Міністерство освіти і науки України
Національний університет „Львівська політехніка”
Кафедра ЕОМ

Звіт

до лабораторної роботи № 9

з дисципліни: «Кросплатформні засоби програмування»

На тему: «Основи об’єктно-орієнтованого програмування у Python»

Виконав:
Студент групи КІ-305
Вознюк О. М.
Прийняв:
Іванов Ю. С.

Львів 2023

Мета роботи: оволодіти навиками реалізації парадигм об’єктно-орієнтованого програмування використовуючи засоби мови Python.

ЗАВДАННЯ

1. Написати та налагодити програму на мові Python згідно варіанту. Програма має задовольняти наступним вимогам:
   - класи програми мають розміщуватися в окремих модулях в одному пакеті;
   - точка входу в програму (main) має бути в окремому модулі;
   - мають бути реалізовані базовий і похідний класи предметної області згідно варіанту;
   - програма має містити коментарі.
2. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.
4. Дати відповідь на контрольні запитання.

Варіант завдання:

Базовий клас: Людина

Похідний клас: Спортсмен

Код програми:

Person.py:

```
class Person:
    def __init__(self, name, age):
        """Initialize a person with a name and age."""
        self.name = name
        self.age = age


    def introduce(self):
        """Return a person's introduction."""
        return f"Hello, my name is {self.name} and I am {self.age} years old."
```

Sporsman.py:

```
from Person import Person

class Sporsman(Person):
    def __init__(self, name, age, sport):
        """Initialize an athlete with a name, age, and sport. Inherits from the
Person class."""
        super().__init__(name, age)
        self.sport = sport
        self.training_hours = 0

    def train(self, hours):
        """Simulate athlete training and increase training hours."""
        self.training_hours += hours
        return f"{self.name} trained for {hours} hours. Total training hours:
{self.training_hours}"
```

```
    def compete(self):
        """Simulate athlete competing and decrease energy level."""
        if self.training_hours >= 10:
            self.training_hours -= 10
            return f"{self.name} successfully competed. Total training hours:
{self.training_hours}"
        else:
            return f"{self.name} needs more training to compete."


if __name__ == "__main__":
    # Create a regular person
    person1 = Person("John", 30)
    print(person1.introduce())

    # Create an athlete
    athlete1 = Athlete("Alice", 25, "Swimming")
    print(f"{athlete1.name} is an athlete in {athlete1.sport}.")
    print(athlete1.train(5))
    print(athlete1.compete())
```

main.py:

```
from Person import Person
from Sporsman import Sporsman

if __name__ == "__main__":
    # Create a regular person
    person1 = Person("John", 30)
    print(person1.introduce())

    # Create an athlete
    sportsman1 = (Sporsman("Alice", 25, "Swimming"))
    print(f"{sportsman1.name} is an athlete in {sportsman1.sport}.")
    print(sportsman1.train(5))
    print(sportsman1.compete())
```

Результати роботи програми:

```
Hello, my name is John and I am 30 years old.
Alice is an athlete in Swimming.
Alice trained for 5 hours. Total training hours: 5
Alice needs more training to compete.
```

Відповіді на контрольні запитання

1. Що таке модулі?

- Модулі в Python - це файли, які містять Python-код. Вони використовуються для організації коду у логічні групи, і можуть містити функції, класи, змінні та інші об'єкти.

2. Як імпортувати модуль? - import модуль
3. Як оголосити клас?

- class МійКлас:

    # Тіло класу

4. Що може міститися у класі?
   - атрибути (змінні), методи (функції), конструктори, спеціальні методи (наприклад,__init__,__str__), властивості та інше.
5. Як називається конструктор класу?
   - Конструктор класу має ім'я___init__. Він викликається при створенні нового об'єкта класу і використовується для ініціалізації атрибутів об'єкта.

6. Як здійснити спадкування?
   - class ПідКлас(БазовийКлас):

   # Тіло підкласу

7. Які види спадкування існують?
   - одиночне спадкування (коли підклас успадковує лише один базовий клас) та множинне спадкування (коли підклас успадковує більше одного базового класу).
8. Які небезпеки є при множинному спадкуванні, як їх уникнути?
   - Небезпеки при множинному спадкуванні включають в себе можливі конфлікти імен методів або атрибутів між базовими класами, що може призвести до непередбачуваної поведінки. Для уникнення цих проблем можна використовувати аліаси, викликати методи базових класів безпосередньо або використовувати композицію замість спадкування.
9. Що таке класи-домішки?
   - це класи, які містять певний функціонал і можуть бути використані для розширення функціональності інших класів. Вони не призначені для створення об'єктів, але можуть бути включені у інші класи за допомогою спадкування, щоб надати їм певну функціональність.
10. Яка роль функції super() при спадкуванні?
    - для виклику методів базового класу з підкласу. Вона допомагає уникнути явного вказівання імен базових класів та робить код більш гнучким при зміні структури спадкування. Наприклад, super().__init__() викликає конструктор базового класу.

Висновок

У ході виконання даної лабораторної роботи, я здобула важливі навички об'єктно-орієнтованого програмування мовою Python. Ознайомилась з ключовими аспектами цієї парадигми, включаючи створення та використання класів, роботу з об'єктами, та використання спадкування та поліморфізму для покращення ефективності програм.