Министерство науки и высшего образования Российской Федерации

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Лабораторная работа №4

по дисциплине

«Компьютерная графика»

Факультет прикладной математики и информатики

Группа ПМ-63

Студенты Майер В.А.

Преподаватели Задорожный А.Г.

Вариант 3

Новосибирск 2019

1. Цель работы

Реализовать программу, отображающий график функции получаемой в результате исполнения соответствующего сплайна.

1. Текст задания

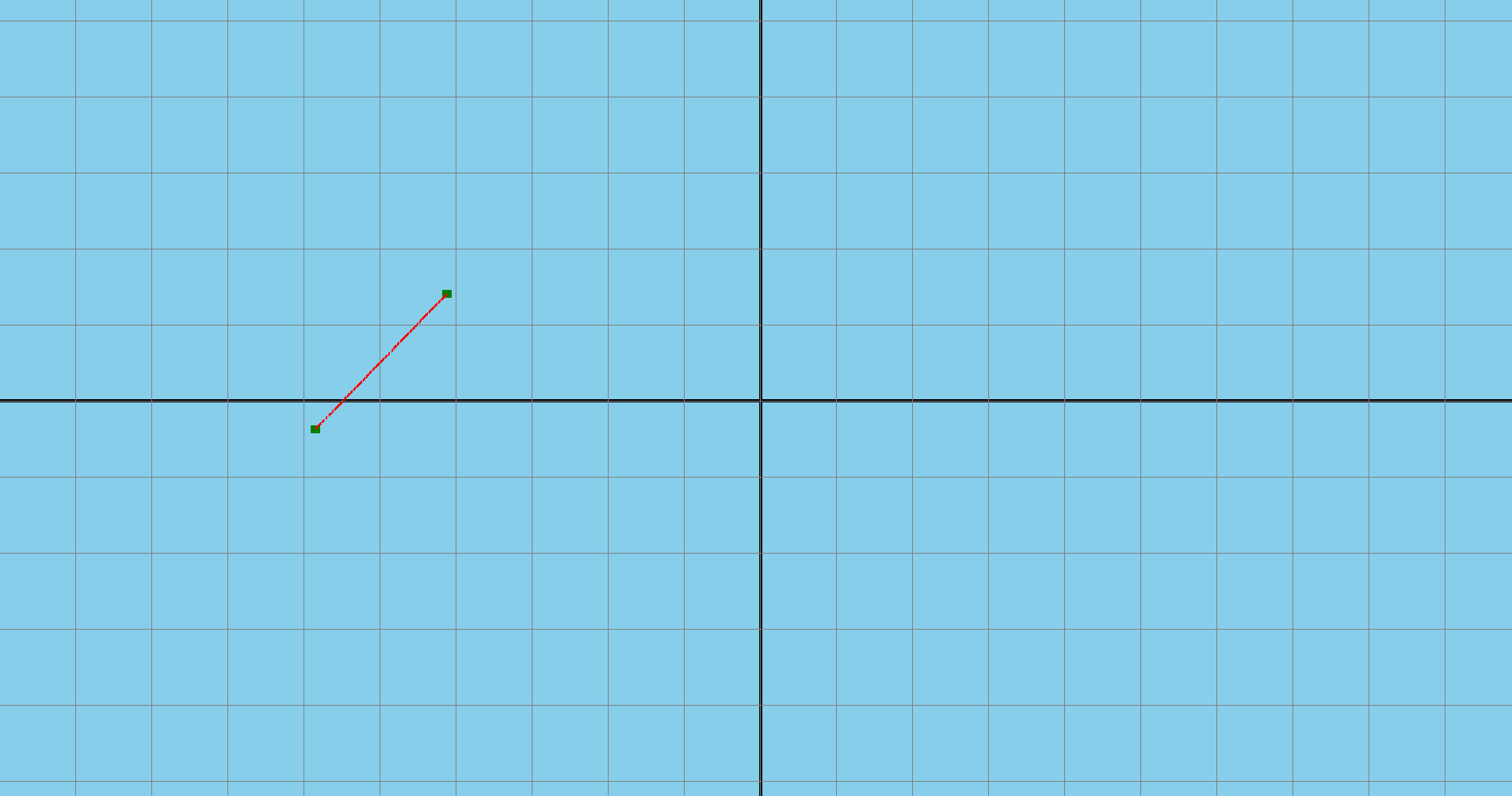
Построить интерполяционный сплайн на основе полинома Лагранджа степени n.

1. Руководство пользователя

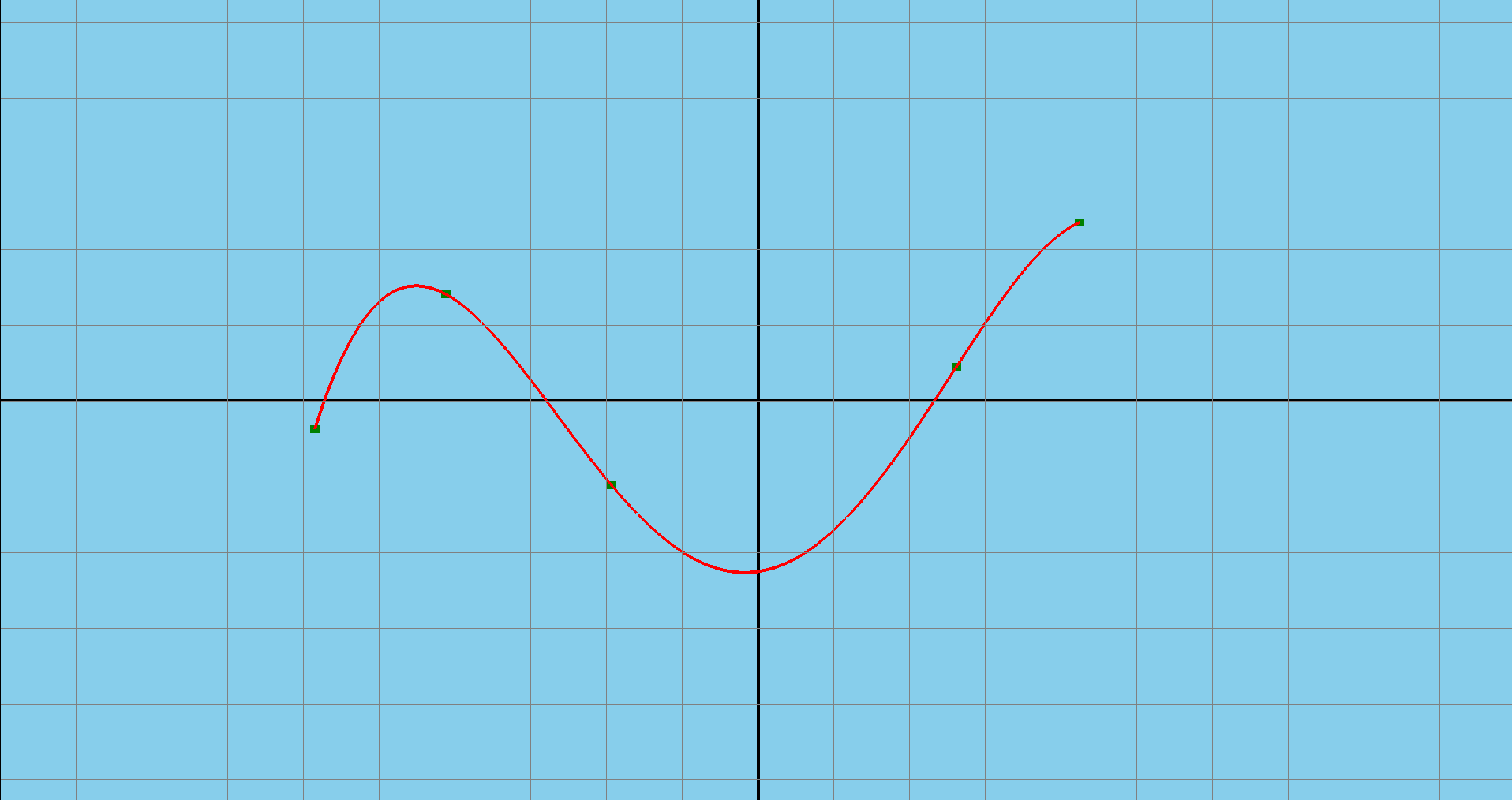
|  |  |
| --- | --- |
| Кнопка | Назначение |
| ЛКМ | Добавить точку |
| WASD | Передвижение |
| + | Увеличить масштаб |
| - | Уменьшить масштаб |
| , | Уменьшить степень полинома Лагранджа |
| . | Увеличить степень полинома Лагранджа |
| Esc | Выход из программы |

1. **Тесты**

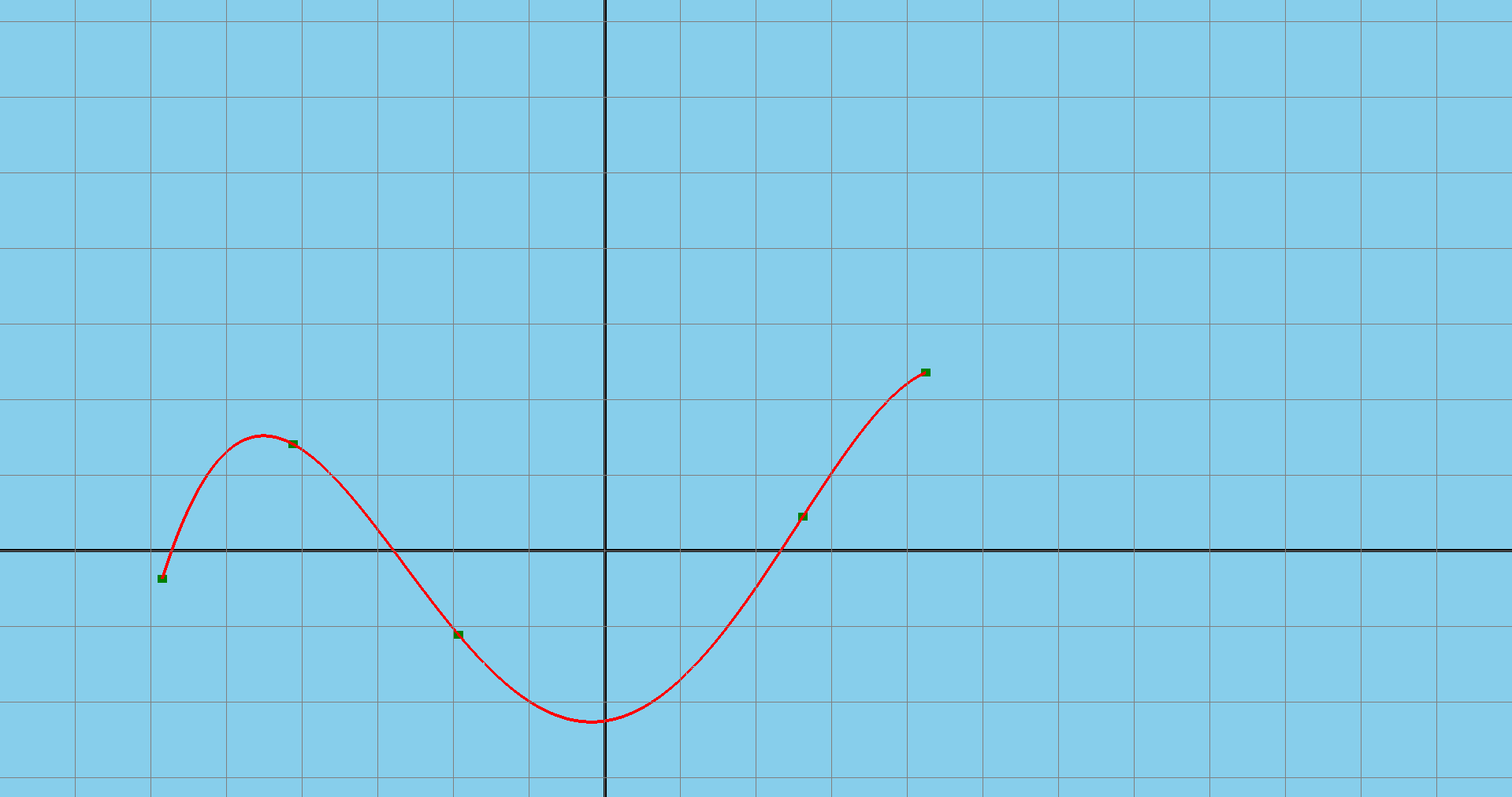
Создание прямой



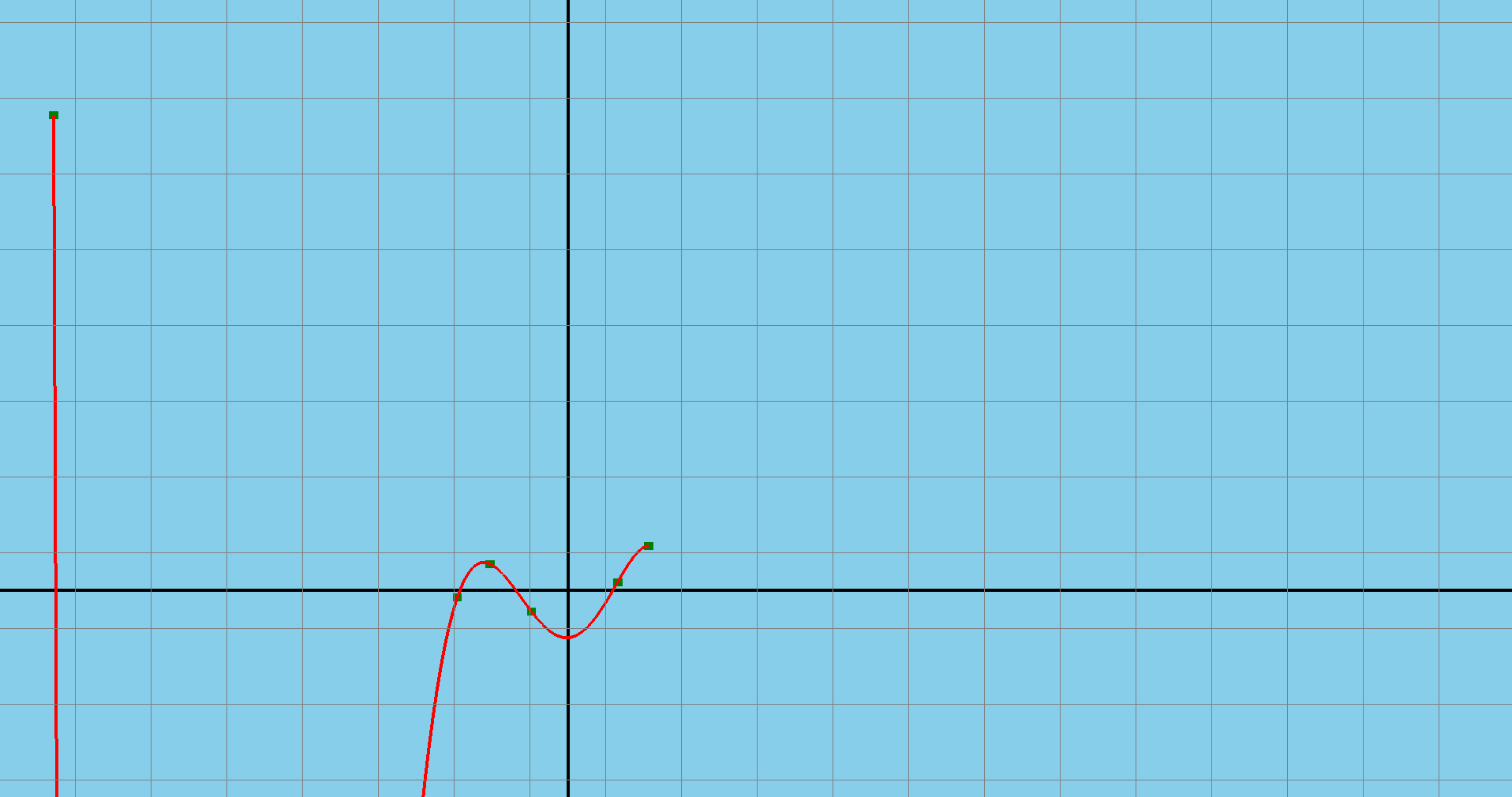
Создание сплайна



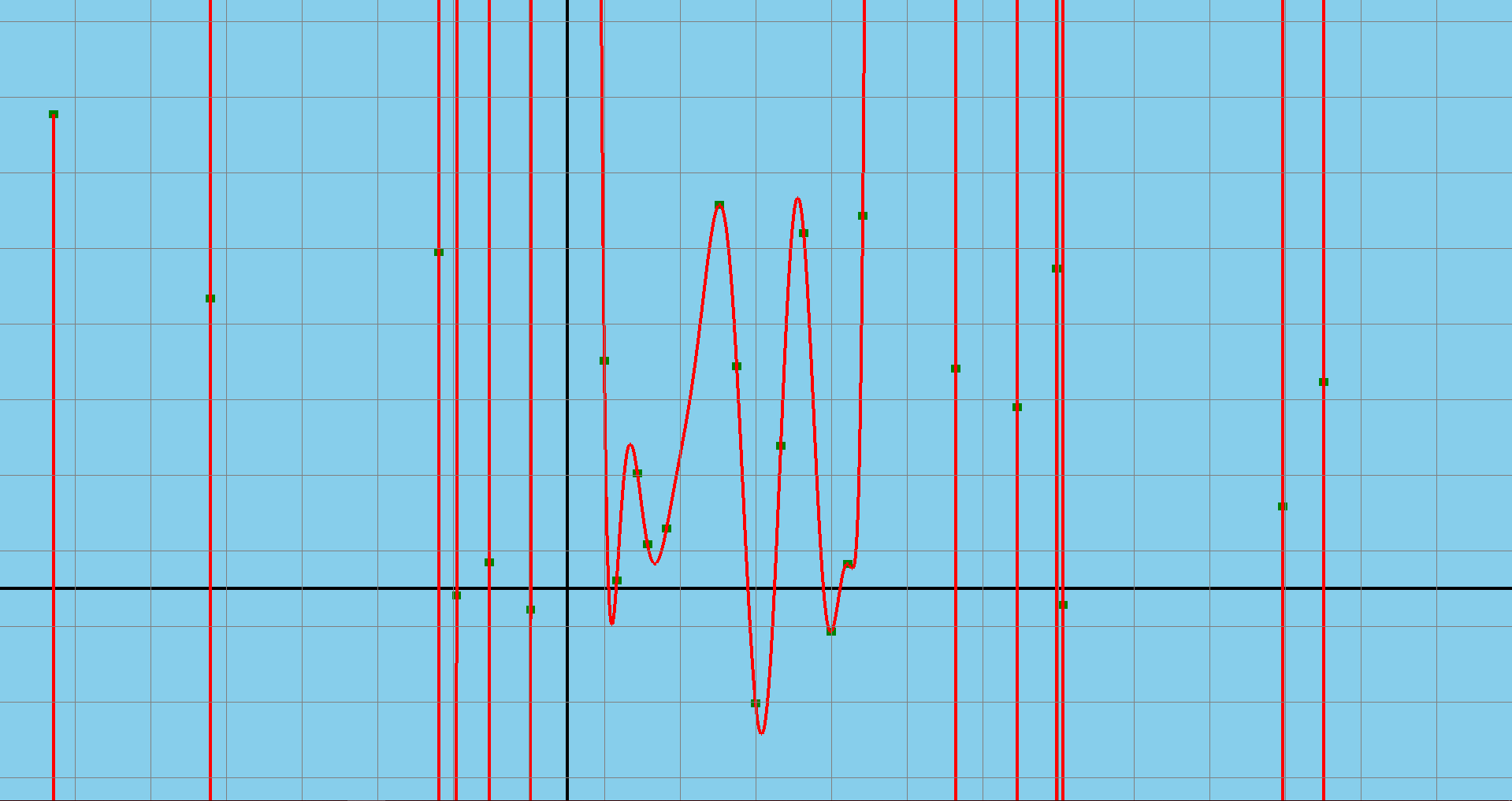
Смещение



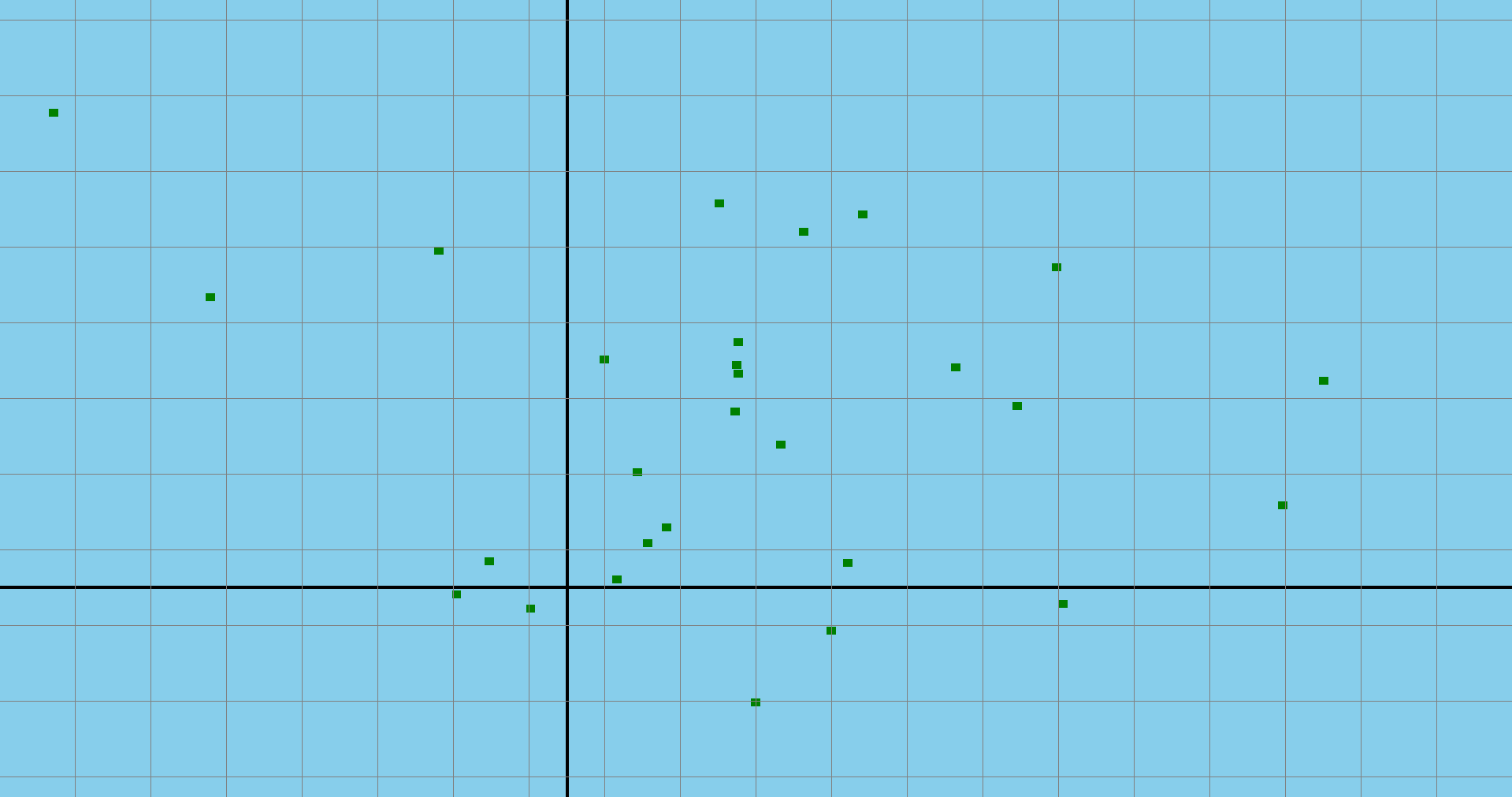
Сжатие



Сплайн с большим числом точек



Сплайн с точками имеющие одинаковый x



1. **Текст программы**

using System;

using System.Threading;

using System.IO;

using System.Text;

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Imaging;

using OpenTK;

using OpenTK.Graphics;

using OpenTK.Graphics.OpenGL;

using OpenTK.Input;

namespace Example

{

class Point

{

public Point(float a, float b)

{

x = a;

y = b;

}

public float x;

public float y;

public Color color;

public static Point operator -(Point a, Point b)

{

return new Point(a.x - b.x, a.y - b.y);

}

public static Point operator /(Point a, float b)

{

return new Point(a.x / b, a.y / b);

}

}

partial class MyApplication

{

public static float Ln(List<Point> points, float x, int n)

{

double mult = points[n].y;

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

if (i != n)

mult \*= (x - points[i].x) / (points[n].x - points[i].x);

return (float)mult;

}

public static Vector2 Lagrange(List<Point> point, float x, Point Co, float Mk)

{

float y = 0;

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

y += Ln(point, x, i);

return (new Vector2(x, y) + new Vector2(Co.x, Co.y)) \* Mk;

}

/// <summary>

/// Рисование сплайна в указанных точках

/// </summary>

/// <param name="points"> Точки сплайна </param>

/// <param name="n"> Количество точек на каждый отрезок </param>

/// <param name="razmer"> Размерность </param>

public static void PrintSpline(List<Point> points, int n, int razmer, Point Co, float Mk)

{

if (points.Count < 2)

return;

int p = 1;

List<Point> basis = new List<Point>();

GL.Color3(Color.Red);

GL.Begin(PrimitiveType.LineStrip);

float hx;

Point h = points[0];

while(p < points.Count)

{

basis.Clear();

basis.Add(h);

for (int i = 0; i < razmer && p < points.Count; i++, p++)

basis.Add(points[p]);

hx = (basis[basis.Count - 1].x - basis[0].x) / n;

if (basis.Count > 1)

{

for (int i = 0; i <= n; i++)

GL.Vertex2(Lagrange(basis, basis[0].x + hx \* i, Co, Mk));

h = basis[basis.Count - 1];

}

}

GL.End();

}

public static void PrintPoints(List<Point> points, Point Co, float Mk)

{

GL.Color3(Color.Green);

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

{

GL.Begin(PrimitiveType.Polygon);

GL.Vertex2(Mk \* (new Vector2(points[i].x - 0.006f / Mk, points[i].y - 0.01f / Mk) + new Vector2(Co.x, Co.y)));

GL.Vertex2(Mk \* (new Vector2(points[i].x + 0.006f / Mk, points[i].y - 0.01f / Mk) + new Vector2(Co.x, Co.y)));

GL.Vertex2(Mk \* (new Vector2(points[i].x + 0.006f / Mk, points[i].y + 0.01f / Mk) + new Vector2(Co.x, Co.y)));

GL.Vertex2(Mk \* (new Vector2(points[i].x - 0.006f / Mk, points[i].y + 0.01f / Mk) + new Vector2(Co.x, Co.y)));

GL.End();

}

}

public static void PrintGrid(GameWindow game)

{

GL.Color3(Color.Gray);

GL.LineWidth(1);

GL.Begin(PrimitiveType.Lines);

for (int i = -10; i < 10; i++)

{

GL.Vertex2(i / 10f, 1);

GL.Vertex2(i / 10f, -1);

}

float k = 10f \* game.Height / game.Width;

for(int i = -(int)k; i < (int)k + 1; i++)

{

GL.Vertex2(1, i/ k);

GL.Vertex2(-1, i / k);

}

GL.End();

GL.LineWidth(4);

}

[STAThread]

public static void Main()

{

List<Point> points;

int n = 10000;

int abvgd = 4;

Point Co = new Point(0, 0); //Сдвиг

float Mk = 1; //Растяжение

using (var game = new GameWindow())

{

game.Title = "Lab4";

game.WindowState = WindowState.Maximized;

game.WindowBorder = WindowBorder.Fixed | WindowBorder.Hidden;

points = new List<Point>();

game.Load += (sender, e) =>

{

game.VSync = VSyncMode.On;

GL.LineWidth(3);

GL.ClearColor(Color.SkyBlue);

GL.Enable(EnableCap.DepthTest);

};

game.MouseDown += (sender, e) =>

{

if (e.Button == MouseButton.Left)

{

points.Add(( new Point((e.X - game.Width / 2f) / game.Width \* 2f, (game.Height / 2f - e.Y) / game.Height \* 2f))/ Mk - Co);

points.Sort(delegate(Point a, Point b) {

if (a.x > b.x) return 1;

if (a.x < b.x) return -1;

return 0;

});

}

};

game.Resize += (sender, e) =>

{

GL.Viewport(0, 0, game.Width, game.Height);

};

game.KeyDown += (sender, e) =>

{

if (e.Key == Key.A)

Co.x -= 1 / 5f / Mk;

if (e.Key == Key.D)

Co.x += 1 / Mk / 5f;

if (e.Key == Key.W)

Co.y += 1 / Mk / (5f \* game.Height / game.Width);

if (e.Key == Key.S)

Co.y -= 1 / Mk / (5f \* game.Height / game.Width);

if (e.Key == Key.Comma)

if(abvgd > 1)

abvgd--;

if (e.Key == Key.Period)

abvgd++;

if (e.Key == Key.Minus)

Mk /= 2;

if (e.Key == Key.Plus)

Mk \*= 2;

if (e.Key == Key.Escape)

game.Close();

};

game.UpdateFrame += (sender, e) =>

{

};

game.RenderFrame += (sender, e) =>

{

// render graphics

GL.Clear(ClearBufferMask.DepthBufferBit | ClearBufferMask.ColorBufferBit | ClearBufferMask.StencilBufferBit);

PrintGrid(game);

PrintSpline(points, n, abvgd, Co, Mk);

PrintPoints(points, Co, Mk);

GL.Color3(Color.Black);

GL.Begin(PrimitiveType.Lines);

GL.Vertex2((new Vector2(-1f, 0) + new Vector2(0, Co.y \* Mk )));

GL.Vertex2((new Vector2(1f, 0) + new Vector2(0, Co.y \* Mk)));

GL.Vertex2((new Vector2(0, -1f) + new Vector2(Co.x \* Mk, 0 )));

GL.Vertex2((new Vector2(0, 1f) + new Vector2(Co.x \* Mk, 0)));

GL.End();

game.SwapBuffers();

};

game.Unload += (sender, e) =>

{

};

//60 кадров в сек

game.Run(60);

}

}

}

class Oper

{

//Норма по осям X, Z

public static double Norm2D(Vector3 p)

{

return Math.Sqrt(Math.Pow(p.X, 2) + Math.Pow(p.Z, 2));

}

public static double Norm3D(Vector3 p)

{

return Math.Sqrt(Math.Pow(p.X, 2) + Math.Pow(p.Y, 2) + Math.Pow(p.Z, 2));

}

public static Vector3 Normal(Vector3 a, Vector3 b, Vector3 c)

{

b = b - a;

c = a - c;

return new Vector3(b.Y \* c.Z - b.Z \* c.Y, b.Z \* c.X - b.X \* c.Z, b.X \* c.Y - b.Y \* c.X);

}

}

}