

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дисциплина « Компьютерная графика»**

**Лабораторная работа №8**

**по теме:**

**«Реализация алгоритма отсечения отрезка произвольным выпуклым отсекателем.»**

**Работу выполнил:**

студент группы ИУ7-43Б

Сукочева А.

**Работу проверил:**

Куров А. В.

2020 г.

**Цель работы:**

Изучение и программная реализация алгоритма отсечения отрезка.

**Задание:**

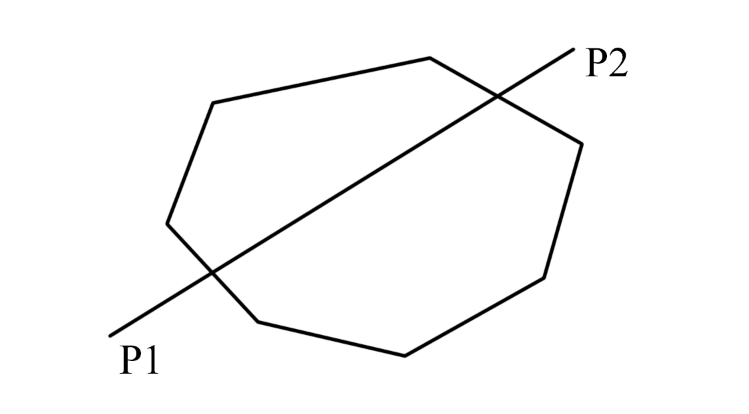
Реализация алгоритма Кируса-Бека.

**Требования:**

1. Необходимо обеспечить ввод отсекателя – произвольного многоугольника. Высветить его первым цветом.
2. Необходимо обеспечить ввод нескольких (до десяти) различных отрезков (высветить их вторым цветом).
3. Реализовать ввод любых отрезков (горизонтальные, вертикальные, имеющие произвольный наклон).
4. Предусмотреть ввод отрезков, параллельных границе отсекателя.
5. Ввод осуществлять с помощью мыши и нажатия других клавиш.
6. Выполнить отсечение отрезков, показав результат третьим цветом. Исходные отрезки не удалять.

**Теоретический материал:**

В данной лабораторной работе мы работаем с выпуклыми отсекателем.

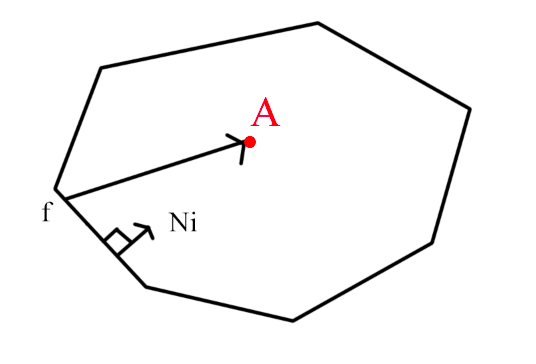
****

Параметрическое уравнение отрезка от P1 до P2:

, где 0<=t<=1

Простых способов определения полностью видимых или полностью невидимых отрезков нет.

Определим, где находится точка: внутри окна, на границе или за его пределам.



f - произвольная точка на границе отсекателя.

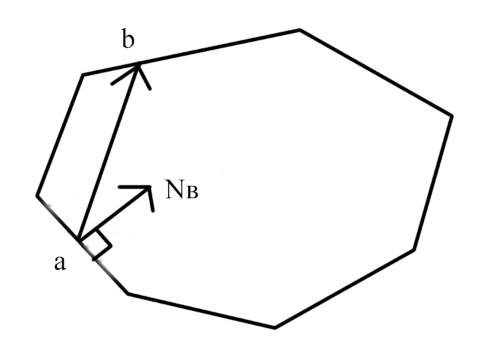
Ni - вектор внутренней нормали.

Ni \* (A – fi) - скалярное произведение.

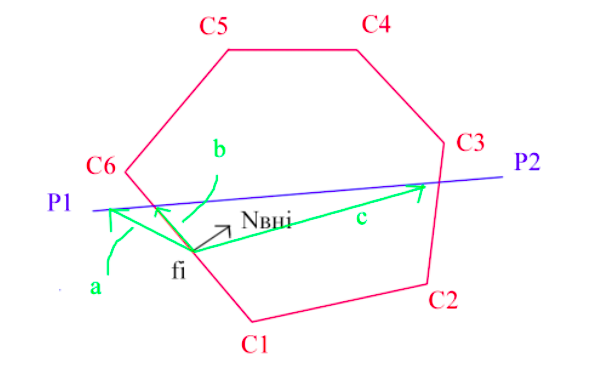
Если произведение:

> 0 - точка лежит по видимую сторону

= 0 - точка лежит на границе отсекателя

< 0 - точка расположена по невидимую сторону отсекателя

Следует отметить, что внутренняя нормаль Nв в произвольной точке a, лежащей на границе отсекателя, должна удовлетворять условию: Nв(b - a) >=0, где b - любая другая точка на границе отсекателя.



Возьмем произвольную точку fi на границе отсекателя. Nвнi - внутренняя нормаль. Рассмотрим три вектора (a, b и c), проведенных из точки fi в точку на отрезке. Рассмотрим скалярные произведения:

сNвнi > 0 => вектор c направлен внутрь нашего отсекателя.

aNвнi < 0 => вектор a направлен во вне нашего отсекателя.

bNвнi = 0 => вектор b ⊥ Nвнi.

Бесконечная прямая пересекает замкнутый выпуклый многоугольник в двух точках. Пусть эти две точки не принадлежат одной границе. Тогда уравнение Nвнi[P(t)-f] = 0 имеет только одно решение. Если точка f лежит на том же ребре, к которому проведена Nвнi, то точка на отрезке P(t) (должна удовлетворять последнему условию) будет пересечением этого отрезка с указанным ребром.

Тем самым имеем условие пересечения отрезка с границей: Nвнi[-fi] = 0

D = P2 – P1 - Директриса отрезка. Определяет направление отрезка.

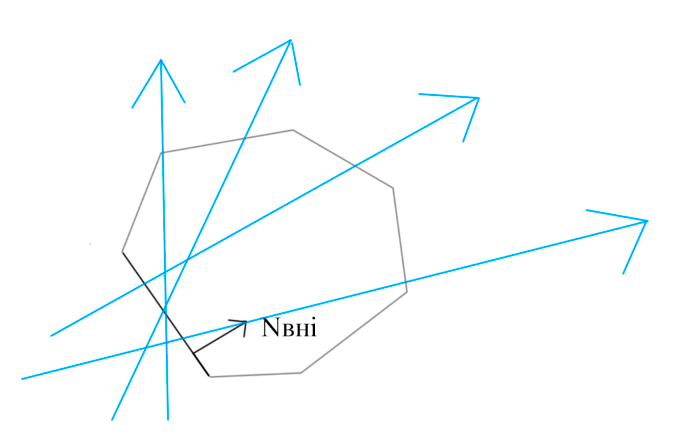
Wi = P1-fi . Выразим t:

, где D может быть равен = 0 при D = 0, а это означает, что отрезок вырожден (превратился в точку (P2 == P1)). Или же если векторы перпендикулярны => отрезок расположен параллельно i-ой стороне отсекателя.

Wск – произведение, стоящее в числителе. Dск – произведение, стоящее в знаменателе.

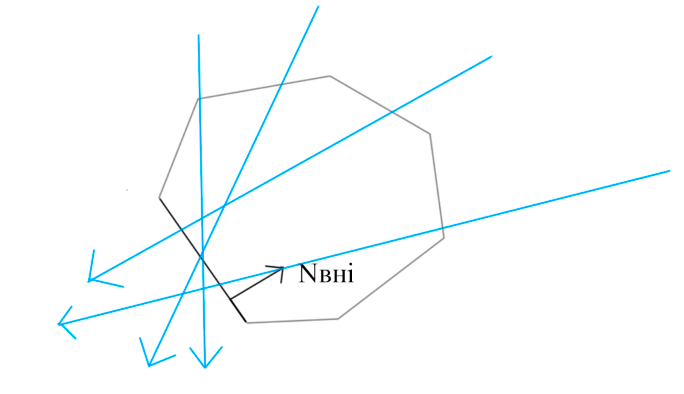
Найденные точки пересечения можно разбить на две группы:

1. Точки пересечения, расположенные ближе к началу отрезка (определяют начало видимой части)
2. Точки пересечения, расположенные ближе к концу (определяют конец видимой части)

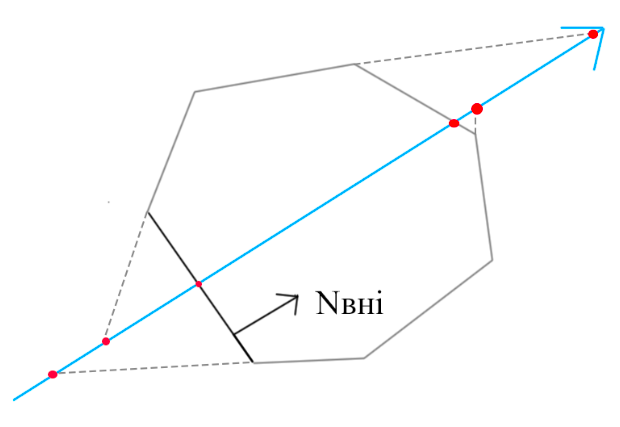
Если Dск > 0 - то точку пересечения нужно отнести к группе, определяющей начало видимой части. 

Рассмотрим приведенный многоугольник (Я высветила более темным цветом рассматриваемую в данный момент сторону многоугольника). Мы имеем Nвнi - вектор внутренней нормали к рассматриваемой стороне. Также мы имеем несколько (показано голубым цветом) векторов, которые пересекают наш многоугольник (именно в рассматриваемой стороне). В данном примере Dск для всех линий будет > 0 и, как мы видим, пересечение является началом видимой части.

Если Dск < 0 - то точку пересечения нужно отнести к группе, определяющей конец видимой части. Рассмотрим другой пример:



Теперь Dск для всех линий будет < 0 и, как мы видим, пересечение является концом видимой части.

Далее из точек, определяющих начало, нужно выбрать максимальное значение, из тех, что определяют конец – минимальное 

Найдя все точки пересечения, мы должны выполнить проверку: tн <= tв.

Чтобы определить, является ли многоугольник выпуклым или нет нужно проанализировать векторное произведения. Если знаки векторов для отдельных углов не совпадают, то многоугольник невыпуклый.

**Алгоритм:**

**def CyrusBeck(edges, line):**

**# Изначально считаем, что отрезок полностью видимый.**

**t\_b, t\_e = 0, 1 # begin, end.**

**# Находим директрису отрезка (Определяем направление).**

**D = FindDirection(line)**

**# Цикл отсечения отрезка.**

**for i in range(len(edges)):**

**# Находим W (fi - верширны многоугольника).**

**W = Find\_W(line, edges[i])**

**# Находим вектор внутренней нормали.**

**if i == len(edges) - 1:**

**N = FindNormal(edges[i], edges[0])**

**else:**

**N = FindNormal(edges[i], edges[i + 1])**

**# Скалярное произведение D на N.**

**Dscalar = scalar(D, N)**

**# Скалярное произведение W на N.**

**Wscalar = scalar(W, N)**

**# Если отрезок расположен параллельно i-ой стороне отсекателя**

**if Dscalar == 0:**

**# И находятся снаружи отсекателя**

**if Wscalar < 0:**

**# то отрезок невидимый.**

**return**

**else:**

**t = -Wscalar / Dscalar**

**# Если Dск > 0 - то точку пересечения**

**# нужно отнести к группе, определяющей начало видимой части.**

**if Dscalar > 0:**

**# Если т. пересечения вне отрезка,**

**# Значит отрезок невидим.**

**if t > 1:**

**return**

**else:**

**# Иначе нужно из точек, определяющих**

**# начало, выбрать максимальное.**

**t\_b = max(t\_b, t)**

**# Если Dск < 0 - то точку пересечения нужно отнести к**

**# группе, определяющей конец видимой части.**

**elif Dscalar < 0:**

**# Если т. пересечения вне отрезка,**

**# Значит отрезок невидим.**

**if t < 0:**

**return**

**else:**

**# Иначе нужно из точек, определяющих**

**# конец, выбрать минимальное.**

**t\_e = min(t\_e, t)**

**# Проверка видимости отрезка**

**if t\_e < t\_b:**

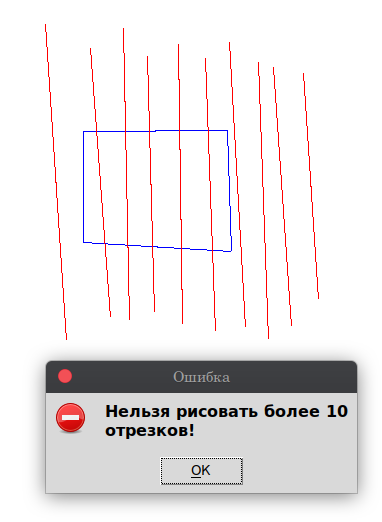
**return**

**# Возвращаем начало и конец видимого отрезка.**

**return [ConvertParametric(line, t\_b), ConvertParametric(line, t\_e)]**

**Результат работы:**

Ввод не более 10 отрезков. При попытке ввести 11-ый отрезок выдает сообщение:



Проверка на выпуклость многоугольника:

