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 .174. 312ст.4 ЛР

Виконав студент гр. _____312ст_____

_____Твердохліб Максим Анатолійович
(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_x; координата_y – для непарних варіантів (номер) координата_x:координата_y – для парних варіантів

Виконання роботи

Вирішення завдання 1

17.	Створити список з трьох точок, порахувати відстань між другою і третьою, пересунути першу на 50 вгору.

Вхідні дані:

points (list[tuple[int, int]]): список із трьох точок, кожна з яких задана координатами x і y.

Координати x та y — цілі числа, допустимий діапазон значень: [-100, 100].

Вихідні дані:

distance (float): Евклідова відстань між другою та третьою 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 представлена на рис. 1

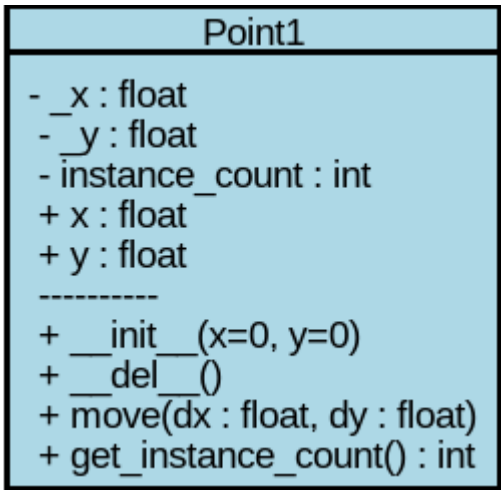

Рисунок 1 – діаграма класів

Рисунок 1.1 – Діаграма активності

Лістинг коду вирішення задачі показано в додатку А стор. Екран роботи наведено у додатку Б. стор.

Висновки:

Під час виконання роботи ми практично закріпили навички створення класів, методів та атрибутів, а також навчились використовувати бібліотеку matplotlib для візуалізації даних. Ми освоїли маніпуляції з даними, збереження інформації у текстовий файл та реалізацію об'єктних абстракцій.

Додаток А

Лістинг коду програму до задачі

```
import math
import matplotlib.pyplot as plt


class Point:
    """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)
        Point.instance_count += 1

    def __del__(self):
        """Зменшує лічильник екземплярів при видаленні об'єкта."""
        Point.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"Point(x={self._x}, y={self._y})"

# Створюємо три точки
points = [Point(10, 20), Point(-50, 150), Point(30, -40)]

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

# Переміщаємо першу точку на 50 вгору
points[0].move(0, 50)

# Список координат точок до і після переміщення
x_coords = [point.x for point in points]
y_coords = [point.y for point in points]

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

```
# Малюємо точки (червоні)
plt.scatter(x_coords, y_coords, color="red", label="Точки")
for i, point in enumerate(points):
    plt.text(x_coords[i], y_coords[i], f"P{i+1}", color="red")

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("Координати точок")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# Зберігаємо дані точок у файл
with open("updated_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 'updated_points_data.txt'.")
```

Додаток Б

Екран роботи вікна виконання програми

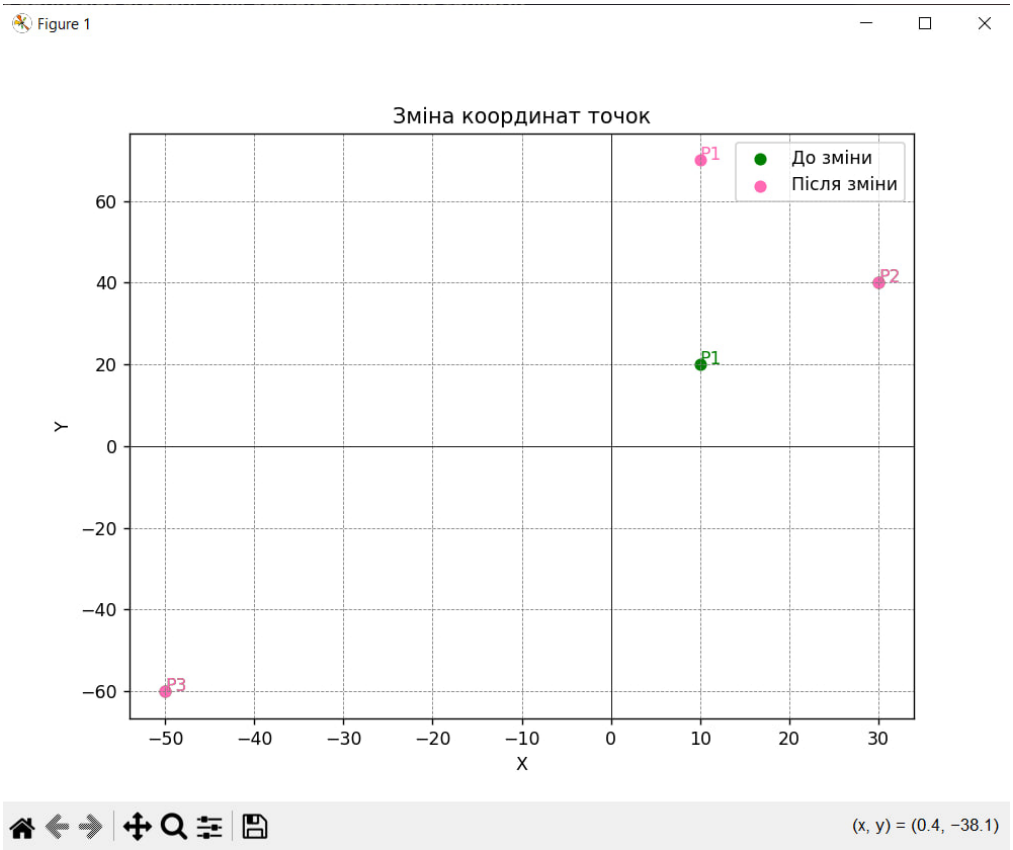

Рисунок 2 – Екран виконання програми до завдання

```
PS C:\Users\cpire> & C:/Users/cpire/AppData/Local/Programs/Python/Python313/python.exe
PS C:\Users\cpire> & C:/Users/cpire/AppData/Local/Programs/Python/Python313/python.exe
PS C:\Users\cpire> & C:/Users/cpire/AppData/Local/Programs/Python/Python313/python.exe
Відстань між другою і третьою точками: 128.06
```

Рисунок 2.1 – Екран виконання програми до завдання