

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

Кафедра систем управління літальними апаратами

Лабораторна робота № 4

з дисципліни «Об'єктно-орієнтоване програмування СУ»

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

ХАІ.301 .173. 310ст.4 ЛР

Виконав студент гр. _____310ст_____

_____Возвишаєв Олексій Андрійович
(підпис, дата) (П.І.Б.)

Перевірив

_____к.т.н., доц. О. В. Гавриленко
_____ас. В. О. Білозерський
(підпис, дата) (П.І.Б.)

2024

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_1`, 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 (див. таб.1).

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 (номер) координата_х:координата_y – для парних варіантів

ВИКОНАННЯ РОБОТИ

Завдання 1. Вирішення задачі 1

1.	Створити список з трьох точок, порахувати відстань між першою і другою, пересунути третю на 10 вліво.
-----------	---

Вхідні дані:

x (int): координата точки по осі X, допустимий діапазон [-100, 100].

y (int): координата точки по осі Y, допустимий діапазон [-100, 100].

dx (int): зміщення по осі X, обмежень немає.

dy (int): зміщення по осі Y, обмежень немає.

Вихідні дані

distance (float): відстань між двома точками.

x_coords_before (list[int]): список координат X точок до переміщення.

y_coords_before (list[int]): список координат Y точок до переміщення.

x_coords_after (list[int]): список координат X точок після переміщення.

y_coords_after (list[int]): список координат Y точок після переміщення.

Діаграма класів показано на рис.1

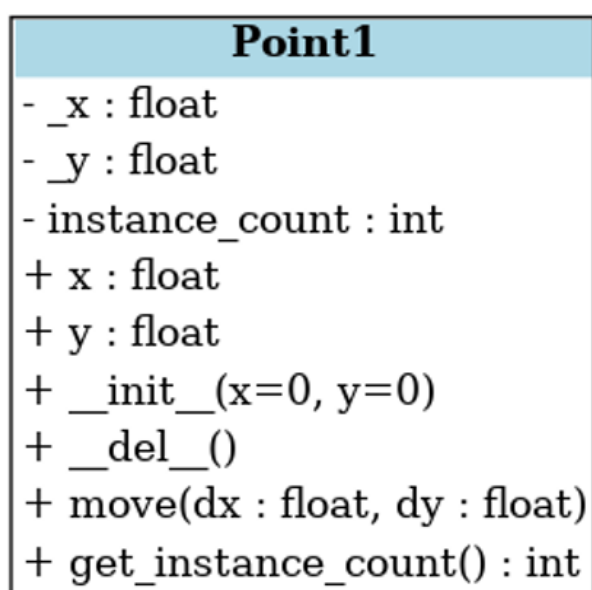


Рисунок 1 – Діаграма класів

Діаграма активності показано на рис.1.1



Рисунок 1.1 - Діаграма активності

Лістинг коду вирішення задачі наведено в дод. А стор.5. Екран роботи програми показаний на рис. Б.8.

ВИСНОВКИ

У ході виконання роботи на практиці закріплено вміння створювати класи, їхні методи та атрибути, а також працювати з бібліотекою `matplotlib` для візуалізації даних. Отримано навички маніпулювання даними, збереження інформації у текстовий файл та реалізації абстракцій об'єктів.

ДОДАТОК А

Лістинг коду програми до задачі Point_1

```

import math
import matplotlib.pyplot as plt

class Point1:
    """Клас, який представляє точку на площині."""
    instance_count = 0 # Лічильник екземплярів класу

    def __init__(self, x=0, y=0):
        """Ініціалізує координати точки та збільшує лічильник екземплярів."""
        self._x = self._validate_coordinate(x)
        self._y = self._validate_coordinate(y)
        Point1.instance_count += 1

    def __del__(self):
        """Зменшує лічильник екземплярів при видаленні об'єкта."""
        Point1.instance_count -= 1
        print(f"Точка з координатами ({self._x}, {self._y}) видалена.")

    @staticmethod
    def _validate_coordinate(value):
        """Перевірка, чи знаходиться значення в діапазоні [-100, 100]."""
        return value if -100 <= value <= 100 else 0

    @property
    def x(self):
        """Повертає координату x."""
        return self._x

    @x.setter
    def x(self, value):
        """Встановлює координату x з перевіркою."""
        self._x = self._validate_coordinate(value)

    @property
    def y(self):
        """Повертає координату y."""
        return self._y

    @y.setter
    def y(self, value):
        """Встановлює координату y з перевіркою."""
        self._y = self._validate_coordinate(value)

    def move(self, dx, dy):
        """Зміщує координати точки на dx і dy."""
        self._x = self._validate_coordinate(self._x + dx)
        self._y = self._validate_coordinate(self._y + dy)

```

```

def __repr__(self):
    """Рядкове представлення об'єкта."""
    return f"Point1(x={self._x}, y={self._y})"

# Створюємо три точки
points = [Point1(10, 20), Point1(-50, 150), Point1(0, 0)]

# Визначаємо відстань між першою і другою точками
distance = math.sqrt((points[0].x - points[1].x) ** 2 + (points[0].y -
points[1].y) ** 2)
print(f"Відстань між першою і другою точками: {distance:.2f}")

# Список координат точок до переміщення
x_coords_before = [point.x for point in points]
y_coords_before = [point.y for point in points]

# Переміщуємо третю точку двічі на 10 вліво
points[2].move(-10, 0)
points[2].move(-10, 0)

# Список координат точок після переміщення
x_coords_after = [point.x for point in points]
y_coords_after = [point.y for point in points]

# Створюємо графік
plt.figure(figsize=(8, 6))

# Малюємо точки до переміщення (зелені)
plt.scatter(x_coords_before, y_coords_before, color="green", label="До зміни")
for i, point in enumerate(points):
    plt.text(x_coords_before[i], y_coords_before[i], f"P{i+1}", color="green")

# Малюємо точки після переміщення (яскраво-розові)
plt.scatter(x_coords_after, y_coords_after, color="hotpink", label="Після
зміни")
for i, point in enumerate(points):
    plt.text(x_coords_after[i], y_coords_after[i], f"P{i+1}", color="hotpink")

plt.axhline(0, color="black", linewidth=0.5) # Горизонтальна лінія
plt.axvline(0, color="black", linewidth=0.5) # Вертикальна лінія
plt.grid(color="gray", linestyle="--", linewidth=0.5) # Сітка
plt.legend()
plt.title("Зміна координат точок")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# Зберігаємо дані точок у файл
with open("points_data.txt", "w") as file:
    for i, point in enumerate(points, start=1):

```

```
file.write(f"{i} {point.x}:{point.y}\n")  
  
print("Дані точок збережені у файл 'points_data.txt'.")
```

ДОДАТОК Б

Скрін-шоти вікна виконання програми

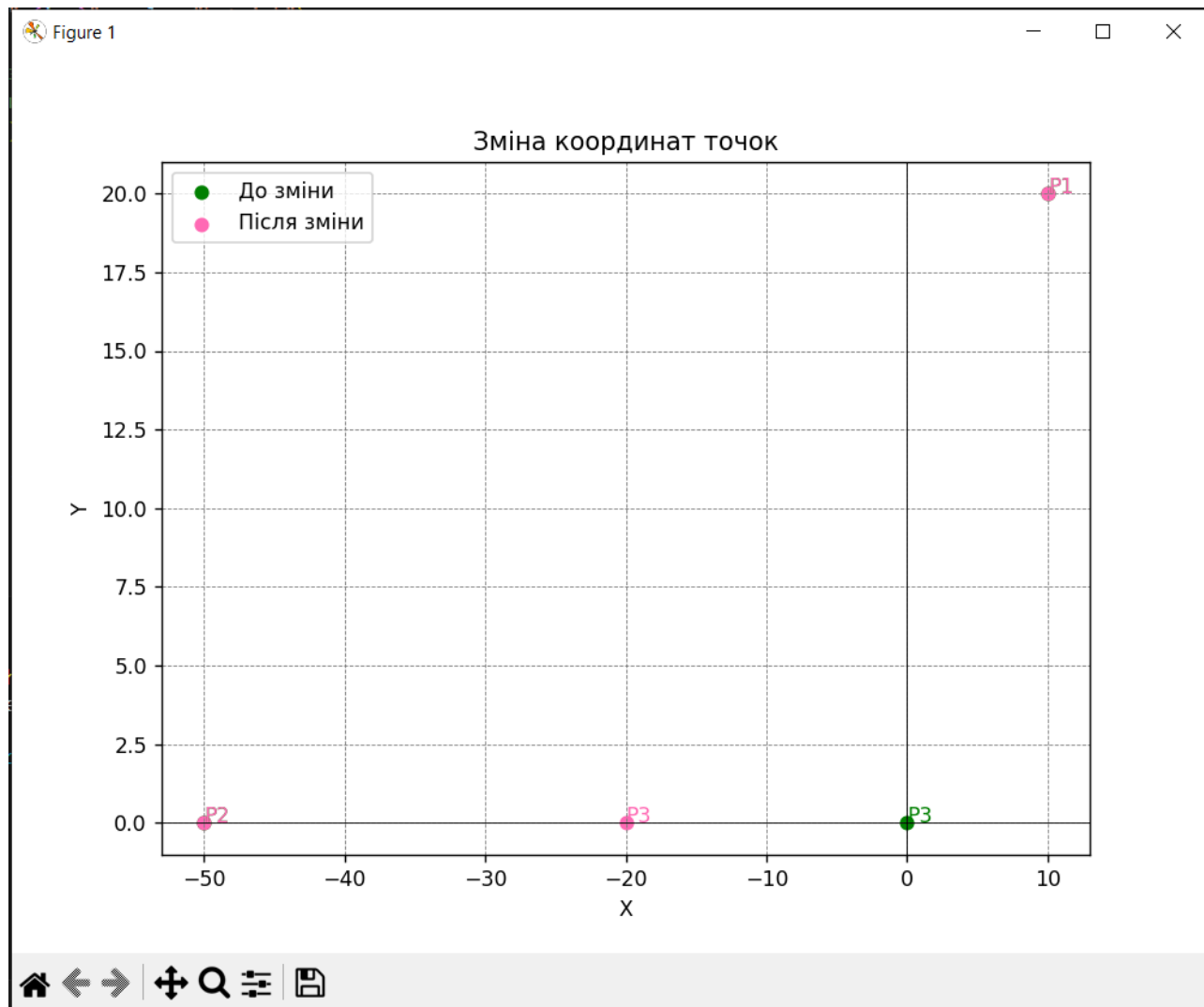
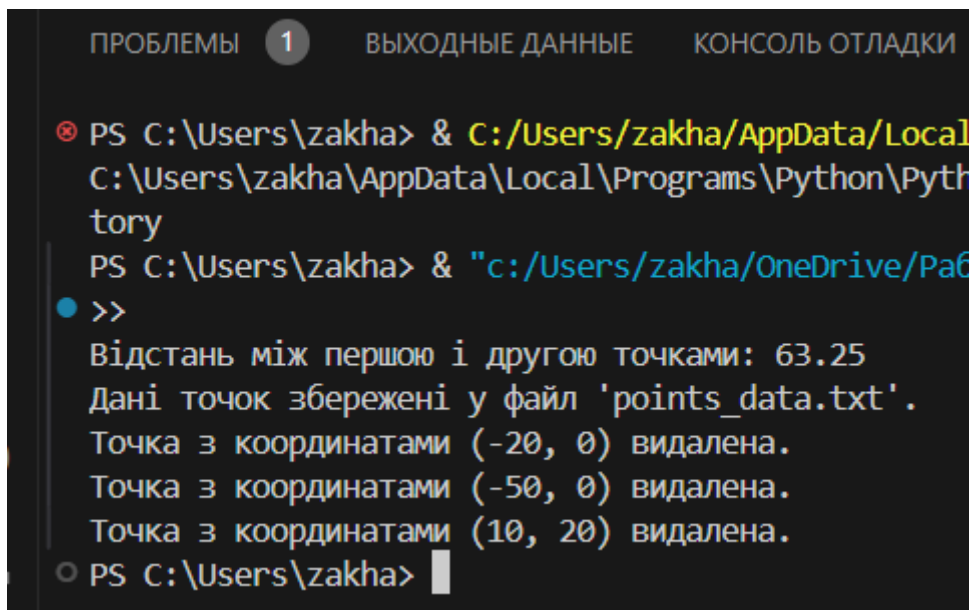
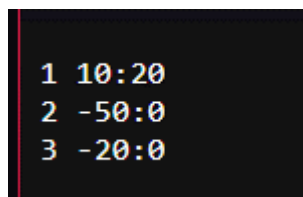


Рисунок Б.1 – Екран виконання програми для вирішення завдання Point_1



```
ПРОБЛЕМЫ 1 ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ  
PS C:\Users\zakha> & C:/Users/zakha/AppData/Local/Programs/Python/Python39-32/python.exe C:/Users/zakha/AppData/Local/Programs/Python/Python39-32/Scripts/python.exe C:/Users/zakha/OneDrive/Pa6  
PS C:\Users\zakha> & "c:/Users/zakha/OneDrive/Pa6  
.>>  
Відстань між першою і другою точками: 63.25  
Дані точок збережені у файл 'points_data.txt'.  
Точка з координатами (-20, 0) видалена.  
Точка з координатами (-50, 0) видалена.  
Точка з координатами (10, 20) видалена.  
PS C:\Users\zakha>
```

Рисунок Б.2 – Екран виконання програми для вирішення завдання Point_1



```
1 10:20  
2 -50:0  
3 -20:0
```

Рисунок Б.3 – Екран виконання програми для вирішення завдання Point_1