерії оцінювання тестового контролю на платформі Moodle

Критерії оцінювання	Кількість балів
Оцінюється робота здобувача вищої освіти, який повністю розкрив всі питання.	5
Оцінюється робота здобувача вищої освіти, який дав відповіді на 85-95% всіх питань.	4
Оцінюється робота здобувача вищої освіти, який дав відповіді на 75-84% всіх питань	3
Оцінюється робота здобувача вищої освіти, який дав відповіді на 45-75% всіх питань	2
Оцінюється робота здобувача вищої освіти, який розкрив сутність менше половини питань.	1
Оцінюється робота здобувача вищої освіти, який взагалі не розкрив сутність визначень.	0

Критерії оцінювання індивідуально-консультаційної роботи.

Індивідуальна робота оцінюється від 0 до 5 балів.

Шкала оцінювання індивідуально-консультаційної роботи здобувачів вищої освіти

Кількість балів	Критерії оцінювання
5	Робота виконана повністю, правильно оформлена. Не містить помилок. Висновки зроблені і правильні. Показано відмінне володіння матеріалом.
4	Робота виконана повністю, допущено неправильне оформлення. Допущені помилки і неточності які призводять до неправильного результату. Висновки не зроблені або можуть бути не повні. Показано задовільне володіння матеріалом.
2-3	Робота виконана не повністю, допущено неправильне оформлення. Допущені помилки і неточності які призводять до неправильного результату. Висновки не зроблені або можуть бути не повні. Показано задовільне володіння матеріалом.
1	Робота виконана не повністю, допущені грубі помилки.
0	Не виконано індивідуальну роботу.

Підсумкове оцінювання знань здобувачів вищої освіти здійснюється за результатами поточного контролю (від 0 до 50 балів) та диференційованого заліку (від 0 до 50 балів). Критерієм успішного проходження здобувачем освіти підсумкового оцінювання є отримання не менше 25 балів за поточний контроль та 25 балів за підсумковий контроль у формі диференційованого заліку.

Критерії оцінювання підсумкового контролю

Розподіл балів за різні види завдань в межах підсумкового контролю

Вид завдання	Максимальна кількість балів за виконання
Теоретичні питання (2 питання)	2*10=20
Практичне завдання (3 завданя)	3*10=30
Всього	50

Критерії оцінювання відповіді на теоретичне питання

Критерії оцінювання	Кількість балів
Оцінюється робота здобувача вищої освіти, який у повному обсязі дав відповіді на всі питання. При цьому використовував актуальну наукову	10

термінологію, належним чином обґрунтовував свої думки та зробив узагальнені підсумки.	
Оцінюється робота здобувача вищої освіти, який у повному обсязі дав відповіді на всі питання. При цьому не використовував актуальну наукову термінологію, належним чином не обґрунтовував свої думки та не зробив узагальнені підсумки.	9-7
Оцінюється робота здобувача вищої освіти, який дав фрагментарні відповіді на теоретичні питання (без аргументації й обґрунтування, підсумків), у відповідях присутні неточності відповідь дана лише на окремі питання.	4-6
Оцінюється робота здобувача вищої освіти, який дав фрагментарні відповіді на теоретичні питання (без аргументації й обґрунтування, підсумків), у відповідях присутні помилки.	1-3
Оцінюється робота здобувача вищої освіти, який дав неправильну відповідь на теоретичне питання, допустив істотні помилки, оперував неактуальною застарілою інформацією або відповіді на питання відсутні взагалі.	0

Критерії оцінювання відповіді на практичне завдання

Критерії оцінювання	**Кількість балів**
Оцінюється робота здобувача вищої освіти, який у повному обсязі розв'язав практичні завдання. При цьому використовував актуальну наукову термінологію, належним чином обґрунтовував свої думки та зробив узагальнені підсумки.	10
Оцінюється робота здобувача вищої освіти, який у повному обсязі розв'язав практичні завдання. При цьому не використовував актуальну наукову термінологію, належним чином не обґрунтовував свої думки та зробив не узагальнені підсумки	9-8
Оцінюється робота здобувача вищої освіти, який в основному розкрив зміст практичного завдання. Проте, допускалися при цьому окремі неістотні неточності та незначні помилки.	7-5
Оцінюється робота здобувача вищої освіти, який в основному розкрив зміст практичного завдання. Проте, при висвітленні деяких питань не вистачало достатньої аргументації, допускалися при цьому окремі неістотні неточності та незначні помилки.	4-3
Оцінюється робота здобувача вищої освіти, який дав фрагментарні відповіді на практичні завдання у відповідях присутні неточності та помилки.	2-1
Оцінюється робота здобувача вищої освіти, який дав неправильну відповідь на всі практичні завдання, допустив істотні помилки, оперував неактуальною застарілою інформацією або відповіді на питання відсутні взагалі.	0

Переведення даних 100-бальної шкали оцінювання в національну шкалу та шкалу за системою ЄКТС здійснюється в такому порядку:

Сума балів за 100-бальною шкалою	**Оцінка в ЄКТС**	**Значення оцінки в ЄКТС**	**Оцінка за національною шкалою**	
			Екзамен	**Диф. залік**
90-100	A	відмінно	відмінно	відмінно
80-89	B	дуже добре	добре	добре
70-79	C	добре		
60-69	D	задовільно	задовільно	задовільно
50-59	E	достатньо		
35-49	FX	незадовільно з можливістю повторного складання	незадовільно	незадовільно

0-34	F	незадовільно з обов'язковим повторним вивченням курсу		

Результати складання диференційованого заліку/екзамену оцінюються за чотирибальною шкалою («відмінно», «добре», «задовільно», «незадовільно») і вносяться у відомість обліку успішності здобувача вищої освіти, залікову книжку, індивідуальний навчальний план здобувача вищої освіти.

5. ЗАСОБИ ДІАГНОСТИКИ РЕЗУЛЬТАТІВ НАВЧАННЯ

Перелік засобів оцінювання, які застосовуються при вивченні навчальної дисципліни:

- диференційований залік;
- тести;
- комп'ютерне тестування на платформі MOODLE ДПУ;
- лабораторні роботи;
- контрольна роботи.

6. ФОРМИ ТА ПЕРЕЛІК ПИТАНЬ ДО ПІДСУМКОВОГО ТА ПОТОЧНОГО КОНТРОЛЮ

Форми поточного контролю:
1) модульна контрольні роботи;
2) письмові, усні опитування на лабораторних заняттях;
3) тестування.

Перелік питань до поточного контролю

Змістовий модуль 1

1. Що таке програмне забезпечення?
2. Які є рівні мов програмування?
3. Для яких цілей використовується Python?
4. Які є базові типи даних мови програмування Python?
5. Які є способи присвоювання значення змінним?
6. Функція перевірки типу даних.
7. Способи перетворення типів даних.
8. Способи видалення змінної.
9. Анотація змінних.
10. Консольне введення та виведення даних.
11. Коментарі.
12. Різновидності математичних операторів та їх застосування.
13. Оператори належності.
14. Пріоритет виконання операторів.
15. Оператори умови.
16. Оператори порівняння.
17. Логічні операції.
18. Оператор in.
19. Умовна конструкція **if…else.**
20. Вкладені конструкції **if.**
21. Умовна конструкція **if…elif…else.**
22. Оператор циклу **for**.
23. Оператор циклу **while.**

24. Вихід із циклу: **break і continue.**
25. Функції **enumerate(), range()**, **len(), zip(), map().**
26. Вкладені цикли.
27. Представлення чисел в десятковій, двійковій, вісімковій та шістнадцятковій системах числення.
28. Вбудовані функції для роботи з числами.
29. Формат представлення та операції над комплексними числами.
30. Основні функції модуля math для роботи з числами.
31. Застосування модуля random для генерації випадкових чисел.
32. Оголошення рядка.
33. Операції над рядками.
34. Екрановані послідовності.
35. Багаторядкові блоки тексту
36. Доступ за індексами.
37. Зрізи.
38. Отримання фрагменту рядка.
39. Засоби перетворення рядків та одиночних символів. Функції **int(), str(), repr(), float(), ord(), chr().**
40. Функції **len(), max(), min().**
41. Функції для роботи з рядками, що визначають особливості рядка: **isalnum(), isalpha(), isascii(), isdecimal(), isdigit(), isidentifier(), islower(), isnumeric(), isprintable(), isspace(), istitle(), isupper().**
42. Функції пошуку та заміни підрядка в рядку: **count(), find(), index(), rfind(), rindex(), replace().**
43. Функції що визначають та обробляють початок та кінець рядка: **endswith(), startswith(), lstrip(), rstrip(), strip().**
44. Функції обробки рядка згідно з форматом чи правилом кодування: **encode(), expandtabs(), format()**. Стилі форматування
45. Функції вирівнювання рядків: **center(), ljust(), rjust(), zfill().**
46. Функції які змінюють регістр символів у рядку: **capitalize(), casefold(), lower(), swapcase(), title(), upper().**
47. Функції розбиття рядків на частини та утворення нових рядків з допомогою кортежів та списків: **join(), partition(), rpartition(), rsplit(), split(), splitlines()**
48. Операції форматування рядків. Вирази форматування рядків. Бінарний оператор **%.**
49. F-рядок.
50. Означення, властивості та застосування списків у мові Python.
51. Операції над списками.
52. Особливості формування та застосування багатовимірних списків.
53. Перебір елементів списку.
54. Методи додавання та видалення елементів списку.
55. Пошук елемента списку й одержання відомостей про значення, які входять в список.
56. Визначення кількості елементів.
57. Перевертання та перемішування списку.
58. Сортування списку.
59. Заповнення списку числами.
60. Перетворення списку на рядок.
61. Функція **rang()** та її параметри.

62. Генератори списків та вирази генератори.
63. Властивості та параметри методу **index().**
64. Означення, властивості та застосування кортежів множин та діапазонів.
65. Незмінювані послідовності типу кортеж.
66. Основна властивість типу «множина».
67. Особливості змінюваних і незмінюваних множин.
68. Методи створення кортежів. Одержання елемента кортежу по індексу.
69. Створення множини за допомогою функції **set().**
70. Перебір елементів множини в циклі **for.**
71. Методи для роботи з множинами.
72. Незмінювані множини типу **frozentset.**
73. Оператори та методи, що підтримують перетворення даних типу **frozenset.**
74. Ітератори.
75. Означення та застосування операторів на множинах, які виконують логічні перетворення множин.
76. Генератори множин та їх синтаксис.
77. Ітератори, що представлені функціями модуля **itertools.**
78. Означення, властивості та застосування словників.
79. Означення словників як відображень та порівняння їх властивостей з властивостями асоціативних масивів в інших мовах програмування.
80. Способи створення словників.
81. Операції над словниками: доступ до елементів словника, перевірка існування ключа.
82. Вилучення елементів словника за функцією **del.**
83. Методи для роботи зі словниками.
84. Генератори словників.
85. Визначення кількості ключів.
86. Перебір елементів словника за допомогою циклу **for**.
87. Сортування по ключах.
88. Схема створення функції за допомогою ключового слова **def.**
89. Інструкція **return.**
90. Функції зворотного виклику.
91. Змінне число параметрів функції.
92. Комбінування параметрів.
93. Атрибути функції, вивід списку атрибутів за допомогою функції **dir().**
94. Анонімні функції **lambda.**
95. Способи задавання та сфери використання анонімних функцій.
96. Декоратори функцій.
97. Видимість глобальних та локальних змінних.
98. Одержання словників глобальних та локальних ідентифікаторів за допомогою функцій **globals()** та **locals().**
99. Рекурсія.
100. Вкладені функції.
101. Анотації функцій.
102. Атрибут об’єкта функції **__anotations__.**
103. Поняття функції-генератора.
104. Виклик функції-генератора з функції-генератора за допомогою ключового слова **yield.**

105. Застосування методу **__next__** до функцій-генераторів.
106. Конструкція match.
107. Кортежі в pattern matching.
108. Альтернативні значення, пропуск елементів, кортеж із невизначеною кількістю елементів.
109. Масиви в pattern matching.
110. Масиви невизначеної довжини, альтернативні значення.
111. Словники в pattern matching.
112. Помилки в програмі Python та методи обробки виключень.
113. Опис типів помилок в програмі Python. Інструкція **try...except...else...finally**. Формати інструкції **try**.
114. Поняття про протокол менеджерів контексту. Інструкція **with...as.** Формати інструкції with...as.
115. Конструкція **with open()** та її використання при обробці виключень.
116. Виключення користувача. Застосування інструкцій **raises** та **assert** для створення виключень користувача.
117. Основне поняття про клас **Exception** та його атрибути.
118. Метод **__exit__()** та його формат.
119. Застосування функції **exc_info()** для одержання інформації про виключення.
120. Визначення модуля в Python. Головний модуль **"__main__".**
121. Атрибут об'єкта модуля **__main__: __name__.**
122. Інструкція **import** та варіанти її застосування для імпортування модулів.
123. Означення пакета. Файл ініціалізації пакета **__init__.py**.
124. Імпорт з головного модуля та з файлів пакета. Імпорт кількох модулів однією інструкцією.
125. Функція **getattr( )** для динамічного формування назви атрибута в ході виконання програми.
126. Перевірка існування атрибута за допомогою функції **hasattr( )**.
127. Способи одержання скомпільованого файла з розширенням ***.pyc**.

Змістовий модуль 1

1. Основні поняття, що пов'язані з об'єктно-орієнтованим програмуванням.
2. Визначення класу й створення екземпляра класу.
3. Синтаксис створення атрибуту класу.
4. Спосіб створення методу класу за допомогою інструкції **def.**
5. Застосування змінної **self** для доступу до атрибутів та методів всередині класу.
6. Методи **__init__()** для задавання початкових значень атрибутам екземпляра класу.
7. Метод **__del__()**, як деструктор, що викликається при видаленні екземпляра класу.
8. Гетери та сетери.
9. Декоратор **@property.**
10. Ідея інкапсуляції та її реалізація в Python за допомогою методів класу.
11. Статичні методи, поля та методи класу.
12. Відношення між сутностями.
13. Успадкування. Поняття базового класу (суперкласу) та похідного класу (підкласу).
14. Множинне успадкування.
15. Асоціація, агрегація, композиція.
16. Використання функції super().
17. Написання класів-домішок.

18. Звертання до класів інших модулів. Загальноприйняті домовленості до задавання імен модулів та класів
19. Методи перевантаження операторів роботи з колекціями.
20. Основні методи для операцій над двійковими числами.
21. Методи для правосторонніх операцій над двійковими числами.
22. Комбіновані методи для операцій над двійковими числами.
23. Методи для операцій, що використовуються з диспетчерами контексту.
24. Поліморфізм.
25. Загальні відомості про перевантаження операторів. Принципи, що лежать в основі перевантаження операторів у класах.
26. Перелік методів, які можна перевантажувати.
27. Перевантаження доступу за індексом. Метод __getitem__().
28. Перевантаження доступу за індексом. Встановлення нового значення. Метод __setitem__().
29. Віртуальні методи.
30. Метод __getattribute__(), який викликається при доступі до будь-якого атрибута класу.
31. Метод __setattr__(), який викликається при спробі присвоювання значення атрибуту екземпляра класу.
32. Метод __delattr__(), який викликається при видаленні атрибута за допомогою інструкції del .
33. Методи, що викликаються при виконанні арифметичних та логічних операцій
34. Поняття абстрактного класу.
35. Абстрактні методи.
36. Метакласи.
37. Інтерфейси.
38. Відкриття файлу та формат функції **open( )**.
39. Абсолютний та відносний шлях до файлу.
40. Кодування текстових файлів та особливості роботи з різними кодуваннями.
41. Методи для роботи з файлами. Функції для маніпулювання файлами.
42. Перевірка наявності файлу, розміру файлу, часу останнього доступу, часу створення, часу останньої зміни.
43. Збереження об’єктів у файлі.
44. Функції модуля pickle для роботи з файлами.
45. Використання модуля shelve для зберігання даних в файлі по ключу.
46. Перетворення відносного шляху у абсолютний.
47. Встановлення шляху до поточного каталогу.
48. Одержання шляху до файлу, що виконується, за допомогою атрибута **__file__**.
49. Права доступу до файлів і каталогів.
50. Функції для роботи з каталогами.
51. Виключення, які виникають при виконанні операцій з файлами.
52. Підключення до SQLite.
53. Отримання курсору.
54. Створення таблиці.
55. Додавання даних.
56. Встановлення параметрів.
57. Отримання даних.
58. Оновлення даних.

59. Видалення даних.
60. Виконання запитів до бази даних.
61. Множинна вставка.
62. Модуль datetime.
63. Клас date.
64. Клас datetime.
65. Перетворення з рядка на дату.
66. Отримання дат та часу.
67. Складання та віднімання дат і часу.
68. Порівняння дат.

Перелік питань до підсумкового контролю

1. Рівні мов програмування.
2. Які є базові типи даних мови програмування Python?
3. Які є способи присвоювання значення змінним?
4. Функція перевірки типу даних. Способи перетворення типів даних.
5. Анотація змінних. Коментарі.
6. Консольне введення та виведення даних.
7. Різновидності математичних операторів та їх застосування.
8. Оператори належності.
9. Пріоритет виконання операторів.
10. Оператори умови.
11. Оператори порівняння.
12. Логічні операції.
13. Оператор in.
14. Умовна конструкція **if…elif…else.**
15. Оператор циклу **for**.
16. Оператор циклу **while.**
17. Вихід із циклу: **break і continue.**
18. Функції **enumerate(), range()**, **len(), zip(), map().**
19. Вбудовані функції для роботи з числами.
20. Основні функції модуля math для роботи з числами.
21. Застосування модуля random для генерації випадкових чисел.
22. Оголошення рядка. Операції над рядками.
23. Зрізи. Отримання фрагменту рядка.
24. Засоби перетворення рядків та одиночних символів. Функції **int(), str(), repr(), float(), ord(), chr().**
25. Функції **len(), max(), min().**
26. Функції для роботи з рядками, що визначають особливості рядка: **isalnum(), isalpha(), isascii(), isdecimal(), isdigit(), isidentifier(), islower(), isnumeric(), isprintable(), isspace(), istitle(), isupper().**
27. Функції пошуку та заміни підрядка в рядку: **count(), find(), index(), rfind(), rindex(), replace().**
28. Функції що визначають та обробляють початок та кінець рядка: **endswith(), startswith(), lstrip(), rstrip(), strip().**
29. Функції обробки рядка згідно з форматом чи правилом кодування: **encode(), expandtabs(), format()**. Стилі форматування

30. Функції вирівнювання рядків: **center(), ljust(), rjust(), zfill().**

31. Функції які змінюють регістр символів у рядку: **capitalize(), casefold(), lower(), swapcase(), title(), upper().**

32. Функції розбиття рядків на частини та утворення нових рядків з допомогою кортежів та списків: **join(), partition(), rpartition(), rsplit(), split(), splitlines()**

33. Операції форматування рядків. Вирази форматування рядків. Бінарний оператор **%.** F-рядок.

34. Означення, властивості та застосування списків у мові Python.

35. Операції над списками.

36. Особливості формування та застосування багатовимірних списків.

37. Перебір елементів списку. Методи додавання та видалення елементів списку.

38. Пошук елемента списку й одержання відомостей про значення, які входять в список.

39. Визначення кількості елементів.

40. Перетворення списку на рядок.

41. Функція **rang()** та її параметри.

42. Генератори списків та вирази генератори.

43. Властивості та параметри методу **index().**

44. Означення, властивості та застосування кортежів, множин.

45. Особливості змінюваних і незмінюваних множин.

46. Методи створення кортежів. Одержання елемента кортежу по індексу.

47. Створення множини за допомогою функції **set().**

48. Перебір елементів множини в циклі **for.**

49. Методи для роботи з множинами.

50. Незмінювані множини типу **frozentset.**

51. Оператори та методи, що підтримують перетворення даних типу **frozenset.**

52. Ітератори.

53. Означення та застосування операторів на множинах, які виконують логічні перетворення множин.

54. Генератори множин та їх синтаксис.

55. Ітератори, що представлені функціями модуля **itertools.**

56. Означення, властивості та застосування словників.

57. Способи створення словників.

58. Операції над словниками: доступ до елементів словника, перевірка існування ключа.

59. Вилучення елементів словника за функцією **del.**

60. Методи для роботи зі словниками.

61. Генератори словників.

62. Перебір елементів словника за допомогою циклу **for**.

63. Схема створення функції за допомогою ключового слова **def.**

64. Інструкція **return.**

65. Атрибути функції, вивід списку атрибутів за допомогою функції **dir().**

66. Анонімні функції **lambda.** Способи задавання та сфери використання анонімних функцій.

67. Декоратори функцій.

68. Видимість глобальних та локальних змінних.

69. Одержання словників глобальних та локальних ідентифікаторів за допомогою функцій **globals()** та **locals().**

70. Вкладені функції.

71. Анотації функцій.

72. Атрибут об’єкта функції **__anotations__.**

73. Поняття функції-генератора. Виклик функції-генератора з функції-генератора за допомогою ключового слова **yield.** Застосування методу **__next__** до функцій-генераторів.

74. Конструкція match.

75. Кортежі в pattern matching.

76. Масиви в pattern matching.

77. Словники в pattern matching.

78. Помилки в програмі Python та методи обробки виключень.

79. Опис типів помилок в програмі Python. Інструкція **try...except...else...finally**. Формати інструкції **try**.

80. Поняття про протокол менеджерів контексту. Інструкція **with...as.** Формати інструкції with...as.

81. Конструкція **with open()** та її використання при обробці виключень.

82. Виключення користувача. Застосування інструкцій **raises** та **assert** для створення виключень користувача.

83. Основне поняття про клас **Exception** та його атрибути.

84. Метод **__exit__()** та його формат.

85. Застосування функції **exc_info()** для одержання інформації про виключення.

86. Визначення модуля в Python.

87. Головний модуль **"__main__".**

88. Інструкція **import** та варіанти її застосування для імпортування модулів.

89. Означення пакета. Файл ініціалізації пакета **__init__.py**.

90. Одержання доступу до ідентифікаторів імпортованого модуля.

91. Імпорт з головного модуля та з файлів пакета. Імпорт кількох модулів однією інструкцією.

92. Функція **getattr( )** для динамічного формування назви атрибута в ході виконання програми.

93. Перевірка існування атрибута за допомогою функції **hasattr( )**.

94. Шляхи пошуку модулів. Повторне завантаження модулів. Формат функції **reload( )**.

95. Визначення класу й створення екземпляра класу. Синтаксис створення атрибуту класу.

96. Спосіб створення методу класу за допомогою інструкції **def.**

97. Застосування змінної **self** для доступу до атрибутів та методів всередині класу.

98. Методи **__init__()** для задавання початкових значень атрибутам екземпляра класу.

99. Метод **__del__()**, як деструктор, що викликається при видаленні екземпляра класу.

100. Гетери та сетери.

101. Декоратор **@property.**

102. Ідея інкапсуляції та її реалізація в Python за допомогою методів класу.

103. Статичні методи, поля та методи класу.

104. Успадкування. Поняття базового класу (суперкласу) та похідного класу (підкласу).

105. Множинне успадкування.

106. Асоціація, агрегація, композиція.

107. Використання функції super().

108. Написання класів-домішок.

109. Звертання до класів інших модулів. Загальноприйняті домовленості до задавання імен модулів та класів

110. Методи для операцій.

111. Поліморфізм.

112. Загальні відомості про перевантаження операторів. Принципи, що лежать в основі перевантаження операторів у класах.

113. Метод __getitem__(). Перевантаження доступу за індексом. Встановлення нового значення. Метод __setitem__().

114. Метод __getattribute__(), який викликається при доступі до будь-якого атрибута класу.

115. Метод __setattr__(), який викликається при спробі присвоювання значення атрибуту екземпляра класу.

116. Метод __delattr__(), який викликається при видаленні атрибута за допомогою інструкції del.

117. Поняття абстрактного класу. Абстрактні методи.

118. Метакласи.

119. Інтерфейси.

120. Відкриття файлу та формат функції **open( )**. Абсолютний та відносний шлях до файлу.

121. Кодування текстових файлів та особливості роботи з різними кодуваннями.

122. Методи для роботи з файлами. Функції для маніпулювання файлами.

123. Перевірка наявності файлу, розміру файлу, часу останнього доступу, часу створення, часу останньої зміни.

124. Збереження об’єктів у файлі.

125. Функції модуля pickle для роботи з файлами.

126. Використання модуля shelve для зберігання даних в файлі по ключу.

127. Встановлення шляху до поточного каталогу. Одержання шляху до файлу, що виконується, за допомогою атрибута **__file__**.

128. Права доступу до файлів і каталогів.

129. Функції для роботи з каталогами.

130. Виключення, які виникають при виконанні операцій з файлами.

131. Підключення до SQLite.

132. Отримання курсору.

133. Створення таблиці.

134. Додавання даних.

135. Отримання даних.

136. Оновлення даних.

137. Видалення даних.

138. Виконання запитів до бази даних.

139. Множинна вставка.

140. Модуль datetime.

141. Клас date.

142. Клас datetime.

143. Отримання дат та часу.

7. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна:

1. Алхімова С.М. Об’єктно-орієнтоване програмування / С.М. Алхімова. Київ: КПІ ім. Ігоря Сікорського, 2019, 190 с.
2. Васильєв О. М. Програмування в PYTHON. Теорія і практика : Навчальний посібник Тернопіль: Ліра-К, 2023. 462 с.
3. Васильєв О. М. Програмування мовою Python. Тернопіль : Навчальна книга – Богдан, 2019. 504 с.
4. Гришанович Т.О. Основи об'єктно-орієнтованого програмування / Т.О. Гришанович. Харків. 2020. 102 с
5. Костюченко А. О. Основи програмування мовою Python : навч. посіб. Чернігів : ФОП Баликіна С. М., 2020. 180 с.
6. Яковенко А. В. Основи програмування. Python. Частина 1 : підручник. Київ : КПІ ім. Ігоря Сікорського, 2018. 195 с.

Допоміжна:

1. Григорович В.Г. Алгоритмізація та програмування. Львів: Навчальний посібник – Магнолія - 2006. 2023. 357с
2. Григорович В.Г. Алгоритмізація та програмування. Львів: Навчальний посібник- Магнолія - 2006. 2024. 268с
3. Злобін Г.Г. Алгоритмізація та програмування. Київ: Підручник- Каравела. 2023. 168с.
4. Ковалюк Т.В. Алгоритмізація та програмування. Львів: Підручник- Магнолія 2006. 2021. 400 с.
5. Технології створення програмних продуктів та інформаційних систем : навч. посібник / М. Ю. Карпенко, Н. О. Манакова, І. О. Гавриленко ; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – Харків : ХНУМГ ім. О. М. Бекетова, 2017. – 93 с

Інформаційні ресурси Інтернет:

1. The Official Home of the Python Programming Language URL: https://www.python.org
2. Документація Python. URL: https://www.python.org/doc/
3. Документація NumPy. URL: https://numpy.org/doc/stable/
4. Підручник з Python UR. : https://docs.python.org/uk/3/tutorial/index.html
5. Tkinter Documentation. URL：https://wiki.python.org/moin/TkInter

Міжнародні видання:

1. E. Matthers. Python Crash Course. – No Starch Press, 2019. – 544 p.
2. Browning J.B., Alchin M.: Pro Python 3: Features and Tools for Professional Development, 3 rd ed. Apress, 2019. – 458 p.
3. Stephenson B.: The Python Workbook: A Brief Introduction with Exercises and Solutions, 2 nd ed. Springer, 2019. – 219 p.
4. J. Hunt: A Beginners Guide to Python 3 Programming. // Springer, 2019. – 527 p.
5. J. Hunt: Advanced Guide to Python 3 Programming. // Springer, 2019. – 524 p.