Чернівецький національний університет імені Юрія Федьковича
Навчально-науковий інститут фізико-технічних та комп'ютерних наук
Відділ комп'ютерних технологій
Кафедра математичних проблем управління і кібернетики

Звіт
про виконання лабораторної роботи №5

Тема: " Апроксимації гладких ліній вищих порядків. Фрактали."

з дисципліни
" Літня обчислювальна практика"
Варіант № 12

Виконав:
ст. гр. 241 Подарунок М.
Прийняв:
доц. Лазорик В. В.

Чернівці – 2025

Практика. Лабораторна роботи №5.

Виконання лабораторної роботи

1. Зайти в свій обліковий запис на github.com.
2. Клонувати репозиторій https://classroom.github.com/a/Berdcd6 в свій обліковий запис на github.com.
3. Розв’язати завдання.
4. Вихідних код записати в створений репозиторій.

Завдання

1. Створити проект Windows Form (C++ або C#).
2. Розробити програму вирішення задач згідно варіантів(варіанти згідно списку) .

Задача 1.

Тема: Апроксимації гладких ліній вищих порядків.

2. На площині задано чотири точки P1(X1,Y1), P2(X2,Y2), P3(X3,Y3), P4(X4,Y4). Розробити програму для відображення параметричної лінії Без’є у чотирикутнику P1P2P3P4.

```python
import matplotlib.pyplot as plt
import numpy as np

# Точки
P1 = np.array([1, 1])
P2 = np.array([2, 3])
P3 = np.array([4, 3])
P4 = np.array([5, 1])

def bezier(t, P1, P2, P3, P4):
    return (1 - t)**3 * P1 + 3*(1 - t)**2 * t * P2 + 3*(1 - t) * t**2 * P3 + t**3 * P4

t_vals = np.linspace(0, 1, 100)
curve = np.array([bezier(t, P1, P2, P3, P4) for t in t_vals])

plt.plot(curve[:,0], curve[:,1], label="Крива Без’є")
plt.plot(*zip(P1, P2, P3, P4), 'ro--', label='Контрольні точки')
plt.legend()
plt.title("Крива Без’є 3-го порядку")
plt.axis('equal')
plt.grid(True)
plt.show()
```

Рис. 1 – Код завдання 1

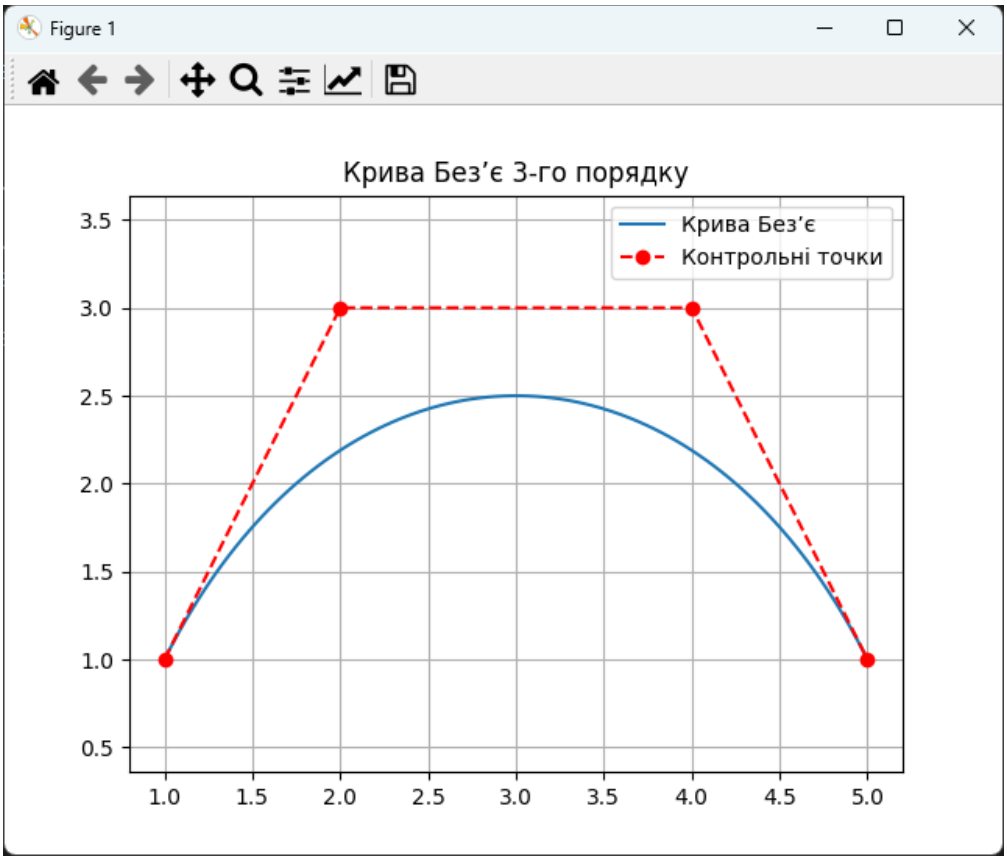

Рис. 2 – Результат виконання завдання 1

Задача 2.

Тема: Фрактали.

4. Реалізувати програму, яка на сторонах квадрата стороною D будує фрактали Коха порядку К з зовнішнім застосуванням правила побудови.

```python
import matplotlib.pyplot as plt
import numpy as np

def koch_curve(p1, p2, order):
    if order == 0:
        return [p1, p2]
    else:
        p1 = np.array(p1)
        p2 = np.array(p2)
        delta = (p2 - p1) / 3
        a = p1 + delta
        b = p1 + 2 * delta

        # Вектор повороту на 60 градусів назовні
        angle = np.pi / 3
        rot = np.array([[np.cos(angle), -np.sin(angle)],
                        [np.sin(angle),  np.cos(angle)]])
        peak = a + rot @ (delta)

        return (koch_curve(p1, a, order-1) +
                koch_curve(a, peak, order-1)[1:] +
                koch_curve(peak, b, order-1)[1:] +
                koch_curve(b, p2, order-1)[1:])

def draw_koch_square(order, size=1):
    # Кутові точки квадрату
    p1 = [0, 0]
    p2 = [size, 0]
    p3 = [size, size]
    p4 = [0, size]

    sides = [
        koch_curve(p1, p2, order),
        koch_curve(p2, p3, order),
        koch_curve(p3, p4, order),
        koch_curve(p4, p1, order)
    ]

    for side in sides:
        x, y = zip(*side)
        plt.plot(x, y)

    plt.axis('equal')
    plt.title(f"Фрактал Коха на квадраті (порядок {order})")
    plt.grid(True)
    plt.show()

draw_koch_square(order=3)
```

Рис. 3 – Код завдання 2

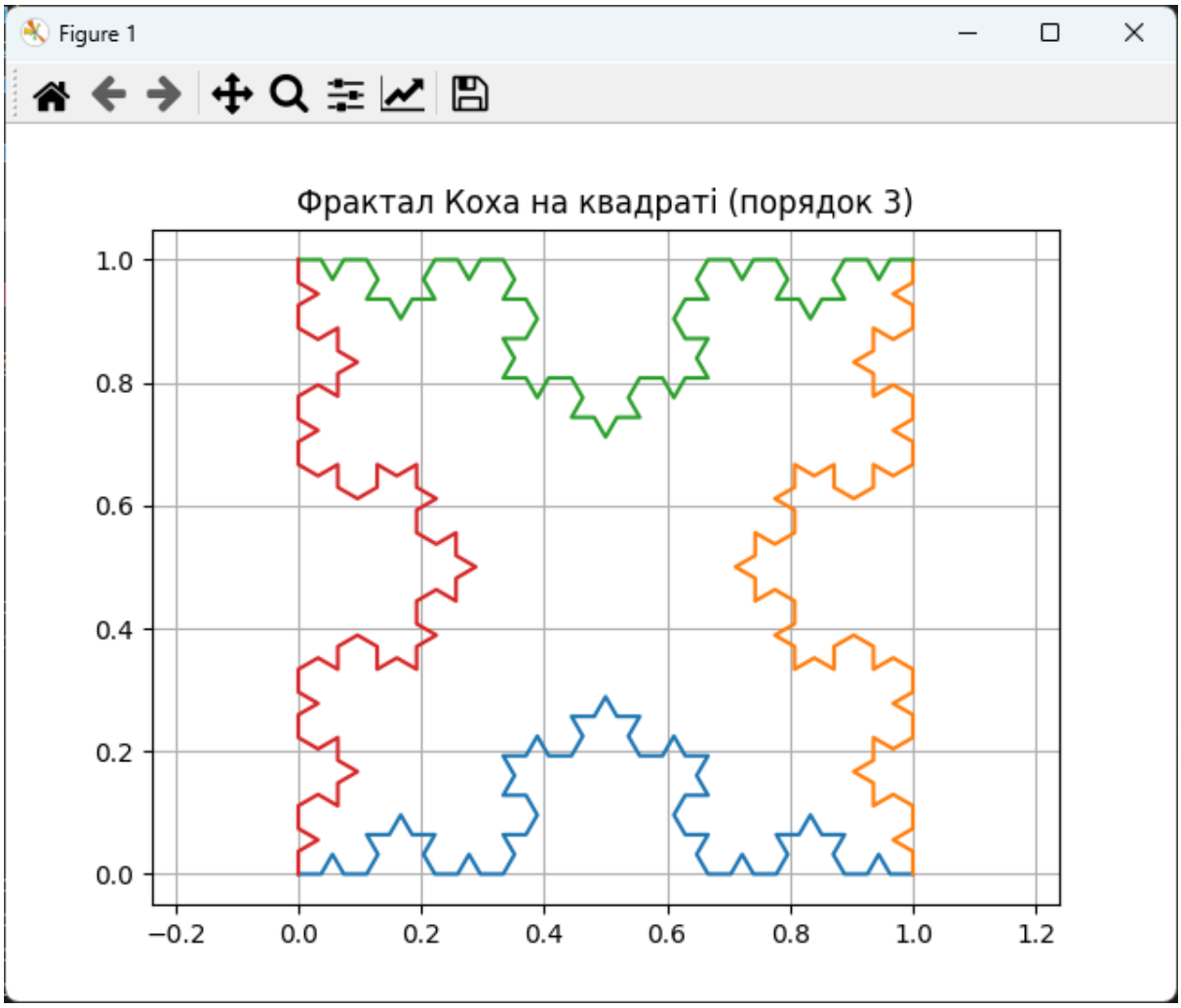

Рис. 4 – Результат виконання завдання 2

Висновок: У ході виконання лабораторної роботи було реалізовано алгоритм побудови фракталу Коха порядку K з внутрішнім застосуванням правила побудови на сторонах квадрата. Застосовано мову програмування Python та графічну бібліотеку turtle, яка дозволила просто та наочно візуалізувати фрактальні конструкції.