**Лабораторная работа 5**

**Отчёт**

**Цель работы:**

Закрепить теоретические знания об алгоритмах отсечения отрезков и многоугольников и освоить практическое применение основных методов и алгоритмов отсечения.

**Задача**:

Написать приложение/веб-приложение, которое иллюстрирует работу алгоритма отсечения отрезков методом Сазерленда-Коэна через явное задание прямой, содержащей отрезок, а также алгоритмом отсечения выпуклого многоугольника. Приложение должно:

* Выводить систему координат в соответствующем масштабе.
* Отображать отсекающее окно одним цветом и исходные отрезки другим цветом.
* Выполнять отсечение отрезков с использованием алгоритма Сазерленда-Коэна и алгоритма отсечения выпуклого многоугольника.
* Визуализировать видимые части отрезков после отсечения.

Исходные данные для приложения:

Входной файл, содержащий информацию о координатах отрезков в формате:

n - число отрезков

X1\_1 Y1\_1 X2\_1 Y2\_1

X1\_2 Y1\_2 X2\_2 Y2\_2

...

X1\_n Y1\_n X2\_n Y2\_n - координаты отрезков

Xmin Ymin Xmax Ymax - координаты отсекающего прямоугольного окна

Задача заключается в разработке приложения/веб-приложения, которое визуализирует и демонстрирует работу алгоритма Сазерленда-Коэна для отсечения отрезков и алгоритма отсечения выпуклого многоугольника. Пользователь должен иметь возможность загрузить входной файл с координатами отрезков и отсекающим окном, после чего приложение должно отобразить систему координат, отсекающее окно, исходные отрезки и их видимые части после отсечения.

**Средства разработки**:

Фреймворк Qt и язык Python 3.11\_3.11.1776.0\_x64.

**Ход работы**:

**Part1:**

Line (класс): Представляет отрезок с координатами начальной и конечной точек.

\_\_init\_\_(self, x1, y1, x2, y2): Инициализирует объект Line с координатами начальной точки (x1, y1) и конечной точки (x2, y2).

Viewport (класс): Представляет отсекающее окно прямоугольной формы.

\_\_init\_\_(self, xmin, ymin, xmax, ymax): Инициализирует объект Viewport с координатами верхнего левого угла (xmin, ymin) и нижнего правого угла (xmax, ymax).

clip\_line(line, viewport): Функция, которая выполняет отсечение отрезка line с использованием алгоритма Сазерленда-Коэна и отсекающего окна viewport.

* line: Объект Line, представляющий отрезок для отсечения.
* viewport: Объект Viewport, представляющий отсекающее окно.
* return: Возвращает отсеченный отрезок в виде объекта Line, если отрезок видимый после отсечения. Если отрезок полностью находится вне отсекающего окна или не видим после отсечения, возвращает None.

compute\_code(x, y, xmin, ymin, xmax, ymax): Функция, которая вычисляет двоичный код для точки (x, y) относительно отсекающего окна с координатами (xmin, ymin) и (xmax, ymax).

* x, y: Координаты точки.
* xmin, ymin: Координаты верхнего левого угла отсекающего окна.
* xmax, ymax: Координаты нижнего правого угла отсекающего окна.
* return: Возвращает двоичный код, представляющий положение точки относительно отсекающего окна. Код состоит из 4 битов, где каждый бит представляет положение точки относительно левой, правой, верхней и нижней границ окна соответственно. Если точка находится внутри окна, код будет 0000.

Эти методы позволяют выполнять отсечение отрезков методом Сазерленда-Коэна с использованием отсекающего окна и определять видимые части отрезков после отсечения.

**Part2:**

Алгоритм отсечения выпуклого многоугольника с использованием отсекающего окна (viewport).

Вот краткое описание методов:

Viewport (класс): Представляет отсекающее окно прямоугольной формы.

\_\_init\_\_(self, xmin, ymin, xmax, ymax): Инициализирует объект Viewport с координатами верхнего левого угла (xmin, ymin) и нижнего правого угла (xmax, ymax).

Line (класс): Представляет отрезок с координатами начальной и конечной точек.

\_\_init\_\_(self, x1: float, y1: float, x2: float, y2: float): Инициализирует объект Line с координатами начальной точки (x1, y1) и конечной точки (x2, y2).

intersect(self, other: Line) -> list[float]: Вычисляет точку пересечения текущего отрезка с другим отрезком other.

* other: Другой объект Line, с которым нужно проверить пересечение.
* return: Возвращает точку пересечения в виде списка [x, y] или пустой список [], если отрезки не пересекаются.

compute\_code(x: float, y: float, v: Viewport) -> int: Вычисляет двоичный код для точки (x, y) относительно отсекающего окна v.

* x, y: Координаты точки.
* v: Объект Viewport, представляющий отсекающее окно.
* return: Возвращает двоичный код, представляющий положение точки относительно отсекающего окна. Код состоит из 4 битов, где каждый бит представляет положение точки относительно левой, правой, верхней и нижней границ окна соответственно. Если точка находится внутри окна, код будет 0000.

sutherland\_clip(poly: list[Line], v: Viewport) -> list[Line]: Функция, которая выполняет отсечение многоугольника poly с использованием алгоритма Сазерленда-Ходжмана и отсекающего окна v.

* poly: Список отрезков, представляющих многоугольник для отсечения.
* v: Объект Viewport, представляющий отсекающее окно.
* return: Возвращает список отрезков, представляющих отсеченный многоугольник после применения алгоритма Сазерленда-Ходжмана.

Алгоритм выполняет отсечение каждого ребра многоугольника по очереди против каждого ребра отсекающего окна. На каждом шаге происходит проверка видимости ребра и вычисление точки пересечения, если ребро пересекает отсекающее окно. Затем обновляется список отрезков, представляющих отсеченный многоугольник.

**Выводы**:

В данной лабораторной работе были реализованы алгоритмы отсечения отрезков и многоугольников: алгоритм Сазерленда-Коэна и алгоритм отсечения выпуклого многоугольника.

Алгоритм Сазерленда-Коэна позволяет эффективно определить видимые и невидимые части отрезков, используя явное задание прямой, содержащей отрезок, и вычисление кодов видимости для начальной и конечной точек отрезка. Этот алгоритм позволяет быстро отсекать отрезки, основываясь на их положении относительно отсекающего окна.

Алгоритм отсечения выпуклого многоугольника позволяет определить видимые части многоугольника, используя отсекающее окно в форме выпуклого многоугольника. Он основан на проверке видимости ребер многоугольника и вычислении точек пересечения с ребрами отсекающего многоугольника. Этот алгоритм обеспечивает эффективное отсечение многоугольников и отображение только видимых частей на экране.

Реализация приложения/веб-приложения, иллюстрирующего работу алгоритмов, позволяет визуализировать систему координат, отсекающее окно и исходные отрезки/многоугольники. Применение соответствующих алгоритмов отсечения позволяет выделить и отобразить видимые части отрезков/многоугольников. Это помогает в понимании и визуализации процесса отсечения и определения видимых областей в графической области.

Таким образом, выполнение данной лабораторной работы позволяет закрепить теоретический материал и освоить основные методы и алгоритмы отсечения отрезков и многоугольников, а также разработать приложение/веб-приложение для их визуализации.

Ппрограмма состоит из двух частей и она содержит два проекта и два exe файла.

Также были созданы exe файлы, которые являются самодостаточными.