Міністерство освіти і науки України

Національний технічний університет України  
“Київський політехнічний інститут імені Ігоря Сікорського”

ЗВІТ  
з лабораторної роботи №3Програмування і алгоритми

*з дисципліни: Алгоритмічні основи обчислювальної геометрії та комп'ютерної графіки*

Виконав:

студент групи КМ-32 Матвій Терещенко

Київ  
КПІ ім. Ігоря Сікорського  
2024

**Завдання**

Необхідно написати програму будь якою мовою з використанням будь яких бібліотек яка

* Зчитує датасет з файлу;
* Знаходить опуклу оболонку і зберігає її у вигляді окремого датасету;
* Встановлює розміри вікна (полотна – canvas size) 960х540 пкс;
* Відображає опуклу оболонку за допомогою відрізків синього кольору;
* Відображає точки вихідного датасету, разом з опуклою оболонкою;
* Виводить результати у файли будь-якого графічного формату.

Знову ж таки у мене ds5, як і в минулій лабі

**Виконання**

Маємо трошки видозмінену другу лабу, тому я знову для візуалізації взяв матплотдіб і єдине що мені потрібно було додати це обчислення опуклого контуру і для цього я використав алгоритм енрю і також написав функції для веторного добутку, щоб не рахувати його кожен раз заново. Також я імпортував нампай для легшої маніпуляції масивами чисел. І для візуалізації самої облонки я використав фунцію plot, яка є частиною матплотліба. Все інше аналогчно другій лабі. Прикріпляю код:

import matplotlib.pyplot as plt

import numpy as np

# Функція для зчитування координат

def read\_dataset(file\_path):

    points = []

    with open(file\_path, 'r') as file:

        for line in file:

            x, y = map(float, line.split())

            points.append((x, y))

    return points

# Функція для обчислення векторного добутку, що визначає орієнтацію трьох точок

def cross\_product(o, a, b):

    return (a[0] - o[0]) \* (b[1] - o[1]) - (a[1] - o[1]) \* (b[0] - o[0])

# Алгоритм Ендрю для обчислення опуклої оболонки

def andrew\_algorithm(points):

    points = sorted(points)

    # Створюємо нижню оболонку

    lower = []

    for p in points:

        while len(lower) >= 2 and cross\_product(lower[-2], lower[-1], p) <= 0:

            lower.pop()

        lower.append(p)

    # Створюємо верхню оболонку

    upper = []

    for p in reversed(points):

        while len(upper) >= 2 and cross\_product(upper[-2], upper[-1], p) <= 0:

            upper.pop()

        upper.append(p)

    # Відкидаємо останні точки з кожної оболонки, оскільки вони дублюються

    return lower[:-1] + upper[:-1]

# Функція для малювання точок та опуклої оболонки

def plot\_points(points, convex\_hull\_points):

    plt.figure(figsize=(9.6, 5.4))

    plt.xlim(0,960)

    plt.ylim(0,540)

    x\_coords, y\_coords = zip(\*points)

    plt.scatter(x\_coords, y\_coords, color='orange')

    # Малюємо опуклу оболонку

    convex\_hull\_points = np.array(convex\_hull\_points)

    plt.plot(np.append(convex\_hull\_points[:, 0], convex\_hull\_points[0, 0]),

            np.append(convex\_hull\_points[:, 1], convex\_hull\_points[0, 1]),

            color='blue', linewidth=2)

    plt.savefig("convex\_hull\_output\_image.png")

# Виклик програми

if \_\_name\_\_ == "\_\_main\_\_":

    dataset\_file = 'DS5.txt'

    points = read\_dataset(dataset\_file)

    convex\_hull\_points = andrew\_algorithm(points)

    plot\_points(points,convex\_hull\_points)