**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра САПР**

**ОТЧЕТ**

**по лабораторной работе №4**

**по дисциплине «Компьютерная графика»**

**Тема: Исследование алгоритмов отсечения отрезков и многоугольников окнами различного вида**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 1302 |  | Марзаева В.И. |
|  |  | Новиков Г.В. |
|  |  | Романова О.В. |
| Преподаватель |  | Колев Г.Ю. |

Санкт-Петербург

2024

**Цель работы**

Обеспечить реализацию алгоритма отсечения массива произвольных отрезков заданным прямоугольным окном с использование алгоритма Коэна Сазерленда. Вначале следует вывести на экран сгенерированные отрезки полностью, а затем другим цветом или яркостью те, которые полностью или частично попадают в область окна.

**Теоретическая часть программы**

В лабораторной работе используется 2-х мерное отсечение (операции отсечения над отрезками). Отсечение необходимо, чтобы из обширной базы данных выделить отдельные элементы для вывода на дисплей.

Есть окно и множество отрезков, для которых нужно определить, какая часть отрезка попадает в окно:

1. виден полностью и его можно вывести в этом окне;
2. виден частично и тогда необходимо определить его видимую часть;
3. не виден полностью и его нужно исключить из дальнейшего рассмотрения.

Этапы алгоритма:

1 этап: выявить полностью видимые отрезки или тривиально невидимые в прямоугольном окне.

2 этап: у оставшихся (других) найти точки пересечения с границами окна, и определить видимую часть или убедиться, что он полностью невидим (нетривиально).

3 этап: вывести на экран.

Алгоритм Сазерленда-Коэна:

Известны размеры окна (параметры – значения координат окна): Х левое, Х правое, У верхнее, У нижнее и параметры концов отрезков.

Идея алгоритма сводится к переносу конечных точек, находящихся вне окна, на линии границ окна с отбрасыванием невидимых частей отрезка.

В начале используются 4-х битовые коды для концов каждого отрезка - – Л, П, Н, В. (левее, правее, ниже, выше окна):

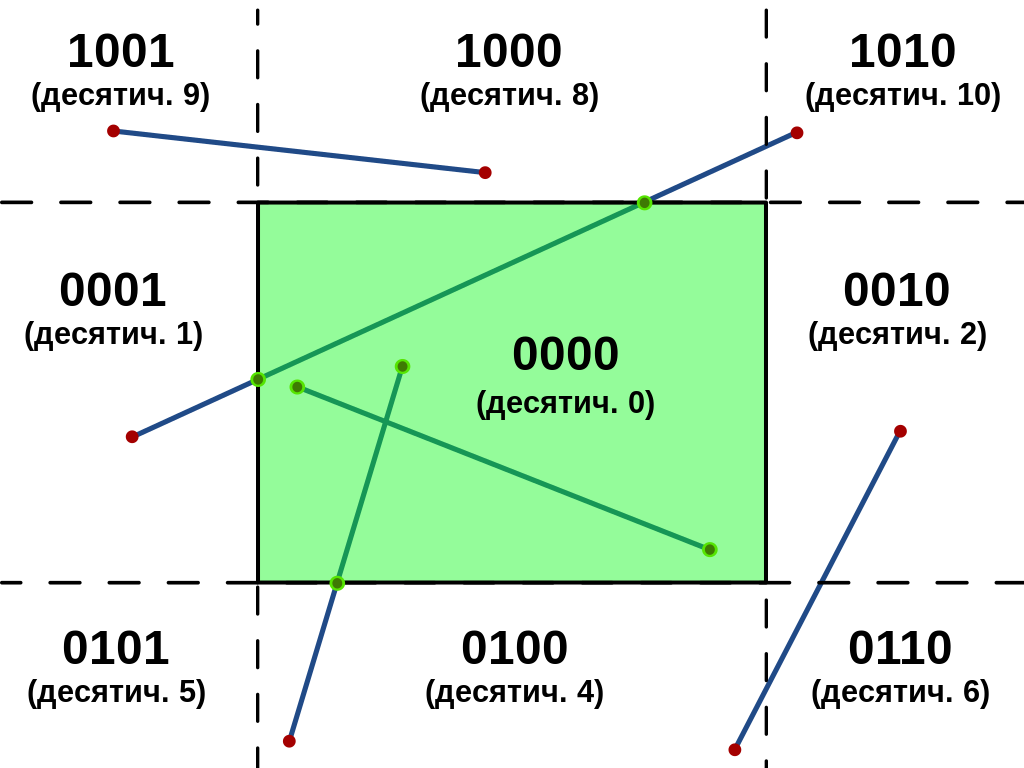


Рис. 1

Исходя из кодов:

Кн & Кк = 0 //полностью видимый отрезок

//т.е. код начала и конца тождественно = 0

Кн \* Кк ≠ 0 // побитовое умножение ≠ 0, то отрезок тривиально не виден

Кн \* Кк = 0 //побитовое умножение = 0, то отрезки могут быть не видны,

либо частично видны

Алгоритм можно представить в следующем виде:

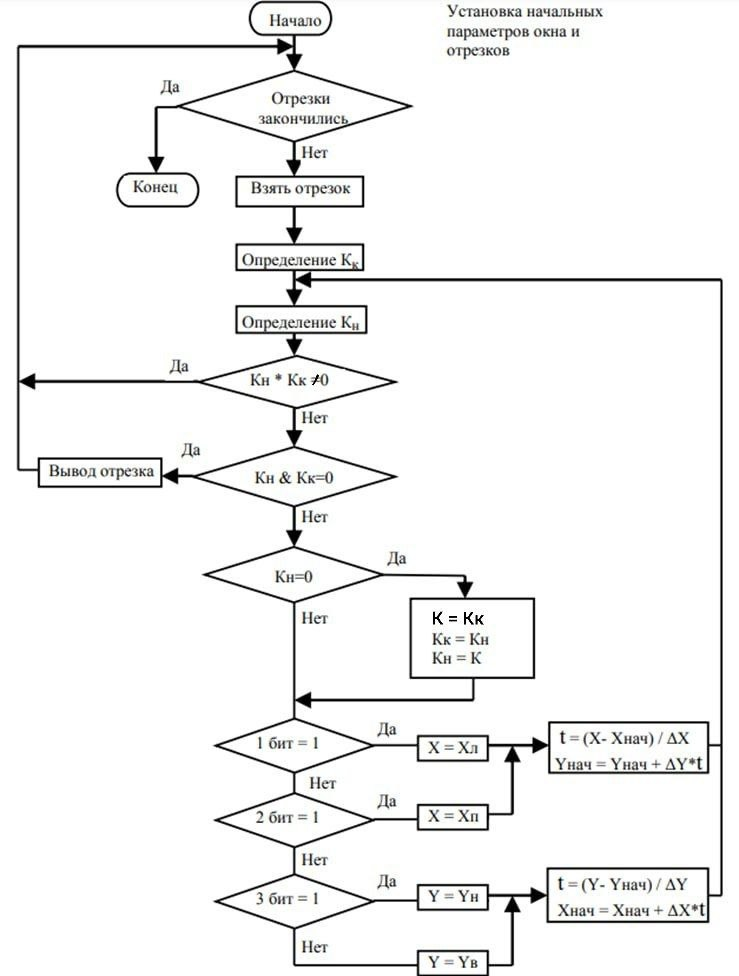


Рис. 2

**Пример работы программы**

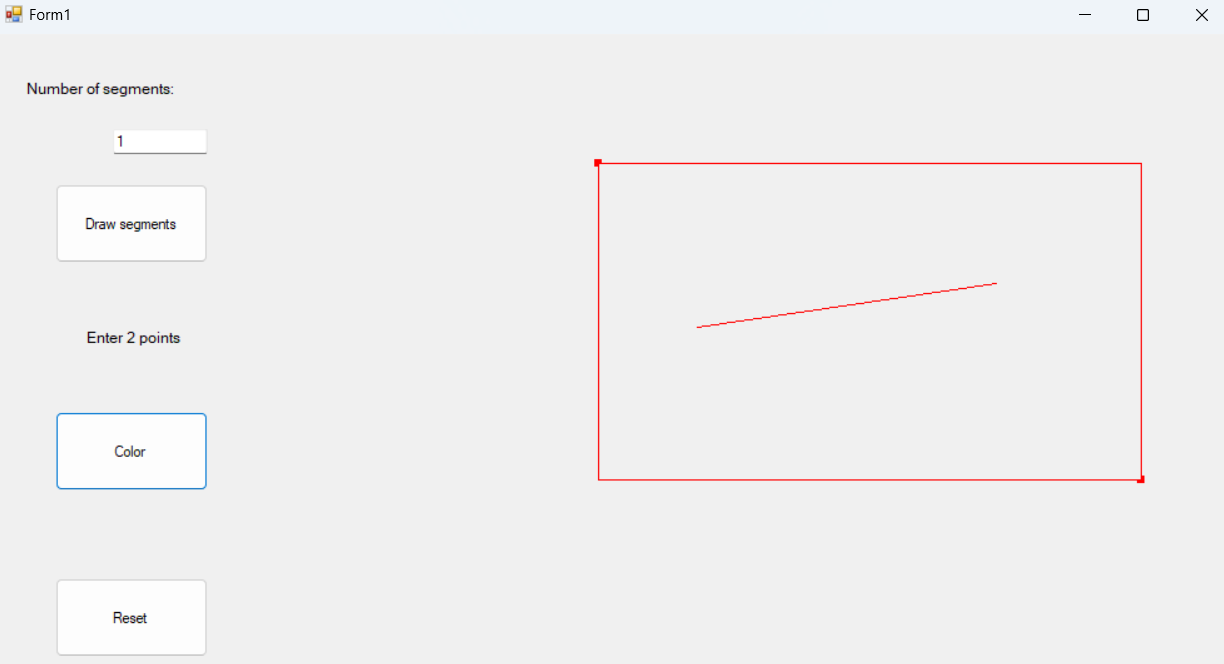


Рис. 3 – Пример работы программы с 1 отрезком

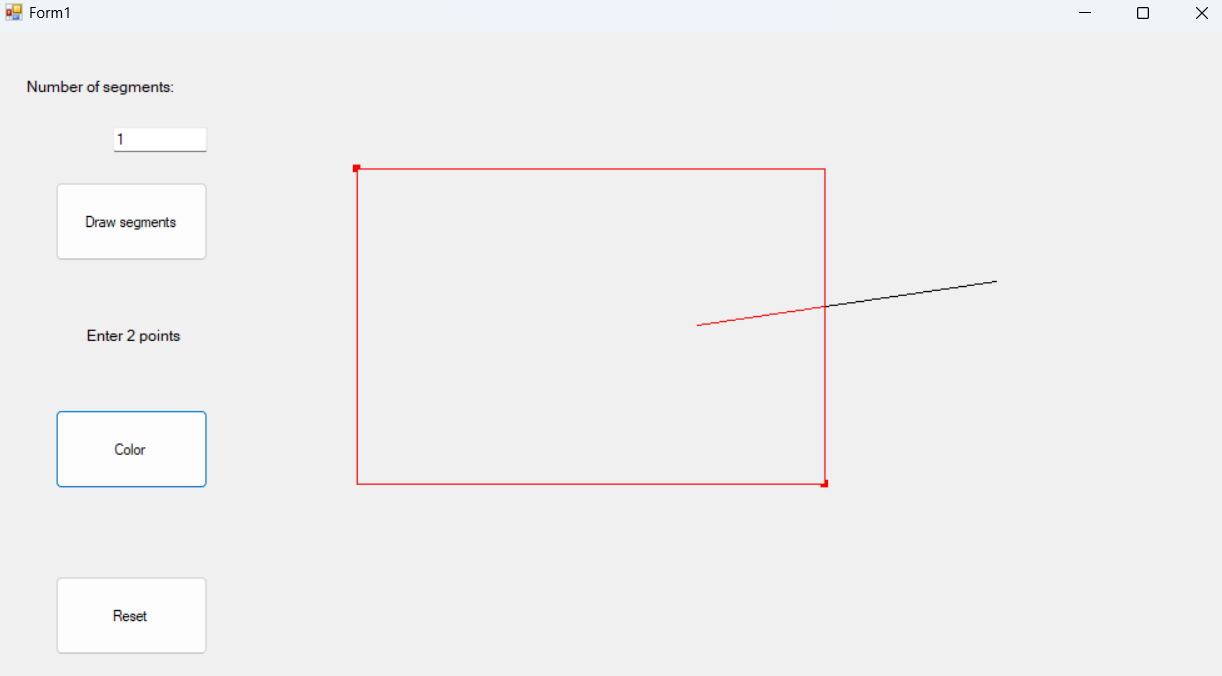


Рис. 4 – Пример работы программы с 1 отрезком

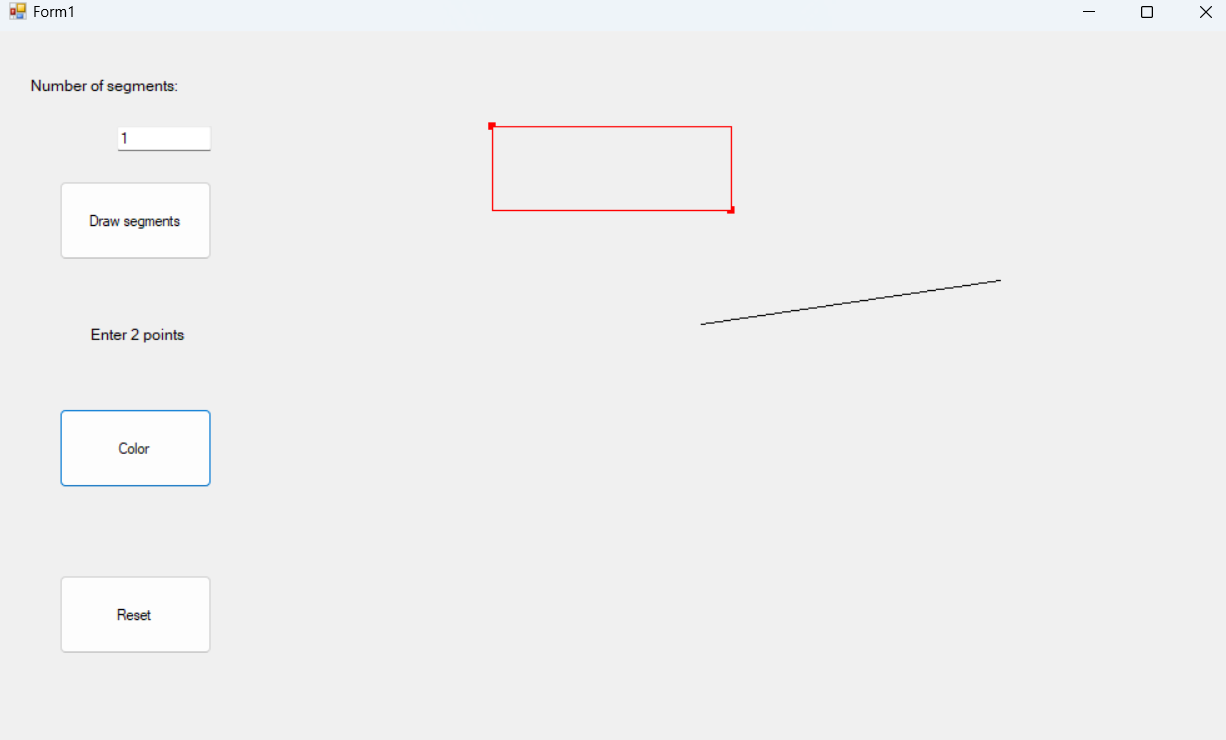


Рис. 5 – Пример работы программы с 1 отрезком

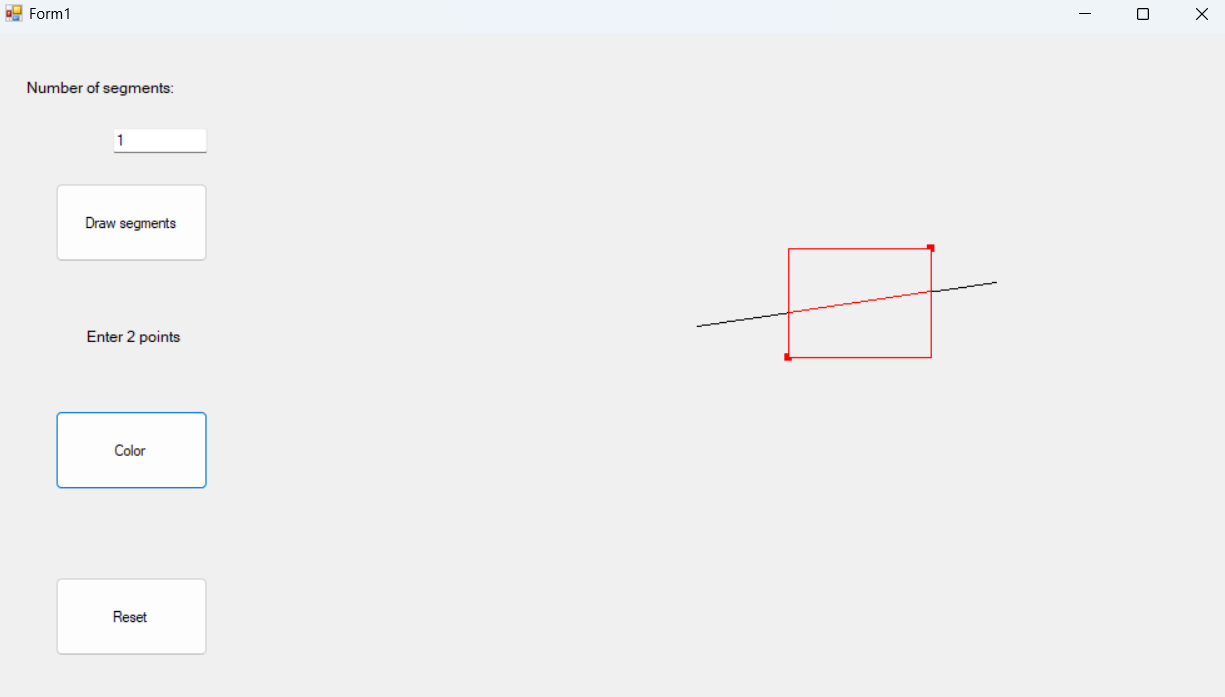


Рис. 6 – Пример работы программы с 1 отрезком

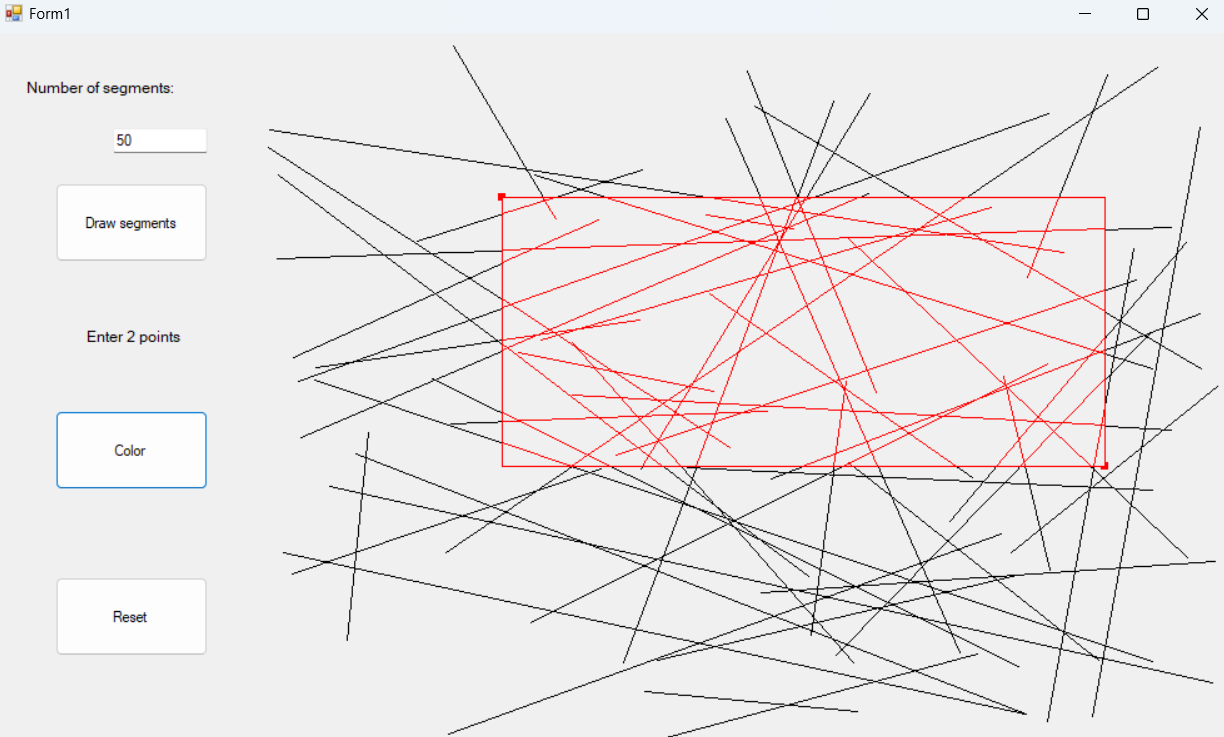


Рис. 7 – Пример работы программы с 50 отрезками

**Код программы**

﻿using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lab4

{

public partial class Form1 : Form

{

double x\_point, y\_point, x\_point2, y\_point2;

double x\_left, x\_right, y\_up, y\_down;

private Random random;

private List<(int, int, int, int)> line = new List<(int, int, int, int)>();

private List<(int, int, int, int)> redLine = new List<(int, int, int, int)>();

bool isFirstClick = true;

public Form1()

{

InitializeComponent();

random = new Random();

}

private void Form1\_Load(object sender, EventArgs e)

{

pictureBox1.Anchor = AnchorStyles.Top | AnchorStyles.Bottom | AnchorStyles.Left | AnchorStyles.Right;

label1.Anchor = AnchorStyles.Top | AnchorStyles.Bottom | AnchorStyles.Left | AnchorStyles.Right;

label2.Anchor = AnchorStyles.Top | AnchorStyles.Bottom | AnchorStyles.Left | AnchorStyles.Right;

textBox1.Anchor = AnchorStyles.Top | AnchorStyles.Bottom | AnchorStyles.Left;

}

private void pictureBox1\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

}

private void button3\_Click(object sender, EventArgs e)

{

Pen pen = new Pen(Color.Red);

SolidBrush brush = new SolidBrush(pictureBox1.BackColor);

Graphics g = pictureBox1.CreateGraphics();

redLine.Clear();

for (int i = 0; i < line.Count; i++)

{

var l = line[i];

string code\_end = GetCode(l.Item3, l.Item4);

string code\_start = GetCode(l.Item1, l.Item2);

string zero = "0000";

bool res1 = false;

bool res2 = false;

char one = '1';

if (code\_start == zero && code\_end == zero)

{

res1 = true;

}

int result = (Convert.ToInt32(code\_start, 2)) & (Convert.ToInt32(code\_end, 2));

string res = Convert.ToString(result, 2).PadLeft(4, '0');

if (res != zero)

{

res2 = true;

}

while (!res2 && !res1)

{

Algorithm(ref l, code\_start, code\_end, one, zero);

code\_start = GetCode(l.Item1, l.Item2);

code\_end = GetCode(l.Item3, l.Item4);

if (code\_start == zero && code\_end == zero)

{

res1 = true;

}

result = (Convert.ToInt32(code\_start, 2)) & (Convert.ToInt32(code\_end, 2));

res = Convert.ToString(result, 2).PadLeft(4, '0');

if (res != zero)

{

res2 = true;

}

}

if (res1)

{

redLine.Add((l.Item1, l.Item2, l.Item3, l.Item4));

}

}

g.FillRectangle(brush, Convert.ToInt32(x\_left) + 1, Convert.ToInt32(y\_up) + 1, Convert.ToInt32(Math.Abs(x\_right - x\_left)) - 1, Convert.ToInt32(Math.Abs(y\_up - y\_down)) - 1);

for (int i = 0; i < redLine.Count; i++)

{

var rl = redLine[i];

g.DrawLine(pen, rl.Item1, rl.Item2, rl.Item3, rl.Item4);

}

}

private void pictureBox1\_MouseClick(object sender, MouseEventArgs e)

{

Graphics g = pictureBox1.CreateGraphics();

Pen pen = new Pen(Color.Red);

if (isFirstClick)

{

g.Clear(BackColor);

DrawLinesFromList(g);

isFirstClick = false;

x\_point = e.X;

y\_point = e.Y;

DrawFatPoint(g, (int)x\_point, (int)y\_point);

}

else

{

isFirstClick = true;

x\_point2 = e.X;

y\_point2 = e.Y;

DrawFatPoint(g, (int)x\_point2, (int)y\_point2);

g.DrawLine(pen, (int)x\_point, (int)y\_point, (int)x\_point2, (int)y\_point);

g.DrawLine(pen, (int)x\_point2, (int)y\_point, (int)x\_point2, (int)y\_point2);

g.DrawLine(pen, (int)x\_point2, (int)y\_point2, (int)x\_point, (int)y\_point2);

g.DrawLine(pen, (int)x\_point, (int)y\_point2, (int)x\_point, (int)y\_point);

if (x\_point < x\_point2)

{

x\_left = x\_point;

x\_right = x\_point2;

if (y\_point < y\_point2)

{

y\_up = y\_point;

y\_down = y\_point2;

}

else

{

y\_up = y\_point2;

y\_down = y\_point;

}

}

else

{

x\_left = x\_point2;

x\_right = x\_point;

if (y\_point > y\_point2)

{

y\_up = y\_point2;

y\_down = y\_point;

}

else

{

y\_up = y\_point;

y\_down = y\_point2;

}

}

}

}

private void button1\_Click(object sender, EventArgs e)

{

Graphics g = pictureBox1.CreateGraphics();

Pen pen = new Pen(Color.Black);

double num\_segments = Convert.ToInt32(textBox1.Text);

if (line.Count > 0)

{

line.Clear();

g.Clear(BackColor);

}

for (int i = 0; i < num\_segments; i++)

{

int x1 = random.Next(pictureBox1.Width);

int y1 = random.Next(pictureBox1.Height);

int x2 = random.Next(pictureBox1.Width);

int y2 = random.Next(pictureBox1.Height);

line.Add((x1, y1, x2, y2));

DrawLinesFromList(g);

}

}

private void button2\_Click(object sender, EventArgs e)

{

Graphics g = pictureBox1.CreateGraphics();

g.Clear(BackColor);

line.Clear();

}

private void pictureBox1\_SizeChanged(object sender, EventArgs e)

{

((PictureBox)sender).Invalidate();

}

private void DrawFatPoint(Graphics g, int x, int y)

{

SolidBrush redBrush = new SolidBrush(Color.Red);

int fatness = 6;

g.FillRectangle(redBrush, x - fatness / 2, y - fatness / 2, fatness, fatness);

}

private void DrawLinesFromList(Graphics g)

{

Pen pen = new Pen(Color.Black);

foreach (var l in line)

{

g.DrawLine(pen, l.Item1, l.Item2, l.Item3, l.Item4);

}

}

private string GetCode(int x, int y)

{

string code\_str = "0000";

StringBuilder sb = new StringBuilder(code\_str);

int indexToChange;

char newChar = '1';

if (x\_left > x)

{

indexToChange = 0;

sb[indexToChange] = newChar;

}

if (x\_right < x)

{

indexToChange = 1;

sb[indexToChange] = newChar;

}

if (y\_down < y)

{

indexToChange = 2;

sb[indexToChange] = newChar;

}

if (y\_up > y)

{

indexToChange = 3;

sb[indexToChange] = newChar;

}

string newCodeStr = sb.ToString();

return newCodeStr;

}

private void Algorithm(ref (int, int, int, int) l, string code\_start, string code\_end, char one, string zero)

{

string k = "";

int it3 = l.Item3, it4 = l.Item4;

double t;

int newX, newY;

if (code\_start == zero)

{

k = code\_start;

code\_start = code\_end;

code\_end = k;

l.Item3 = l.Item1;

l.Item4 = l.Item2;

}

if (code\_start[0] == one)

{

t = (x\_left - l.Item1) / (it3 - l.Item1);

newY = (int)(l.Item2 + (it4 - l.Item2) \* t);

l.Item2 = Math.Max(newY, 0);

l.Item1 = (int)x\_left;

}

else if (code\_start[1] == one)

{

t = (x\_right - l.Item1) / (it3 - l.Item1);

newY = (int)(l.Item2 + (it4 - l.Item2) \* t);

l.Item2 = Math.Max(newY, 0);

l.Item1 = (int)x\_right;

}

else if (code\_start[2] == one)

{

t = (y\_down - l.Item2) / (it4 - l.Item2);

newX = (int)(l.Item1 + (it3 - l.Item1) \* t);

l.Item1 = Math.Max(newX, 0);

l.Item2 = (int)y\_down;

}

else

{

t = (y\_up - l.Item2) / (it4 - l.Item2);

newX = (int)(l.Item1 + (it3 - l.Item1) \* t);

l.Item1 = Math.Max(newX, 0);

l.Item2 = (int)y\_up;

}

}

}

}

**Выводы**

В данной работе с помощью Windows Forms на C# была реализована программа, которая выводит отрезки на экран, потом пользователь вводит координаты прямоугольного окна, внутри которого отрезки выделяются другим цветом. Для отсечения был использован алгоритм Коэна-Сазерленда.