Учреждение образования

«Гомельский Государственный технический университет имени П.О.Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Отчет по лабораторной работе №5

«Реализация программ освещения и закраски»

По дисциплине «Алгоритмические основы современной компьютерной графики»

Выполнил студент гр.ИТЗ-21

Касаткин Д.О.

Проверил преподаватель

Комракова М.В.

Гомель 2019

**Цель работы:** получить теоретические и практические знания по созданию моделей освещенности.

**Задание:** разработать программу, создающую простую модель освещения.

Листинг программы:

using SharpGL;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab\_5

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

inti\_figuries();

drow\_figures();

radioButton1.Select();

}

List<Osxy> osxy\_triangle\_up\_down = new List<Osxy>();

List<Osxy> osxy\_triangle\_left\_rigth = new List<Osxy>();

public void inti\_figuries()

{

osxy\_triangle\_up\_down.Add(new Osxy(0.0f, 6.0f, 0.0f));

osxy\_triangle\_up\_down.Add(new Osxy(2.0f, 2.0f, 0.0f));

osxy\_triangle\_up\_down.Add(new Osxy(-2.0f, 2.0f, 0.0f));

osxy\_triangle\_left\_rigth.Add(new Osxy(2.0f, 2.0f, 0.0f));

osxy\_triangle\_left\_rigth.Add(new Osxy(3.0f, 0.0f, 0.0f));

osxy\_triangle\_left\_rigth.Add(new Osxy(2.0f, -2.0f, 0.0f));

}

private void trackBar1\_ValueChanged(object sender, EventArgs e)

{

angle\_x = trackBar1.Value;

drow\_figures();

}

private void trackBar2\_ValueChanged(object sender, EventArgs e)

{

angle\_y = trackBar2.Value;

drow\_figures();

}

private void trackBar3\_ValueChanged(object sender, EventArgs e)

{

angle\_z = trackBar3.Value;

drow\_figures();

}

private void radioButton2\_CheckedChanged(object sender, EventArgs e)

{

OpenGL gl = openGLControl1.OpenGL;

gl.LoadIdentity();

gl.Enable(OpenGL.GL\_DEPTH\_TEST);

float[] global\_ambient = new float[] { 0.5f, 0.5f, 0.5f, 1.0f };

float[] light0pos = new float[] { 1.0f, 1.0f, 1.0f, 1.0f };

float[] light0ambient = new float[] { 1.0f, 1.0f, 0.0f, 1.0f };

float[] light0difuse = new float[] { 0.3f, 0.3f, 0.3f, 1.0f };

float[] light0specular = new float[] { 0.8f, 0.8f, 0.8f, 1.0f };

float[] lmodel\_ambient = new float[] { 0.2f, 0.2f, 0.2f, 1.0f };

gl.LightModel(OpenGL.GL\_LIGHT\_MODEL\_AMBIENT, lmodel\_ambient);

gl.LightModel(OpenGL.GL\_LIGHT\_MODEL\_AMBIENT, global\_ambient);

gl.Light(OpenGL.GL\_LIGHT0, OpenGL.GL\_POSITION, light0pos);

gl.Light(OpenGL.GL\_LIGHT0, OpenGL.GL\_AMBIENT, light0ambient);

gl.Light(OpenGL.GL\_LIGHT0, OpenGL.GL\_DIFFUSE, light0difuse);

gl.Light(OpenGL.GL\_LIGHT0, OpenGL.GL\_SPECULAR, light0specular);

gl.Enable(OpenGL.GL\_LIGHTING);

gl.Enable(OpenGL.GL\_LIGHT0);

gl.ShadeModel(OpenGL.GL\_SMOOTH);

drow\_figures();

gl.Disable(OpenGL.GL\_LIGHTING);

gl.Disable(OpenGL.GL\_LIGHT0);

}

private void radioButton1\_CheckedChanged(object sender, EventArgs e)

{

drow\_figures();

}

Single angle\_x = 0, angle\_y = 0, angle\_z = 0;

private void drow\_figures()

{

OpenGL gl = this.openGLControl1.OpenGL;

gl.Clear(OpenGL.GL\_COLOR\_BUFFER\_BIT | OpenGL.GL\_DEPTH\_BUFFER\_BIT);

gl.Enable(OpenGL.GL\_DEPTH\_TEST);

gl.PushMatrix();

gl.Enable(OpenGL.GL\_LIGHT0);

gl.Translate(0, 0, -20.0f);

gl.Rotate(angle\_x, 1, 0, 0);

gl.Rotate(angle\_y, 0, 1, 0);

gl.Rotate(angle\_z, 0, 0, 1);

gl.Translate(0.0f, 0.0f, 2.5f);

gl.PolygonMode(OpenGL.GL\_FRONT\_AND\_BACK, OpenGL.GL\_FILL);

gl.Begin(OpenGL.GL\_POLYGON);

gl.Color(1.0f, 0.0f, 0.0f);

foreach (Osxy osxy in osxy\_triangle\_up\_down)

{

gl.Vertex(osxy.x, osxy.y, osxy.z);

}

gl.End();

gl.Begin(OpenGL.GL\_POLYGON);

gl.Color(1.0f, 1.0f, 0.0f);

foreach (Osxy osxy in osxy\_triangle\_up\_down)

{

gl.Vertex(osxy.x, -osxy.y, osxy.z);

}

gl.End();

gl.Begin(OpenGL.GL\_POLYGON);

gl.Color(1.0f, 0.0f, 0.0f);

foreach (Osxy osxy in osxy\_triangle\_up\_down)

{

gl.Vertex(osxy.x, osxy.y, osxy.z - 5);

}

gl.End();

gl.Begin(OpenGL.GL\_POLYGON);

gl.Color(1.0f, 0.0f, 0.0f);

foreach (Osxy osxy in osxy\_triangle\_up\_down)

{

gl.Vertex(osxy.x, -osxy.y, osxy.z - 5);

}

gl.End();

gl.Color(0.0f, 1.0f, 1.0f);

for (int i = osxy\_triangle\_up\_down.Count - 1; i > 0; i--)

{

gl.Begin(OpenGL.GL\_POLYGON);

gl.Vertex(osxy\_triangle\_up\_down[i].x, osxy\_triangle\_up\_down[i].y, osxy\_triangle\_up\_down[i].z);

gl.Vertex(osxy\_triangle\_up\_down[i].x, osxy\_triangle\_up\_down[i].y, osxy\_triangle\_up\_down[i].z - 5);

gl.Vertex(osxy\_triangle\_up\_down[i - 1].x, osxy\_triangle\_up\_down[i - 1].y, osxy\_triangle\_up\_down[i - 1].z - 5);

gl.Vertex(osxy\_triangle\_up\_down[i - 1].x, osxy\_triangle\_up\_down[i - 1].y, osxy\_triangle\_up\_down[i - 1].z);

gl.End();

}

gl.Begin(OpenGL.GL\_POLYGON);

gl.Vertex(osxy\_triangle\_up\_down[0].x, osxy\_triangle\_up\_down[0].y, osxy\_triangle\_up\_down[0].z);

gl.Vertex(osxy\_triangle\_up\_down[0].x, osxy\_triangle\_up\_down[0].y, osxy\_triangle\_up\_down[0].z - 5);

gl.Vertex(osxy\_triangle\_up\_down[2].x, osxy\_triangle\_up\_down[2].y, osxy\_triangle\_up\_down[2].z - 5);

gl.Vertex(osxy\_triangle\_up\_down[2].x, osxy\_triangle\_up\_down[2].y, osxy\_triangle\_up\_down[2].z);

gl.End();

gl.Color(0.0f, 1.0f, 1.0f);

for (int i = osxy\_triangle\_up\_down.Count - 1; i > 0; i--)

{

gl.Begin(OpenGL.GL\_POLYGON);

gl.Vertex(osxy\_triangle\_up\_down[i].x, -osxy\_triangle\_up\_down[i].y, osxy\_triangle\_up\_down[i].z);

gl.Vertex(osxy\_triangle\_up\_down[i].x, -osxy\_triangle\_up\_down[i].y, osxy\_triangle\_up\_down[i].z - 5);

gl.Vertex(osxy\_triangle\_up\_down[i - 1].x, -osxy\_triangle\_up\_down[i - 1].y, osxy\_triangle\_up\_down[i - 1].z - 5);

gl.Vertex(osxy\_triangle\_up\_down[i - 1].x, -osxy\_triangle\_up\_down[i - 1].y, osxy\_triangle\_up\_down[i - 1].z);

gl.End();

}

gl.Begin(OpenGL.GL\_POLYGON);

gl.Vertex(osxy\_triangle\_up\_down[0].x, -osxy\_triangle\_up\_down[0].y, osxy\_triangle\_up\_down[0].z);

gl.Vertex(osxy\_triangle\_up\_down[0].x, -osxy\_triangle\_up\_down[0].y, osxy\_triangle\_up\_down[0].z - 5);

gl.Vertex(osxy\_triangle\_up\_down[2].x, -osxy\_triangle\_up\_down[2].y, osxy\_triangle\_up\_down[2].z - 5);

gl.Vertex(osxy\_triangle\_up\_down[2].x, -osxy\_triangle\_up\_down[2].y, osxy\_triangle\_up\_down[2].z);

gl.End();

gl.Begin(OpenGL.GL\_POLYGON);

foreach (Osxy osxy in osxy\_triangle\_left\_rigth)

{

gl.Vertex(osxy.x, osxy.y, osxy.z);

}

gl.End();

gl.Begin(OpenGL.GL\_POLYGON);

gl.Color(1.0f, 1.0f, 0.0f);

foreach (Osxy osxy in osxy\_triangle\_left\_rigth)

{

gl.Vertex(-osxy.x, osxy.y, osxy.z);

}

gl.End();

gl.Begin(OpenGL.GL\_POLYGON);

gl.Color(1.0f, 0.0f, 0.0f);

foreach (Osxy osxy in osxy\_triangle\_left\_rigth)

{

gl.Vertex(osxy.x, osxy.y, osxy.z - 5);

}

gl.End();

gl.Begin(OpenGL.GL\_POLYGON);

gl.Color(1.0f, 0.0f, 0.0f);

foreach (Osxy osxy in osxy\_triangle\_left\_rigth)

{

gl.Vertex(-osxy.x, osxy.y, osxy.z - 5);

}

gl.End();

gl.Color(0.0f, 1.0f, 1.0f);

for (int i = osxy\_triangle\_left\_rigth.Count - 1; i > 0; i--)

{

gl.Begin(OpenGL.GL\_POLYGON);

gl.Vertex(-osxy\_triangle\_left\_rigth[i].x, osxy\_triangle\_left\_rigth[i].y, osxy\_triangle\_left\_rigth[i].z);

gl.Vertex(-osxy\_triangle\_left\_rigth[i].x, osxy\_triangle\_left\_rigth[i].y, osxy\_triangle\_left\_rigth[i].z - 5);

gl.Vertex(-osxy\_triangle\_left\_rigth[i - 1].x, osxy\_triangle\_left\_rigth[i - 1].y, osxy\_triangle\_left\_rigth[i - 1].z - 5);

gl.Vertex(-osxy\_triangle\_left\_rigth[i - 1].x, osxy\_triangle\_left\_rigth[i - 1].y, osxy\_triangle\_left\_rigth[i - 1].z);

gl.End();

}

gl.Begin(OpenGL.GL\_POLYGON);

gl.Vertex(-osxy\_triangle\_left\_rigth[0].x, osxy\_triangle\_left\_rigth[0].y, osxy\_triangle\_left\_rigth[0].z);

gl.Vertex(-osxy\_triangle\_left\_rigth[0].x, osxy\_triangle\_left\_rigth[0].y, osxy\_triangle\_left\_rigth[0].z - 5);

gl.Vertex(-osxy\_triangle\_left\_rigth[2].x, osxy\_triangle\_left\_rigth[2].y, osxy\_triangle\_left\_rigth[2].z - 5);

gl.Vertex(-osxy\_triangle\_left\_rigth[2].x, osxy\_triangle\_left\_rigth[2].y, osxy\_triangle\_left\_rigth[2].z);

gl.End();

gl.Color(0.0f, 1.0f, 1.0f);

for (int i = osxy\_triangle\_left\_rigth.Count - 1; i > 0; i--)

{

gl.Begin(OpenGL.GL\_POLYGON);

gl.Vertex(osxy\_triangle\_left\_rigth[i].x, osxy\_triangle\_left\_rigth[i].y, osxy\_triangle\_left\_rigth[i].z);

gl.Vertex(osxy\_triangle\_left\_rigth[i].x, osxy\_triangle\_left\_rigth[i].y, osxy\_triangle\_left\_rigth[i].z - 5);

gl.Vertex(osxy\_triangle\_left\_rigth[i - 1].x, osxy\_triangle\_left\_rigth[i - 1].y, osxy\_triangle\_left\_rigth[i - 1].z - 5);

gl.Vertex(osxy\_triangle\_left\_rigth[i - 1].x, osxy\_triangle\_left\_rigth[i - 1].y, osxy\_triangle\_left\_rigth[i - 1].z);

gl.End();

}

gl.Begin(OpenGL.GL\_POLYGON);

gl.Vertex(osxy\_triangle\_left\_rigth[0].x, osxy\_triangle\_left\_rigth[0].y, osxy\_triangle\_left\_rigth[0].z);

gl.Vertex(osxy\_triangle\_left\_rigth[0].x, osxy\_triangle\_left\_rigth[0].y, osxy\_triangle\_left\_rigth[0].z - 5);

gl.Vertex(osxy\_triangle\_left\_rigth[2].x, osxy\_triangle\_left\_rigth[2].y, osxy\_triangle\_left\_rigth[2].z - 5);

gl.Vertex(osxy\_triangle\_left\_rigth[2].x, osxy\_triangle\_left\_rigth[2].y, osxy\_triangle\_left\_rigth[2].z);

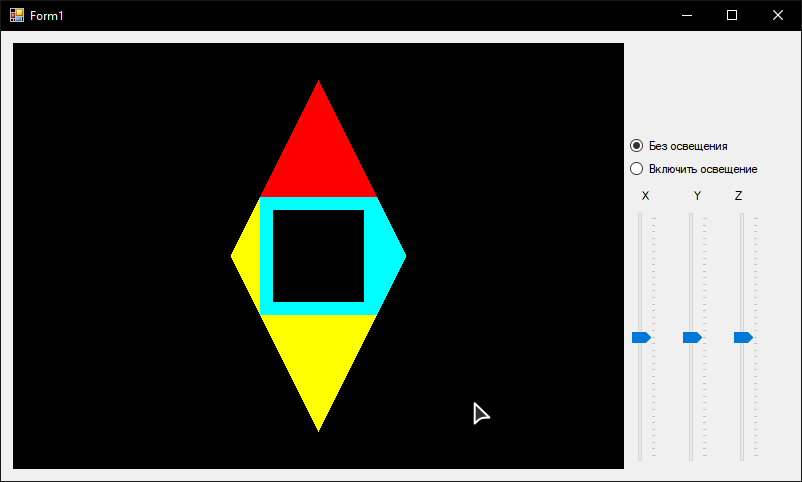
gl.End();

gl.PopMatrix();

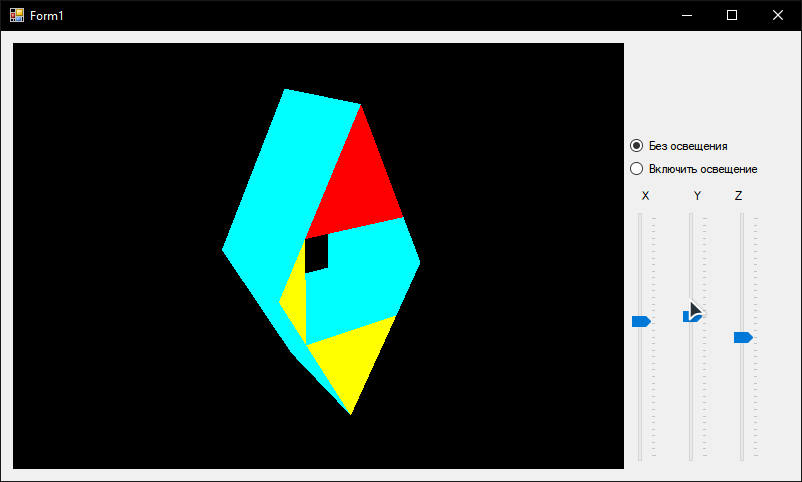
}

}

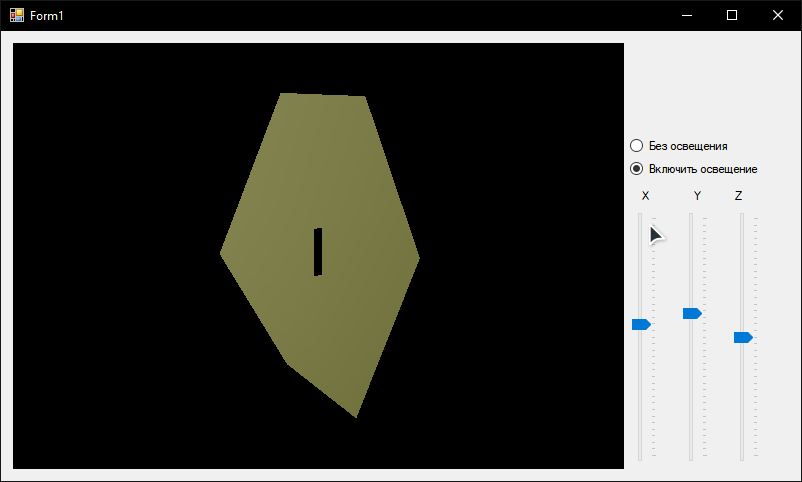
}



Скриншот 1 : запуск программы.



Скриншот 2 : поворот фигуры.



Скриншот 3: Включение освещения

**Вывод:** в ходе выполнения работы получили теоретические и практические знания по созданию моделей освещенности.