МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа №2 по дисциплине:

«Компьютерная графика»

«Алгоритмы вычерчивания отрезков прямых»

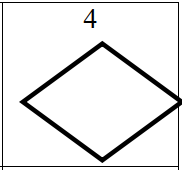
Вариант №4

Выполнил: ст. гр. АСОИ-181

Остапенко А. К.

Проверил: Шилов А. В.

**Задание:** Разработать программу для отображения объекта №1.



Листинг кода программы:

using System;

using System.Windows;

using System.Windows.Media;

using System.Windows.Media.Imaging;

namespace CG\_lab2\_Ostapenko

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

private int \_imageWidth;

private int \_imageHeight;

private double err;

private Figure rhombus;

delegate bool ConditionCheker();

public MainWindow()

{

InitializeComponent();

}

private void CreateFigureButton\_Click(object sender, RoutedEventArgs e)

{

try

{

\_imageHeight = \_imageWidth = int.Parse(ResolutionTextBox.Text);

}

catch

{

\_imageHeight = \_imageWidth = 500;

}

var bitmap = new WriteableBitmap(\_imageWidth + 10, \_imageHeight + 10, 96, 96, PixelFormats.Bgr32, null);

ImageBox.Source = bitmap;

CreateFigure();

for (int index = 0; index < rhombus.Lines.Length; index++)

{

err = -(1.0 / 2.0);

double delta = (double)Math.Abs(rhombus.Lines[index].Y1 - rhombus.Lines[index].Y2) / (double)Math.Abs(rhombus.Lines[index].X1 - rhombus.Lines[index].X2);

int y = 0;

double y2 = 0;

int x = 0;

double x2 = 0;

if ((int)rhombus.Lines[index].X1 < rhombus.Lines[index].X2)

{

x = (int)rhombus.Lines[index].X1;

x2 = rhombus.Lines[index].X2;

y = (int)rhombus.Lines[index].Y1;

y2 = rhombus.Lines[index].Y2;

}

else

{

x2 = rhombus.Lines[index].X1;

x = (int)rhombus.Lines[index].X2;

y2 = rhombus.Lines[index].Y1;

y = (int)rhombus.Lines[index].Y2;

}

ConditionCheker isLineEnded;

int yCrement = 0;

if (y > y2)

{

isLineEnded = () => y < y2 && x > x2;

yCrement = -1;

}

else

{

isLineEnded = () => y > y2 && x > x2;

yCrement = 1;

}

while (!isLineEnded())

{

try

{

FillPixel(bitmap, x, y);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

break;

}

err += delta;

if (err > 0)

{

y += yCrement;

err--;

}

x++;

}

}

}

private void FillPixel(WriteableBitmap bitmap, int x, int y)

{

byte red = 255;

byte green = 255;

byte blue = 255;

byte[] colorData = { blue, green, red, 255 };

var rect = new Int32Rect(x, y, 1, 1);

bitmap.WritePixels(rect, colorData, 4, 0);

}

private void CreateFigure()

{

Line[] lines =

{

new Line(0, \_imageHeight / 2, \_imageWidth / 2, 0),

new Line(\_imageWidth / 2, 0, \_imageWidth, \_imageHeight / 2),

new Line(\_imageWidth, \_imageHeight / 2, \_imageWidth / 2, \_imageHeight),

new Line(\_imageWidth / 2, \_imageHeight, 0, \_imageHeight / 2)

};

rhombus = new Figure(lines);

}

}

}

namespace CG\_lab2\_Ostapenko

{

class Figure

{

public Line[] Lines { get; set; }

public Figure(Line[] lines)

{

Lines = lines;

}

}

}

namespace CG\_lab2\_Ostapenko

{

public class Line

{

public double X1 { get; set; }

public double Y1 { get; set; }

public double X2 { get; set; }

public double Y2 { get; set; }

public Line(double x1, double y1, double x2, double y2)

{

X1 = x1;

Y1 = y1;

X2 = x2;

Y2 = y2;

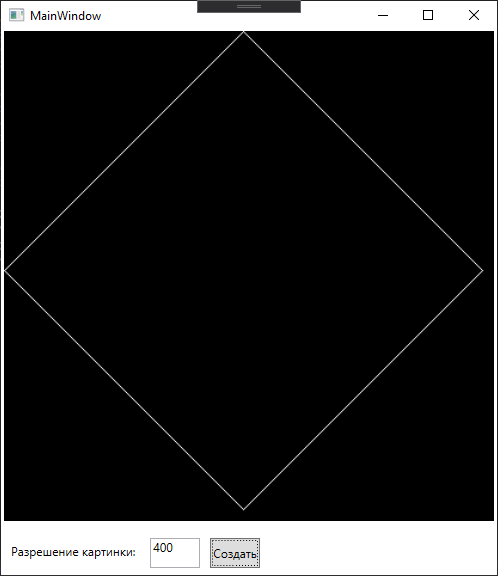
}

}

}

Блок-схема алгоритма:

Результат работы программы:



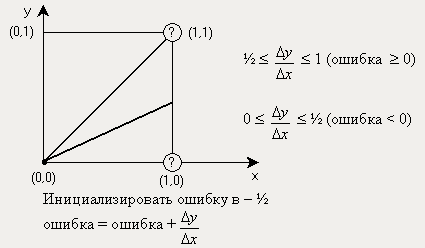
Вопросы:

1.Записать рекуррентное соотношение для последовательных значений по методу ЦДА.

2.Алгоритм формирования отрезка методом ЦДА

1) аппроксимируем длину отрезка и задаем начальные координаты, 2) полагаем большее из приращений Dx или Dy равным единице растра, 3)используем цикл для нахождения новых координат

3.Представить график ошибки в алгоритме Брезенхема



4.Как изменяется величина ошибки при закрашивании пиксела по оси Х

ē = - Δx

(е = -1/2 + 1/Δx)

5.Как изменяется величина ошибки при закрашивании пиксела по оси Y

ē =2Δy

(е = -1/2 + Δy)

6.Пояснить схему алгоритма построения отрезка методом Брезенхема

Этот алгоритм определяет, какие точки двумерного растра нужно закрасить, чтобы получить близкое приближение прямой линии между двумя заданными точками. Большее из приращений, либо Δx, либо Δy, выбирается в качестве единицы растра. В процессе работы одна из координат (в зависимости от углового коэффициента) изменяется на единицу. Изменение другой координаты (на 0 или 1) зависит от расстояния между действительным положением отрезка и ближайшими координатами сетки, т.е. ошибкой. Алгоритм построен так, что требуется проверять лишь знак этой ошибки.