**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. ІГОРЯ СІКОРСЬКОГО”**

**НАВЧАЛЬНО НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ**

Комп’ютерна графіка

Модульна контрольна робота

Виконав:

Студент 2 курсу

Групи ФІ-21

Поштаренко Сергій

Київ 2024

***Завдання:***

8. Визначити фрактальну розмірність фрактала на площині, що складається з точок (x; y), де x ∈ [0; 1], y ∈ [0; 1], причому в десятковому представленні числа x відсутні цифри 2 та 7, а в десятковому представленні числа y відсутні цифри 3, 5 та 8. Розробити програмне забезпечення для побудови даного фрактала.

***Результат:***

import numpy as np  # Імпорт бібліотеки numpy для чисельних обчислень

import matplotlib.pyplot as plt  # Імпорт бібліотеки matplotlib для візуалізації

# Перевірка, чи число не містить заборонені цифри

def is\_valid\_number(num, forbidden\_digits):

    num\_str = str(num)  # Перетворюємо число у рядок

    return all(digit not in num\_str for digit in forbidden\_digits)  # Перевіряємо, що жодна заборонена цифра не входить до рядка

# Генерація точок фрактала

def generate\_fractal\_points(x\_forbidden, y\_forbidden, precision=4):

    points = []  # Список для зберігання точок фрактала

    for i in range(10\*\*precision):  # Перебираємо всі можливі значення x з заданою точністю

        x = i / (10\*\*precision)  # Розраховуємо значення x

        if not is\_valid\_number(i, x\_forbidden):  # Перевіряємо, чи містить x заборонені цифри

            continue  # Пропускаємо, якщо містить

        for j in range(10\*\*precision):  # Перебираємо всі можливі значення y з заданою точністю

            y = j / (10\*\*precision)  # Розраховуємо значення y

            if is\_valid\_number(j, y\_forbidden):  # Перевіряємо, чи не містить y заборонені цифри

                points.append((x, y))  # Додаємо точку (x, y) до списку, якщо умови виконуються

    return points  # Повертаємо список точок фрактала

# Параметри генерації

x\_forbidden = ['2', '7']  # Заборонені цифри для x

y\_forbidden = ['3', '5', '8']  # Заборонені цифри для y

precision = 4  # Точність обчислень (кількість знаків після коми)

points = generate\_fractal\_points(x\_forbidden, y\_forbidden, precision)  # Генеруємо точки фрактала

# Візуалізація фрактала

x, y = zip(\*points)  # Розпаковуємо список точок у два списки: x і y

plt.figure(figsize=(8, 8))  # Створюємо нову фігуру для графіку з розмірами 8x8 дюймів

plt.scatter(x, y, s=0.1)  # Наносимо точки на графік з розміром маркерів 0.1

plt.title("Фрактал на площині")  # Додаємо заголовок графіку

plt.xlabel("x")  # Додаємо підпис до осі x

plt.ylabel("y")  # Додаємо підпис до осі y

plt.show()  # Відображаємо графік

# Обчислення кількості коробок для кожного розміру

def box\_counting(points, sizes):

    counts = []  # Список для зберігання кількості коробок для кожного розміру

    for size in sizes:  # Перебираємо всі задані розміри коробок

        num\_boxes = 0  # Лічильник для коробок, що містять частину фрактала

        for i in range(0, int(1 / size)):  # Перебираємо всі можливі позиції коробок по осі x

            for j in range(0, int(1 / size)):  # Перебираємо всі можливі позиції коробок по осі y

                for (x, y) in points:  # Перебираємо всі точки фрактала

                    if i \* size <= x < (i + 1) \* size and j \* size <= y < (j + 1) \* size:  # Перевіряємо, чи потрапляє точка у поточну коробку

                        num\_boxes += 1  # Збільшуємо лічильник, якщо точка потрапляє у коробку

                        break  # Виходимо з внутрішнього циклу, якщо знайшли точку у поточній коробці

        counts.append(num\_boxes)  # Додаємо кількість коробок для поточного розміру до списку

    return counts  # Повертаємо список кількості коробок

# Обчислення фрактальної розмірності

sizes = np.logspace(-1, -4, num=20)  # Генеруємо 20 логарифмічно рівномірно розподілених значень розмірів від 0.1 до 0.0001

counts = box\_counting(points, sizes)  # Обчислюємо кількість коробок для кожного розміру

log\_sizes = np.log(sizes)  # Обчислюємо логарифми розмірів коробок

log\_counts = np.log(counts)  # Обчислюємо логарифми кількості коробок

# Побудова графіку для визначення фрактальної розмірності

plt.figure()  # Створюємо нову фігуру для графіку

plt.plot(log\_sizes, log\_counts, 'o-')  # Будуємо графік логарифмів розмірів проти логарифмів кількості коробок

plt.title("Метод коробок")  # Додаємо заголовок графіку

plt.xlabel("log(size)")  # Додаємо підпис до осі x

plt.ylabel("log(count)")  # Додаємо підпис до осі y

plt.show()  # Відображаємо графік

# Обчислення нахилу графіку (фрактальна розмірність)

fractal\_dimension = -np.polyfit(log\_sizes, log\_counts, 1)[0]  # Обчислюємо нахил лінійної регресії на графіку, що є фрактальною розмірністю

print(f"Фрактальна розмірність: {fractal\_dimension:.4f}")  # Виводимо результат

