**Минобрнауки РФ**

**ЮЗГУ**

**ФФиПИ**

Кафедра программной инженерии

**ЛАБОРАТОРНАЯ РАБОТА №6**

**АЛГОРИТМЫ ОТСЕЧЕНИЯ ОТРЕЗКОВ**

Выполнил: ст. гр. ПО-22б

Алешин К.Р.

Проверил: Ефремов В. В.

Курск, 2024 г

**Задание:**  
Написать программу (на языке высокого уровня), реализующую алгоритмы отсечения отрезков с их последующей отрисовкой. Количество отрезков и их координаты, а также координаты и размеры окна задаются пользователем.

**Листинг программы:**import matplotlib.pyplot as plt

# Кодовые области для алгоритма Коэна-Сазерленда

INSIDE = 0 # Внутри окна

LEFT = 1 # Слева от окна

RIGHT = 2 # Справа от окна

BOTTOM = 4 # Ниже окна

TOP = 8 # Выше окна

def compute\_out\_code(x, y, x\_min, x\_max, y\_min, y\_max):

"""Вычисляет кодовую область точки."""

code = INSIDE

if x < x\_min: # Слева от окна

code |= LEFT

elif x > x\_max: # Справа от окна

code |= RIGHT

if y < y\_min: # Ниже окна

code |= BOTTOM

elif y > y\_max: # Выше окна

code |= TOP

return code

def cohen\_sutherland\_clip(x1, y1, x2, y2, x\_min, x\_max, y\_min, y\_max):

"""Алгоритм Коэна-Сазерленда для отсечения отрезков."""

out\_code1 = compute\_out\_code(x1, y1, x\_min, x\_max, y\_min, y\_max)

out\_code2 = compute\_out\_code(x2, y2, x\_min, x\_max, y\_min, y\_max)

accept = False

while True:

if not (out\_code1 | out\_code2):

# Если оба конца внутри окна

accept = True

break

elif out\_code1 & out\_code2:

# Если оба конца снаружи окна в одном регионе

break

else:

# Отсечь часть отрезка

x, y = 0, 0

out\_code\_out = out\_code1 if out\_code1 else out\_code2

if out\_code\_out & TOP:

x = x1 + (x2 - x1) \* (y\_max - y1) / (y2 - y1)

y = y\_max

elif out\_code\_out & BOTTOM:

x = x1 + (x2 - x1) \* (y\_min - y1) / (y2 - y1)

y = y\_min

elif out\_code\_out & RIGHT:

y = y1 + (y2 - y1) \* (x\_max - x1) / (x2 - x1)

x = x\_max

elif out\_code\_out & LEFT:

y = y1 + (y2 - y1) \* (x\_min - x1) / (x2 - x1)

x = x\_min

if out\_code\_out == out\_code1:

x1, y1 = x, y

out\_code1 = compute\_out\_code(x1, y1, x\_min, x\_max, y\_min, y\_max)

else:

x2, y2 = x, y

out\_code2 = compute\_out\_code(x2, y2, x\_min, x\_max, y\_min, y\_max)

if accept:

return [(x1, y1), (x2, y2)]

else:

return None

def plot\_clipping(lines, x\_min, x\_max, y\_min, y\_max):

"""Отображение отсечённых отрезков."""

fig, ax = plt.subplots()

# Рисуем окно

ax.plot([x\_min, x\_max, x\_max, x\_min, x\_min],

[y\_min, y\_min, y\_max, y\_max, y\_min], 'r-', label='Окно')

# Рисуем исходные отрезки

for line in lines:

x1, y1, x2, y2 = line

ax.plot([x1, x2], [y1, y2], 'b--', label='Исходный отрезок' if line == lines[0] else "")

# Отсекаем и рисуем отсечённые отрезки

for line in lines:

x1, y1, x2, y2 = line

clipped\_line = cohen\_sutherland\_clip(x1, y1, x2, y2, x\_min, x\_max, y\_min, y\_max)

if clipped\_line:

(cx1, cy1), (cx2, cy2) = clipped\_line

ax.plot([cx1, cx2], [cy1, cy2], 'g-', label='Отсечённый отрезок' if line == lines[0] else "")

ax.legend()

ax.set\_xlim(min(x\_min, -10), max(x\_max, 10))

ax.set\_ylim(min(y\_min, -10), max(y\_max, 10))

ax.set\_aspect('equal', adjustable='box')

ax.grid(True)

plt.show()

def main():

# Ввод данных

n = int(input("Введите количество отрезков: "))

lines = []

for i in range(n):

x1, y1 = map(float, input(f"Введите координаты начала отрезка {i + 1} (x1 y1): ").split())

x2, y2 = map(float, input(f"Введите координаты конца отрезка {i + 1} (x2 y2): ").split())

lines.append((x1, y1, x2, y2))

x\_min, y\_min = map(float, input("Введите координаты нижнего левого угла окна (x\_min y\_min): ").split())

x\_max, y\_max = map(float, input("Введите координаты верхнего правого угла окна (x\_max y\_max): ").split())

# Отображение отсечения

plot\_clipping(lines, x\_min, x\_max, y\_min, y\_max)

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

main()

**Пример работы программы:**

**Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание**

**Изображение выглядит как текст, снимок экрана, линия, График

Автоматически созданное описание**