Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра «Електронних обчислювальних машин»

Звіт
з лабораторної роботи № 9
з дисципліни: «Кросплатформенні засоби програмування»
на тему: «Основи Об'єктно-Орієнтованого програмування у Python»

Виконав:
студент групи КІ-306
Гапонова Дарина
Прийняв:
доцент кафедри ЕОМ
Іванов Ю. С.

Львів – 2023

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

Завдання (варіант № 29)

4. Кіт

4. Піддослідний кіт

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.

Вихідний код програми

Файл Main.py

```
from experimental_cat import ExperimentalCat

def main():
    my_cat = ExperimentalCat("Nomi", 2, "Common")

    my_cat.display_info()

    my_cat.set_name("Fluffy")
    my_cat.set_age(4)
    my_cat.set_breed("Persian")

    print("new cat info:")

    my_cat.display_info()

    my_cat.meow()
    my_cat.run()
    my_cat.jump()
    my_cat.purr()
    my_cat.sleep()
    my_cat.eat("tuna")
    my_cat.play("a ball")
```

```
    print(my_cat.get_subject_type())

if __name__ == "__main__":
    main()
```

Файл cat.py

```
import os

class Cat:
    def __init__(self, name="", age=0, breed=""):
        self.name = name
        self.age = age
        self.breed = breed

    def set_name(self, name):
        self.name = name

    def set_age(self, age):
        self.age = age

    def set_breed(self, breed):
        self.breed = breed

    def get_name(self):
        return self.name

    def get_age(self):
        return self.age

    def get_breed(self):
        return self.breed

    def meow(self):
        print(self.name + " says: Meow!")

    def run(self):
        print(self.name + " is running.")

    def jump(self):
        print(self.name + " jumped.")

    def purr(self):
        print(self.name + " is purring.")

    def sleep(self):
        print(self.name + " is sleeping.")

    def eat(self, food):
        print(self.name + " is eating " + food + ".")
```

```
    def play(self, toy):
        print(self.name + " is playing with " + toy + ".")

    def display_info(self):
        print("Name:", self.name)
        print("Age:", self.age)
        print("Breed:", self.breed)
```

Файл experimental_cat.py

```
from cat import Cat

class ExperimentalCat(Cat):
    def __init__(self, name, age, breed):
        super().__init__(name, age, breed)

    def get_subject_type(self):
        return "Experimantal cat"
```

Результат виконання програми

```
Name: Nomi
Age: 2
Breed: Common
new cat info:
Name: Fluffy
Age: 4
Breed: Persian
Fluffy says: Meow!
Fluffy is running.
Fluffy jumped.
Fluffy is purring.
Fluffy is sleeping.
Fluffy is eating tuna.
Fluffy is playing with a ball.
Experimantal cat
```

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.