

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет ім. М.Є. Жуковського
«Харківський авіаційний інститут»

Кафедра систем управління літальних апаратів

Лабораторна робота №4

з дисципліни «Об'єктно-орієнтоване проектування систем авіоніки»

Тема: «Реалізація класу і робота з об'єктами»

XAI.301.173.320.4ЛР

Виконав студент гр. 320

Черватюк В.О.

Перевірив

к.т.н., доц. О.В.Гавриленко

ас. В. О. Білозерський

МЕТА РОБОТИ

Застосувати теоретичні знання з основ програмування на мові Python з використанням об'єктів і класів, навички використання бібліотеки для візуалізації масивів даних, і навчитися розробляти скрипти для роботи з об'єктами призначених для користувача класів.

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Визначити клас `Point_n`, який реалізує абстракцію з атрибутами:

1) дві дійсні координати точки на площині (властивості, приховані змінні екземпляра),

– для кожної метод-геттер (повертає відповідну координату),

– для кожної метод-сеттер (записує відповідну координату, якщо вона у межах $[-100, 100]$, інакше – дорівнює 0))

2) кількість створених екземплярів точки (змінна класу),

3) метод класу (повертає кількість створених примірників),

4) конструктор з двома параметрами (за замовчуванням),

5) деструктор, що виводить відповідне повідомлення,

6) метод, що змінює координати точки з двома вхідними дійсними параметрами:

– зсув по x ,

– зсув по y .

Завдання 2. Виконати операції з об'єктами даного класу відповідно до варіанту.

19.	Створити список з трьох точок, порахувати відстань між першою і другою, пересунути третю на 12 вниз і на 23 вліво.
------------	--

Завдання 3. Використовуючи пакет `matplotlib`, відобразити створені об'єкти в графічному вікні до і після змін.

Завдання 4. Зберегти координати точок у текстовому файлі у форматі: номер: координата_х; координата_y – для непарних варіантів

ВИКОНАННЯ РОБОТИ

Завдання 1. Вирішення завдання варіанту 19

Вхідні дані (ім'я, опис, тип, обмеження):

1) створення класу Point_19

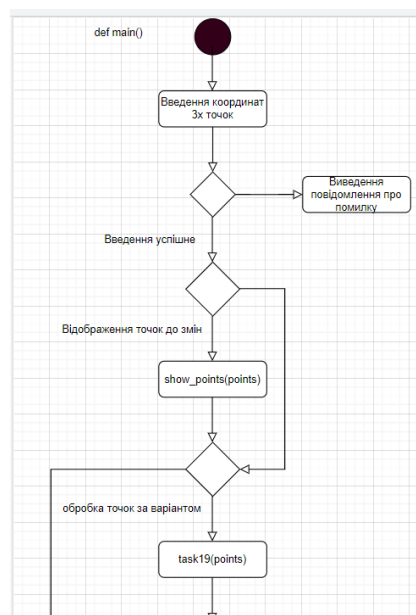
Вихідні дані (ім'я, опис, тип):

1) задані точки, змінені точки

Алгоритм вирішення показано на рис.1-2

Point_19
+name : string
-set_x : float
-set_y : float
<u>+point_19_count: Integer</u>
<u>-point_19_count : Integer</u>
+set_x() : float
+set_y() : float
+set_x(float)
+set_y(float)
+getCount()
<u>+savePoints()</u>

Рисунок 1 – Діаграма класів функції завдання варіанту 19



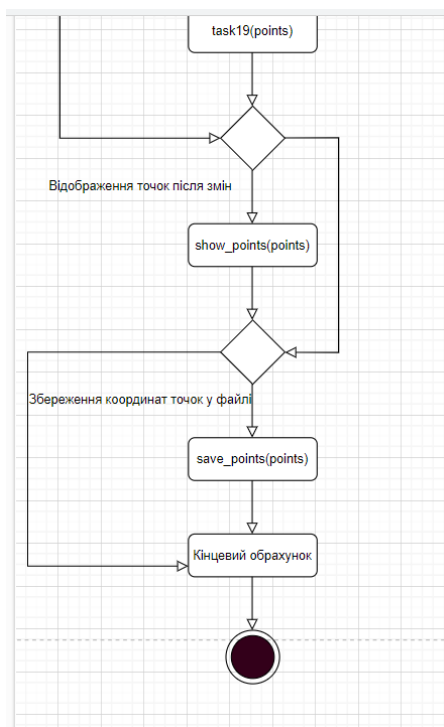


Рисунок 2 – Діаграма активності функції завдання варіанту 19

Лістинг коду вирішення задачі наведено в дод. А (стор. 5-7). Екран роботи програми наведено в дод. Б (стор. 8-9)

ВИСНОВКИ

Були закріплені теоретичні знання з основ програмування на мові Python з використанням об'єктів і класів, навички використання бібліотеки для візуалізації масивів даних. Навчилися розробляти скрипти для роботи з об'єктами призначених для користувача класів.

ДОДАТОК А

Лістинг коду програми до завдання варіанту 19

Point_19.py

```
class Point_19:
    """
    A class to represent a point in 2D space"""

    __x: float = 0.0
    __y: float = 0.0
    __point_count: int = 0

    def __init__(self, x: float, y: float):
        self.set_x(x)
        self.set_y(y)
        Point_19.__point_count += 1

    def __del__(self):
        print("Point has been deleted")
        Point_19.__point_count -= 1

    def get_x(self):
        return self.__x

    def set_x(self, value):
        if value <= 100 and value >= -100:
            self.__x = value
        else:
            self.__x = 0.0

    def set_y(self, value):
        if value <= 100 and value >= -100:
            self.__y = value
        else:
            self.__y = 0.0

    def get_y(self):
        return self.__y

    def shift(self, x_shift: float, y_shift: float):
        self.set_x(self.get_x() + x_shift)
        self.set_y(self.get_y() + y_shift)

    @staticmethod
    def get_count():
        return Point_19.__point_count
```

main.py

```

import matplotlib.pyplot as plt
from Point_19 import Point_19
import math

# Основна функція
def main():
    # пустий масив точок
    points = []
    # Вводимо координати точок
    # та створюємо екземпляри класу Point_1
    print("Enter X,Y values for 3 points:")
    for i in range(3):
        try:
            tmp_x = float(input("X{}: ".format(i+1)))
            tmp_y = float(input("Y{}: ".format(i + 1)))
        except ValueError:
            print("Wrong values for points!")
            exit()
        else:
            tmp_point = Point_19(tmp_x, tmp_y)
            print(Point_19.get_count())
            points.append(tmp_point)
    # Відображення точок до змін
    show_points(points)
    # Обробка точок за варіантом
    task19(points)
    # Відображення точок після змін
    show_points(points)
    # Збереження координат точок у файлі
    save_points(points)

# Функція для обробки точок за варіантом
def task19(list_of3_points):
    """Створити список з трьох точок, порахувати відстань між першою і
    другою, пересунути третю на 12 вниз і на 23 вліво."""
    point_1 = list_of3_points[0]
    point_2 = list_of3_points[1]
    point_3 = list_of3_points[2]
    length = math.sqrt(math.pow(point_2.get_x() - point_1.get_x(), 2) +
    math.pow(point_2.get_y() - point_1.get_y(), 2))
    point_3.shift(-23.0, -12.0)
    print("Length = {}".format(length))

# відображення графічних об'єктів

```

```
def show_points(list_of_points):  
    # work with plot  
    x = [point.get_x() for point in list_of_points]  
    y = [point.get_y() for point in list_of_points]  
    plt.plot(x, y, 'ro')  
    plt.grid()  
    plt.show()  
  
# збереження координат у файлу  
def save_points(list_of_points):  
    with open("output.txt", "w") as f:  
        for num, point in enumerate(list_of_points): # 0: point1, 1: point2, 2:  
point3  
            f.write(f"{num+1}: {point.get_x()}; {point.get_y()}\n")  
  
if __name__ == '__main__':  
    main()
```

ДОДАТОК Б

Скрін-шоти вікна виконання програми

```
Enter X,Y values for 3 points:  
X1: 1  
Y1: 1  
1  
X2: 1.2  
Y2: 2.2  
2  
X3: 2.1  
Y3: 2.3  
3  
Length = 1.216552506059644  
█
```

Рисунок Б.1 – Екран виконання програми для вирішення завдання варіанту 19

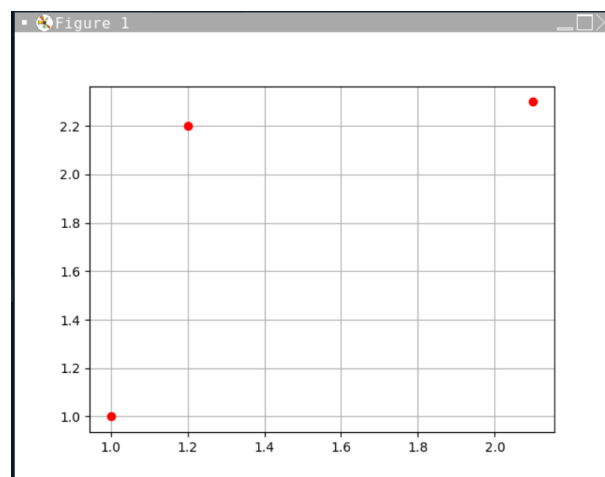


Рисунок Б.2 – Нанесення точок, заданих в консоль

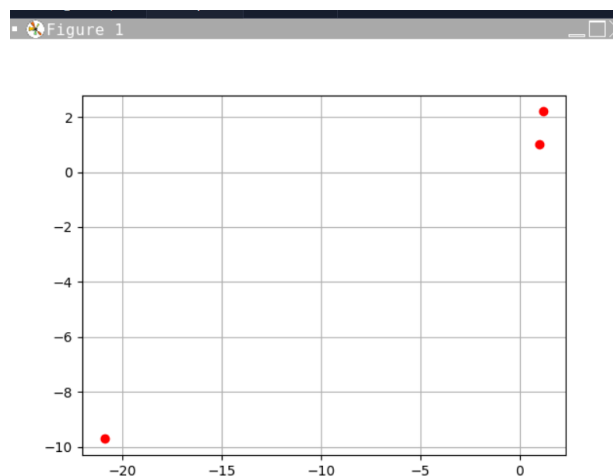
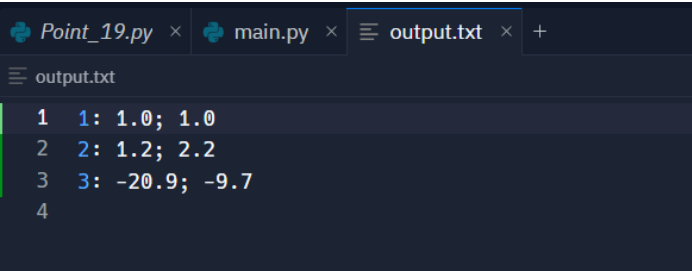


Рисунок Б.3 – Змінені точки



```
Point_19.py x main.py x output.txt x +
output.txt
1 1: 1.0; 1.0
2 2: 1.2; 2.2
3 3: -20.9; -9.7
4
```

Рисунок Б.4 – Запис змінених точок до файлу