МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ПРИКАРПАТСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАСИЛЯ СТЕФАНИКА

Мазуренко В.В., Дмитришин М.І., Василишин П.Б.

ОБ'ЄКТНО–ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ з PYTHON

ЛАБОРАТОРНИЙ ПРАКТИКУМ

Івано-Франківськ — 2023

УДК 004.43
ББК 32.973-018.1
М13

Рекомендовано Вченою радою факультету математики та інформатики Прикарпатського національного університету імені Василя Стефаника (протокол № 10 від 21.11.2023 р.).

Рецензенти:

Олександр МАХНЕЙ, кандидат фізико-математичних наук, доцент (Прикарпатський національний університет імені Василя Стефаника, Івано-Франківськ);

Роман ДМИТРИШИН, доктор фізико-математичних наук, професор (Прикарпатський національний університет імені Василя Стефаника, Івано-Франківськ).

М13 Мазуренко В.В., Дмитришин М.І., Василишин П.Б. Об'єктно–орієнтоване програмування з Python: Лабораторний практикум. – Ів.-Фр.: Голіней, 2023. – 48 с.

У лабораторному практикумі наведено методичні рекомендації до виконання лабораторних робіт з об'єктно-орієнтованого програмування мовою Python.

Для студентів спеціальності «прикладна математика». Практикум буде корисним також для студентів галузей знань «математика і статистика» та «інформаційні технології».

ЗМІСТ

ПЕРЕДМОВА

В об'єктно-орієнтованому стилі програмування (ООП) будь-яку програму розглядають як сукупність «об'єктів», які взаємодіють між собою і кожен з яких є екземпляром якогось «класу». Така парадигма програмування є ефективною при розробці складних програмних проєктів. Програми, написані в об'єктно-орієнтованому стилі, мають більшу модульність і гнучкість, оптимізованість коду і читабельність, захищеність даних і адаптованість до змін.

Лабораторний практикум охоплює важливі аспекти ООП, починаючи від базових понять, таких як класи, об'єкти та атрибути, і завершуючи високорівневими поняттями, такими як абстрактні класи, інтерфейси та шаблони проєктування. Здобувачі освіти мають можливість поглибити свої навики зі створення функцій і методів, використання спеціальних полів і магічних методів, а також розуміння ключових концепцій ООП: абстракції, інкапсуляції, успадкування і поліморфізму. Вивчення шаблонів ООП на основі абстрактних класів та інтерфейсів стимулює творчий підхід до розробки програмного забезпечення та розвиває навики реалізації архітектурно складних програм.

Практикум зорієнтований на використання мови програмування Python. Це високорівнева, інтерпретована мова програмування, яка відзначається простотою синтаксису і читабельністю коду. Її універсальність у поєднанні з численними бібліотеками робить Python ідеальним вибором для широкого спектру завдань — від автоматизації задач і наукових обчислень, аж до інтелектуального аналізу даних і веб-розробки. Водночас варто відзначити, що завдання лабораторного практикуму є інваріантними стосовно мови програмування і за потреби можуть бути використані для вивчення ключових концепцій і практик ООП іншими мовами.

Практикум комбінує індивідуальні лабораторні роботи (для кожної з яких вказана тема, короткий перелік необхідних для виконання роботи теоретичних знань, 15 варіантів завдань та вимоги до оформлення звіту) разом з лабораторними роботами для виконання у команді з двох-чотирьох осіб, що сприяють, зокрема, формуванню м'яких навиків у здобувачів освіти.

.

Лабораторна робота 2

КЛАСИ, ОБ'ЄКТИ, АТРИБУТИ

I. Мета лабораторної роботи

формування навиків створення класів, об'єктів (екземплярів) класу та методів мовою Python

II. Для виконання лабораторної роботи необхідно знати

- що таке клас і об'єкт (екземпляр) класу, шаблон класу
- як ініціалізувати (створити) об'єкт з допомогою конструктора
- як деініціалізувати (знищити) об'єкт з допомогою деструктора
- що таке атрибути (поля і методи) об'єкта і класу
- чим метод відрізняється від функції і для чого призначений вказівник self
- спосіб виклику атрибутів
- як створюються поверхневі і повні копії об'єктів

РЕКОМЕНДОВАНА ЛІТЕРАТУРА: [**2**, п. 6], [**3**, п. 14.1-3], [**4**, п. 1.1], [**5**, п. 10.1-3,6]
ДОДАТКОВА ЛІТЕРАТУРА: [**8, 9**]

III. Завдання на лабораторну роботу

Написати програми для виконання наступних завдань (***номер варіанту є порядковим номером студента в електронному журналі академгрупи***).

Структурою-парою називають клас з двома полями, які зазвичай мають імена `first` і `second`.

У кожному варіанті потрібно:

- реалізувати представлення даних структурою-парою з такими методами:
  – конструктор `__init__` має контролювати значення аргументів на коректність (для перевірки приналежності об'єкта певному класу даних можна використати функцію `isinstance(Object, Class)`, яка повертає логічне значення `True`, в разі якщо `Object` **in** `Class`, або `False`, якщо `Object` **not in** `Class`; для перевірки істинності тверджень можна використовувати інструкцію `assert Condition, Message`, яка генерує повідомлення

`Message`, якщо умова `Condition` є хибною; для обробки винятків можна використовувати інструкцію `try..except..else`) та деструктор `__del__`;

– метод `read` для введення даних користувачем;

– метод `display` для виведення структури даних на екран;

– метод, описаний у відповідному варіанті;

- визначити зовнішню функцію `make`, яка повинна отримувати в якості аргументів значення для полів структури і повертати відповідну структуру.

Варіант 1. Поле `first` — дійсне число, поле `second` — ціле число (показник степеня). Реалізувати метод `power` — піднесення числа `first` до степеня `second`.

Варіант 2. Поле `first` — ціле додатне число (чисельник дробу), поле `second` — ціле додатнє число (знаменник дробу). Реалізувати метод *ipart* — виділення цілої частини дробу $\frac{\mathsf{first}}{\mathsf{second}}$.

Варіант 3. Поле `first` — дійсне додатне число з двома десятковими знаками (ціна товару), поле `second` — ціле додатне число (кількість одиниць товару). Реалізувати метод `cost` — обчислення вартості купленого товару.

Варіант 4. Поле `first` — ціле додатне число (калорійність 100 г продукту), поле `second` — дійсне додатне число (маса продукту). Реалізувати метод `calor` — обчислення загальної калорійності продукту.

Варіант 5. Поле `first` — дійсне число (ліва межа діапазону), поле `second` — дійсне число (права межа діапазону). Реалізувати метод `check` — перевірка заданого числа x на приналежність діапазону.

Варіант 6. Поле `first` — ціле додатне число (години), поле `second` — дійсне додатне число (хаилини). Реалізувати метод `minutes` — переведення часу у хвилини.

Варіант 7. Поле `first` — дійсне число (коефіцієнт k у рівнянні $y = kx + b$), поле `second` — дійсне число (коефіцієнт b). Реалізувати метод `calculate` — обчислення для заданого x значення функції y.

Варіант 8. Поле `first` — дійсне число (координата x точки на площині), поле `second` — дійсне число (координата y точки на площині). Реалізувати метод `distan` — відстань від точки (x, y) до початку координат.

Варіант 9. Поле `first` — дійсне додатне число (катет прямокутного трикутника), поле `second` — дійсне додатне число (гіпотенуза цього трикутника). Реалізувати метод `leg` — обчислення іншого катета трикутника.

Варіант 10. Поле `first` — дійсне число (перший член геометричної прогресії), поле `second` — дійсне число (знаменник геометричної прогресії). Реалізувати метод `evaluate` — обчислення заданого члена геометричної прогресії.

Варіант 11. Поле `first` — дійсне число (перший член арифметичної прогресії), поле `second` — дійсне число (різниця арифметичної прогресії). Реалізувати метод `sum` — обчислення перших n членів арифметичної прогресії.

Варіант 12. Поле `first` — ціле додатне число (n), поле `second` — ціле додатне число (k). Реалізувати метод `combination` — обчислення комбінацій без повторень з n елементів по k.

Варіант 13. Поля `first` та `second` — дійсні додатні числа. Реалізувати метод `hypoten` — обчислення гіпотенузи прямокутного трикутника з катетами `first` та `second`.

Варіант 14. Поле `first` — ціле невід'ємне число (тривалість телефонної розмови у хвилинах), поле `second` — дійсне (з двома десятковими знаками) додатне число (вартість хвилини розмови у гривнях). Реалізувати метод `cost` — обчислення вартості телефонної розмови.

Варіант 15. Поле `first` — ціле додатне число (n), поле `second` — ціле додатне число (k). Реалізувати метод `permutation` — обчислення розміщень без повторень з n елементів по k.

IV. Звіт по лабораторній роботі

Зміст звіту:

1. Титульний аркуш зі стандартними атрибутами
2. Код програми з коментарями
3. Тестування програми
4. Висновки

СПИСОК ЛІТЕРАТУРИ

[1] Будай А. Дизайн-патерни — просто як двері. – Ел. вид., 2012. – Режим доступу: https://sites.google.com/site/designpatternseasy

[2] Васильєв О.М. Програмування мовою Python. – Тернопіль: Навчальна книга — Богдан, 2019.

[3] Висоцька В. А., Оборська О. В. Python: алгоритмізація та програмування: навчальний посібник. – Львів: Новий Світ — 2000, 2021.

[4] Креневич А.П. Python у прикладах і задачах. Частина 2. Об'єктно-орієнтоване програмування: Навчальний посібник. – К.: ВПЦ "Київський університет", 2020.

[5] Мізюк О. Путівник мовою програмування Python. – Ел. вид., 2020. – Режим доступу: https://pythonguide.rozh2sch.org.ua

[6] Швець О. Занурення в патерни проектування. – Ел. вид., 2021. – Режим доступу: https://refactoring.guru/uk/design-patterns/book

[7] Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns. Elements of Reusable Object-Oriented Software. – Addison-Wesley Professional, 1994.

[8] PEP 8 — Style Guide for Python Code. – Access mode: https://peps.python.org/pep-0008/#source-file-encoding

[9] The Python Tutorial. – Access mode: https://docs.python.org/3/tutorial/index.html