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.301.172.519.СТ.5

Виконав студент гр. _____519СТ_____

_____Пащенко Нікіта___
(підпис, дата) (П.І.Б.)

Перевірив
_____ к.т.н., доц. О. В. Гавриленко
(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_14, 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 (див. рис.1).

| | |
|---|---|
| **14.** | Створити список з чотирьох точок, порахувати відстань між першою і четвертою, пересунути другу на 43 вгору. |

Рис 1 - Завдання 2

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

Завдання 1. Вирішення задачі Point_14

Вхідні дані:

x: Координата точки по осі X (тип: float, обмеження: [-100, 100]).

y: Координата точки по осі Y (тип: float, обмеження: [-100, 100]).

dx: Зсув точки по осі X (тип: float, обмеження: немає).

dy: Зсув точки по осі Y (тип: float, обмеження: немає).

Вихідні дані:

distance: Відстань між першою та четвертою точками (тип: float).

new_coords: Нові координати точки після зсуву (тип: tuple).

output_file: Файл із координатами точок після змін (тип: string).

Алгоритм вирішення:

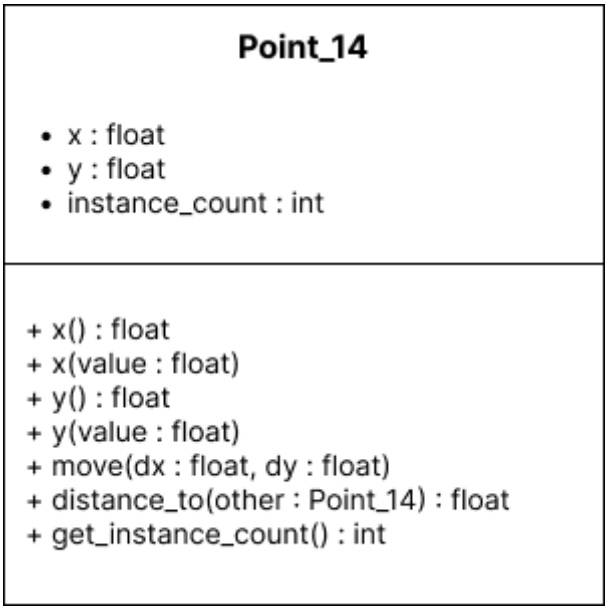


Рисунок 1 - Блок схема до Завдання 1

Завдання 2. Вирішення задачі побудова графіка за допомогою matplotlib

Вхідні дані:

points: Список із координат точок (тип: list, обмеження: довжина списку = 4).

Вихідні дані:

graph_before: Графік координат точок до змін (тип: matplotlib object).

graph_after: Графік координат точок після змін (тип: matplotlib object).

Алгоритм вирішення:

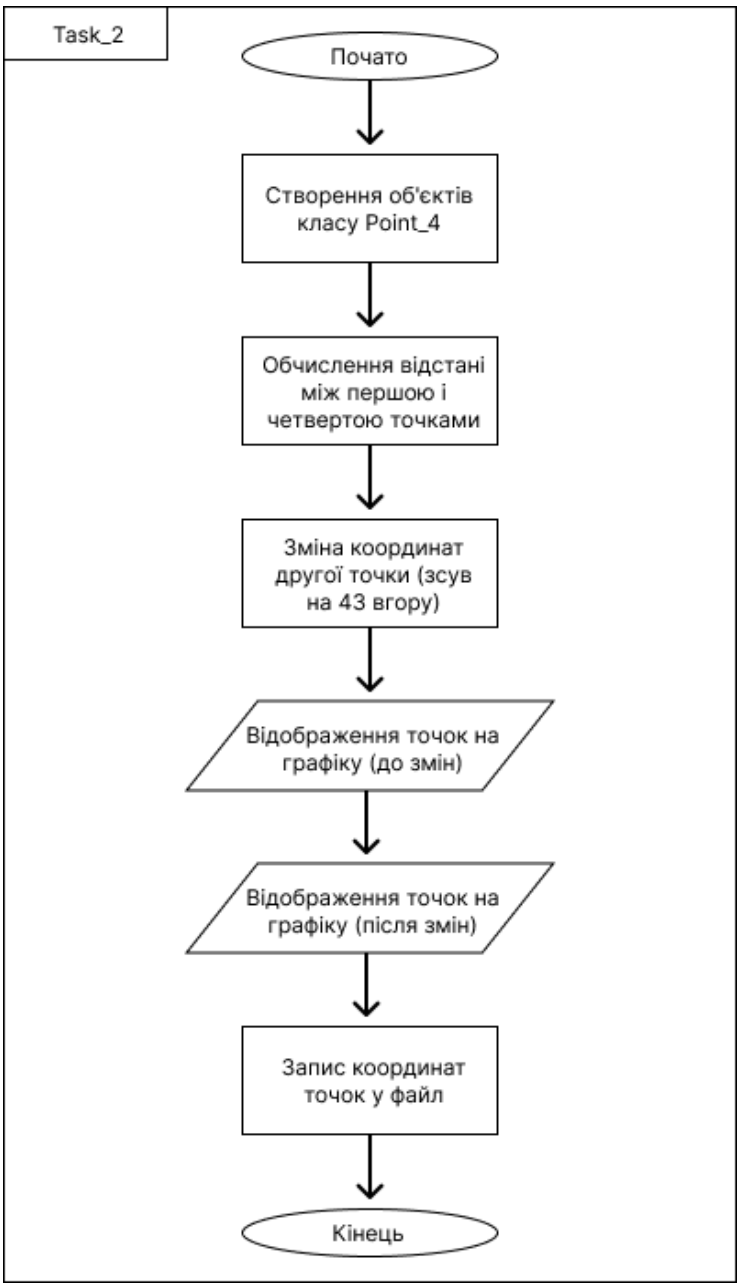


Рисунок 2 - Блок-схема до Завдання 2

Лістинг коду вирішення задачі наведено в дод. А (стор. 6). Екран роботи програми показаний на рис. Б. (стор. 9).

Завдання 3. Вирішення задачі Point_14

Вхідні дані:

points: Список із координат точок (тип: list, обмеження: довжина списку = 4).

filename: Ім'я файлу для запису (тип: string, обмеження: розширення .txt).

Вихідні дані:

output_file: Файл із записаними координатами (тип: string).

# ВИСНОВКИ

У процесі виконання завдання було створено клас Point_14, який реалізує роботу з точками на площині, включаючи зсув координат, обчислення відстаней і відображення результатів на графіку. Основні операції успішно реалізовані та результати записані у файл згідно з вимогами. Завдання демонструє практичне застосування ООП і бібліотеки matplotlib для роботи з графічними об'єктами.

# ДОДАТОК А
## Лістинг коду програми до задачі Proc20

```
<
   import math
import matplotlib.pyplot as plt

class Point_4:
    # Змінна класу для підрахунку екземплярів
    instance_count = 0

    def __init__(self, x=0.0, y=0.0):
        # Використання сеттерів для ініціалізації координат
        self.x = x
        self.y = y
        Point_4.instance_count += 1

    # Деструктор
    def __del__(self):
        Point_4.instance_count -= 1
        print(f"Точка ({self.__x}, {self.__y}) знищена.")

    # Геттери
    @property
    def x(self):
        return self.__x

    @property
    def y(self):
        return self.__y

    # Сеттери
    @x.setter
    def x(self, value):
        self.__x = value if -100 <= value <= 100 else 0

    @y.setter
    def y(self, value):
        self.__y = value if -100 <= value <= 100 else 0

    # Метод класу для отримання кількості створених екземплярів
    @classmethod
    def get_instance_count(cls):
        return cls.instance_count

    # Метод для зсуву точки
    def move(self, dx, dy):
        self.x += dx
        self.y += dy
```

```
    # Метод для обчислення відстані між точками
    def distance_to(self, other):
        return math.sqrt((self.x - other.x) ** 2 + (self.y - other.y) ** 2)


# Завдання 2: Створення точок і виконання операцій
# Створення списку точок
points = [Point_4(10, 20), Point_4(30, 40), Point_4(-50, -60), Point_4(70, 80)]

# Збереження початкових координат
x_coords_before = [p.x for p in points]
y_coords_before = [p.y for p in points]

# Відстань між першою та четвертою точками
distance = points[0].distance_to(points[3])
print(f"Відстань між першою і четвертою точками: {distance:.2f}")

# Пересуваємо другу точку на 43 одиниці вгору
points[1].move(0, 43)
print(f"Нова координата другої точки: ({points[1].x}, {points[1].y})")

# Збереження координат після змін
x_coords_after = [p.x for p in points]
y_coords_after = [p.y for p in points]

# Завдання 3: Відображення об'єктів у графічному вікні
plt.figure(figsize=(12, 6))

# Графік "до змін"
plt.subplot(1, 2, 1)
plt.scatter(x_coords_before, y_coords_before, color='blue', label='До змін')
for i, (x, y) in enumerate(zip(x_coords_before, y_coords_before), start=1):
    plt.text(x + 2, y + 2, f"{i}")
plt.title('До змін')
plt.xlabel('X')
plt.ylabel('Y')
plt.grid()
plt.legend()

# Графік "після змін"
plt.subplot(1, 2, 2)
plt.scatter(x_coords_after, y_coords_after, color='red', label='Після змін')
for i, (x, y) in enumerate(zip(x_coords_after, y_coords_after), start=1):
    plt.text(x + 2, y + 2, f"{i}")
plt.title('Після змін')
plt.xlabel('X')
plt.ylabel('Y')
plt.grid()
plt.legend()

plt.tight_layout()
```

```
plt.show()

# Завдання 4: Збереження координат у файл
with open("points_output.txt", "w") as file:
    for i, point in enumerate(points, start=1):
        file.write(f"{i}) {point.x}:{point.y}\n")

print("Координати точок збережено у файл 'points_output.txt'.")

>
```

# ДОДАТОК Б

## Скрін-шоти вікна виконання програми

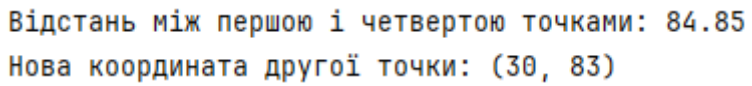


Рисунок Б.1 – Екран виконання програми для вирішення завдання 1. Point_14