

# Компьютерная графика

Шлеймович Михаил Петрович

# Темы

1. Введение в компьютерную графику
2. Преобразования модели
3. Преобразования координат
4. Преобразования проецирования
5. Растровая графика
6. Трехмерное моделирование
7. Модели освещения
8. Текстура

# Литература

1. Баяковский Ю.М., Игнатенко А.В. Начальный курс OpenGL. – М.: «Планета знаний», 2007
2. Божко А.Н., Жук Д.М., Маничев В.Б. Компьютерная графика: Учеб. пособие для вузов. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2007.
3. Боресков А.В. Графика трехмерной компьютерной игры на основе OpenGL. – М.: ДИАЛОГ-МИФИ, 2004.
4. Ву М., Девис Т., Нейдер Дж., Шрайнер Д. OpenGL. Руководство по программированию. Библиотека программиста. 4-е издание. – СПб.: Питер, 2006.
5. Евченко А.И. OpenGL и DirectX: программирование графики. Для профессионалов (+CD). – СПб.: Питер, 2006.
6. Никулин Е.А. Компьютерная геометрия и алгоритмы машинной графики. – СПб.: БХВ-Петербург, 2003.
7. Райт Р.С., Липчак Б. OpenGL. Суперкнига, 3-издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2006.

# Литература

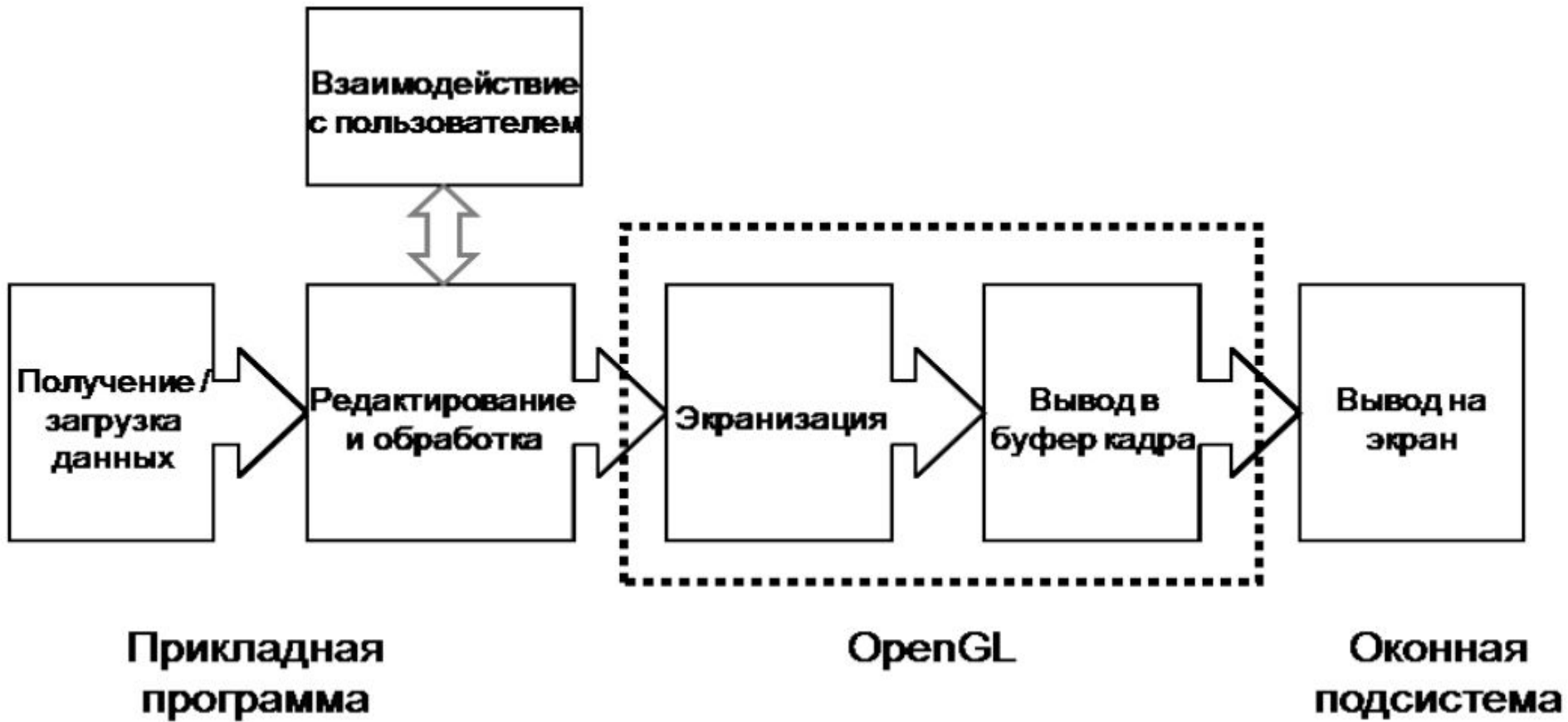
8. Роджерс Д. Алгоритмические основы машинной графики: Пер. с англ. – М.: Мир, 1989.
9. Роджерс Д., Адамс Дж. Математические основы машинной графики: Пер. с англ. – М.: Мир, 2001.
0. Роджерс Д., Адамс Дж. Математические основы машинной графики. – М.: Машиностроение, 1980.
1. Сиденко Л.А. Компьютерная графика и геометрическое моделирование: Учебное пособие. – СПб.: Питер, 2009.
2. Херн Д., Бейкер М. Паулин. Компьютерная графика и стандарт OpenGL, 3-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2005.
3. Хилл Ф. OpenGL. Программирование компьютерной графики. Для профессионалов. – СПб.: Питер, 2002.
4. Эйнджел Э. Интерактивная компьютерная графика. Вводный курс на базе OpenGL, 2-е изд.: Пер. с англ. – М.: Издательский дом «Вильямс», 2001.

# Учебное пособие

Набережнов Г.М., Максимов Н.Н.  
Компьютерная геометрия и графика:  
Учебное пособие / Под ред. канд. техн.  
наук Г.М. Набережнова. Казань: Изд-во  
Казан. гос. техн. ун-та, 2009.

# 1. Введение в компьютерную графику

# Графический процесс

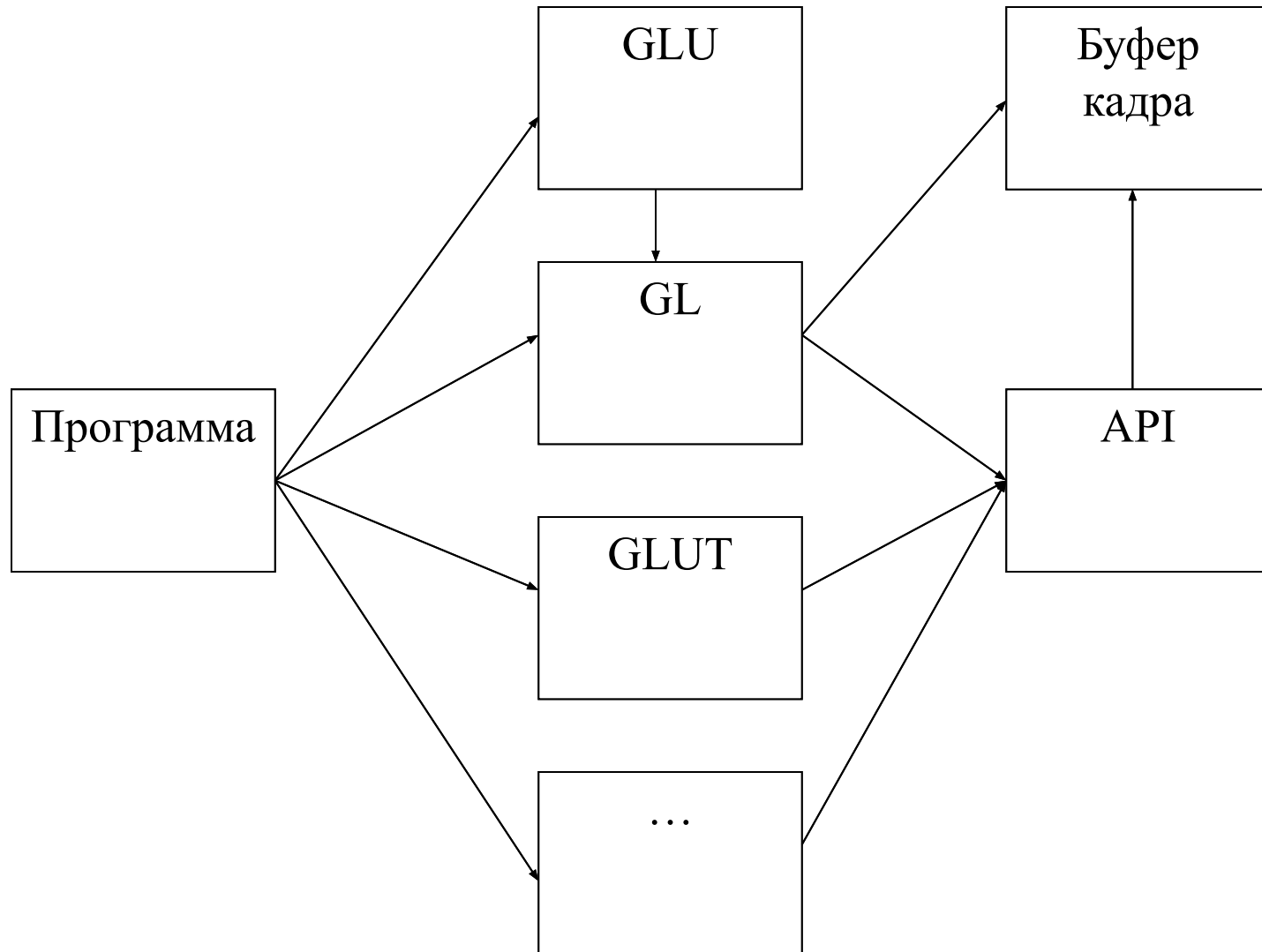


# OpenGL

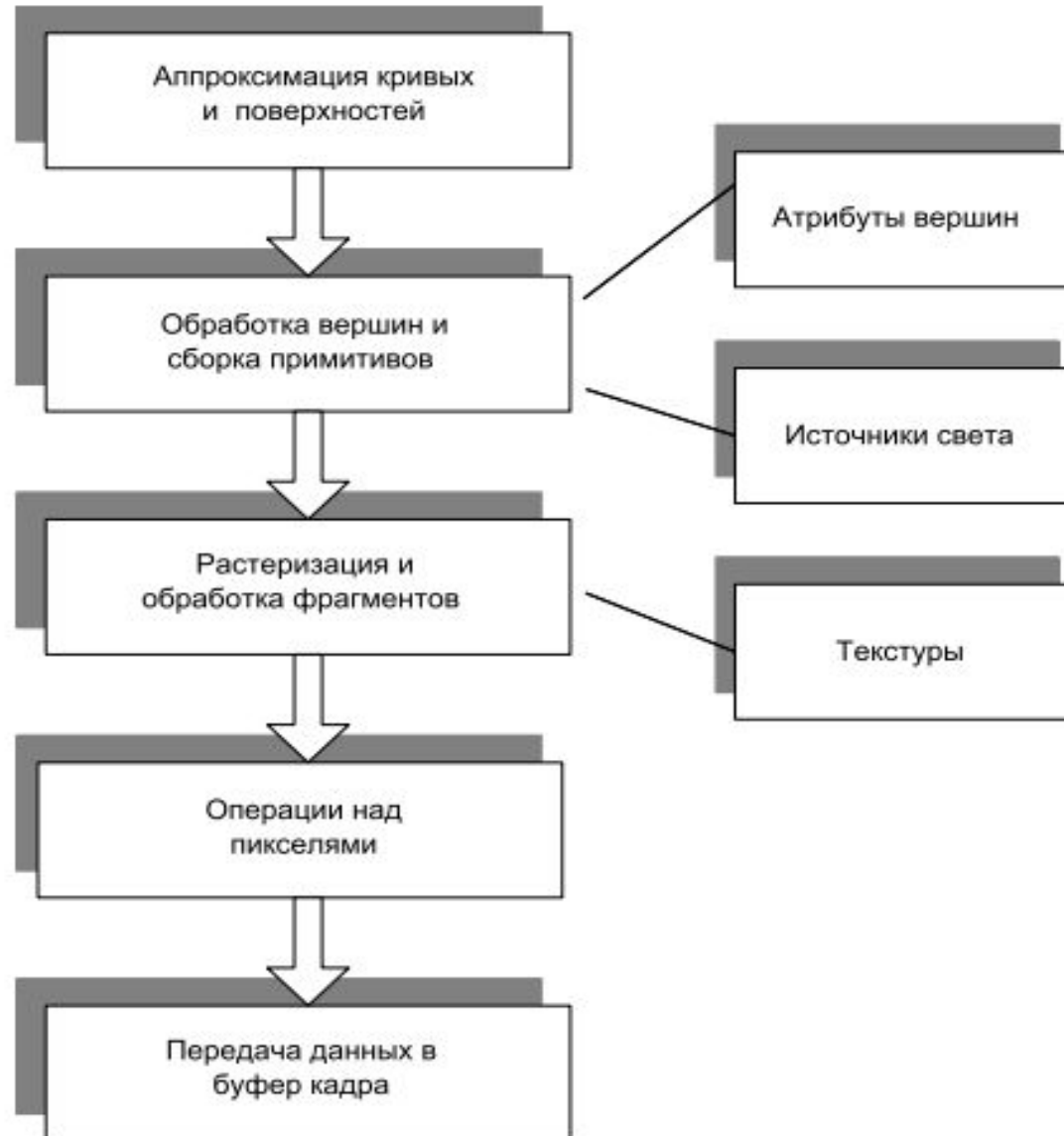
<http://www.opengl.org>



# OpenGL



# Конвейер OpenGL



# Пример программы 1

```
#include <GL/glut.h>
void init();
void draw();
void main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(640, 480);
    glutInitWindowPosition(50, 50);
    glutCreateWindow("My program");
    init();
    glutDisplayFunc(draw);
    glutMainLoop();
}
```

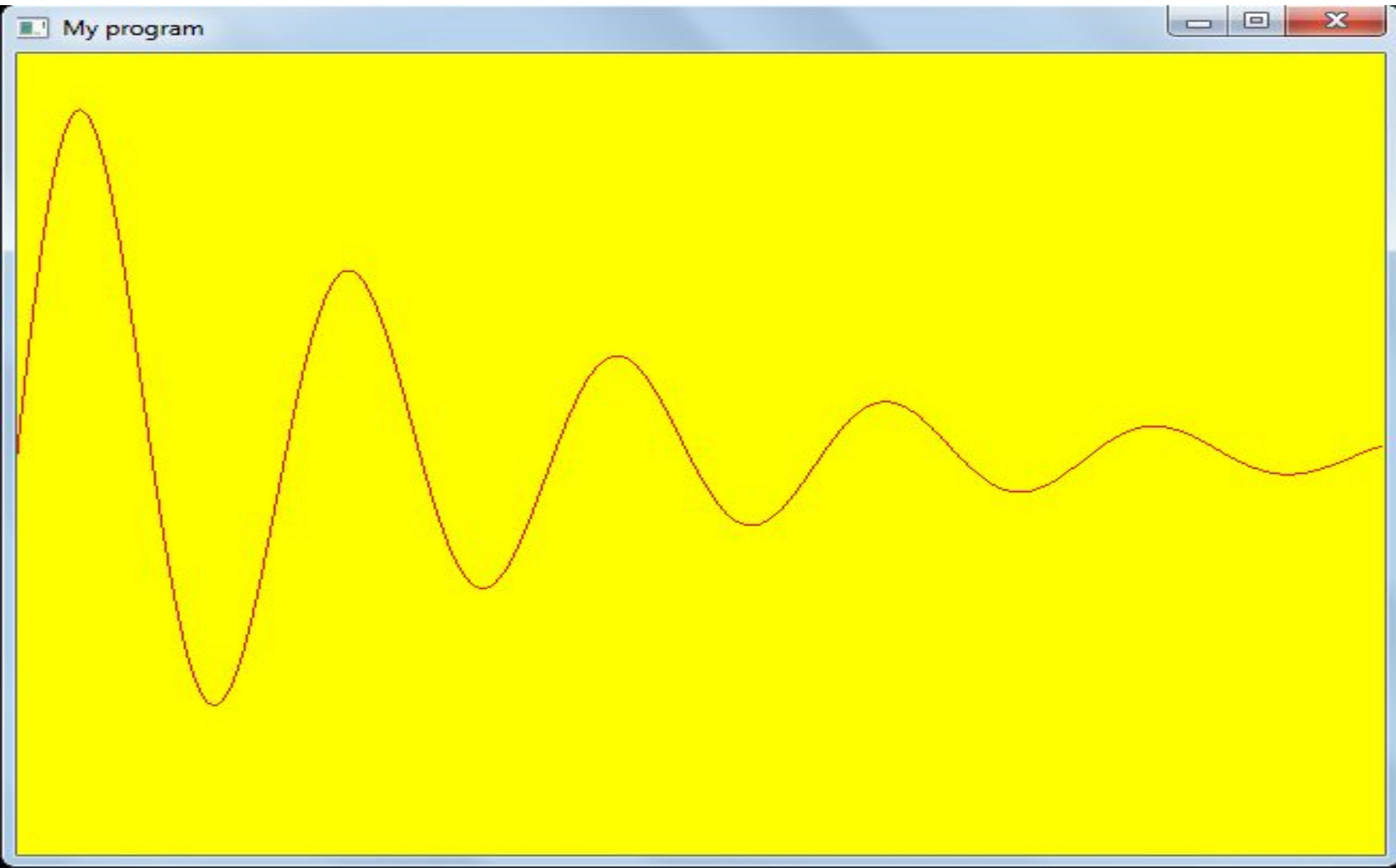
# Пример программы 1

```
void init(void)
{
    glClearColor(1.0,1.0,0.0,0.0);
    glClear(GL_COLOR_BUFFER_BIT);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0, 640, 0, 480);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}
```

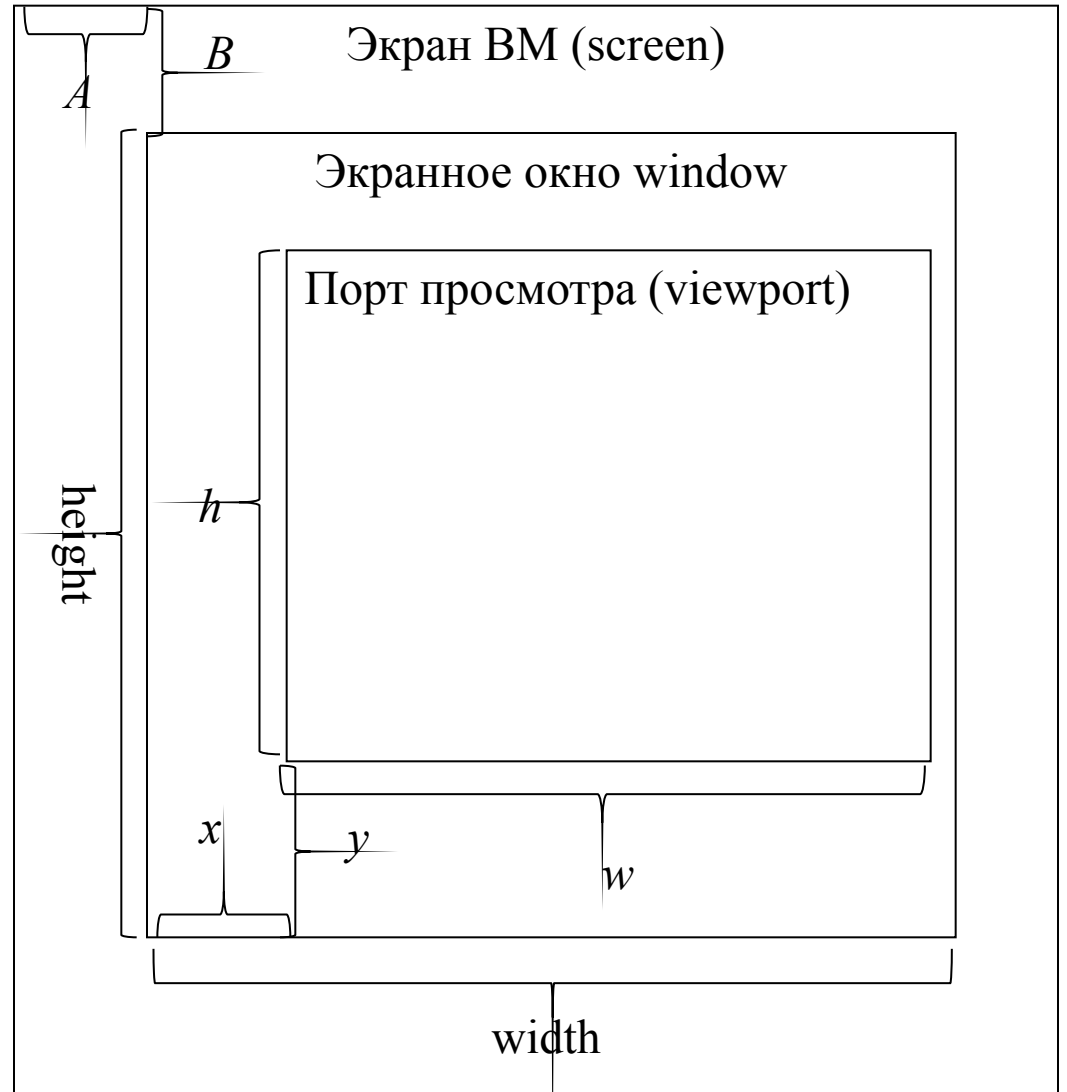
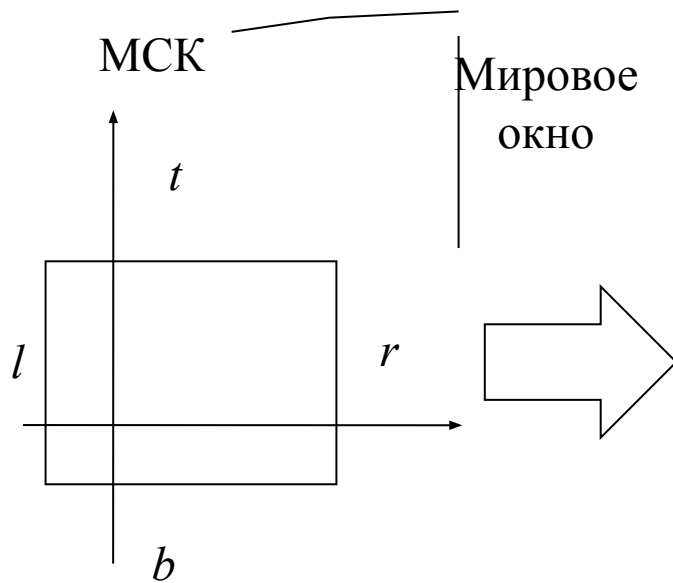
# Пример программы 1

```
void draw(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glViewport(0, 0, 640, 480);
    glColor3f(1.0, 0.0, 0.0);
    glBegin(GL_LINE_STRIP);
    GLfloat x, y;
    for(x = 0; x < 640; x++)
    {
        y = 240 * (1 + sin(0.05 * x) * exp(-0.005 * x));
        glVertex2f(x, y);
    }
    glEnd();
    glFlush();
}
```

# Пример программы 1



# OpenGL



# Типы OpenGL

Суффикс	Тип	Имя
b	8 – бит, целое	GLbyte
s	16 – бит, целое	GLshort
i	32 – бит, целое	GLint, GLsizei
f	32 – бит, вещественное	GLfloat, GLclampf
d	64 – бит, вещественное	GLdouble, GLclampd
ub	8 – бит, целое, без знака	GLubyte, GLboolean
us	16 – бит, целое, без знака	GLushort
ui	32 – бит, целое, без знака	GLuint, GLenum, GLbitfield



# Примитивы

GL_POINTS	Обрабатывает каждую вершину как отдельную точку. Вершина $n$ определяет точку $n$ .
GL_LINES	Обрабатывает каждую пару вершин как независимый линейный сегмент. Вершины $2n-1$ и $2n$ определяют прямую $n$ . Рисуется $N/2$ прямых.
GL_LINE_STRIP	Рисует связанную группу линейных сегментов от первой вершины до последней. Вершины $n$ и $n+1$ определяют прямую $n$ . Рисуется $N-1$ прямых.
GL_LINE_LOOP	Рисует связанную группу линейных сегментов от первой вершины до последней, а затем назад к первой. Вершины $n$ и $n+1$ определяют прямую $n$ . Последняя прямая определена вершинами 1 и $N$ . Рисуется $N$ прямых.
GL_TRIANGLES	Обрабатывает тройку вершин как независимый треугольник. Вершины $3n-2$ , $3n-1$ и $3n$ определяют треугольник $n$ . Рисуется $N/3$ треугольников.

# Примитивы

GL_TRIANGLE_STRIP	Рисует связанную группу треугольников. Для нечетного значения $n$ вершины $n$ , $n+1$ и $n+2$ определяют треугольник $n$ . Для четного значения $n$ вершины $n+1$ , $n$ и $n+2$ определяют треугольник $n$ . Рисуются $N-2$ треугольников.
GL_TRIANGLE_FAN	Рисует связанную группу треугольников. Вершины $1$ , $n+1$ и $n+2$ определяют треугольник $n$ . Рисуются $N-2$ треугольников.
GL_QUADS	Обрабатывает каждую группу из четырех вершин в качестве независимого четырехугольника. Вершины $4n-3$ , $4n-2$ , $4n-1$ и $4n$ определяют четырехугольник $n$ . Рисуются $N/4$ четырехугольников.
GL_QUAD_STRIP	Рисует связанную группу четырехугольников. Вершины $2n-1$ , $2n$ , $2n+2$ и $2n+1$ определяют четырехугольник $n$ . Рисуются $N/2-1$ четырехугольников.
GL_POLYGON	Рисует отдельный выпуклый многоугольник.

# Пример программы 2

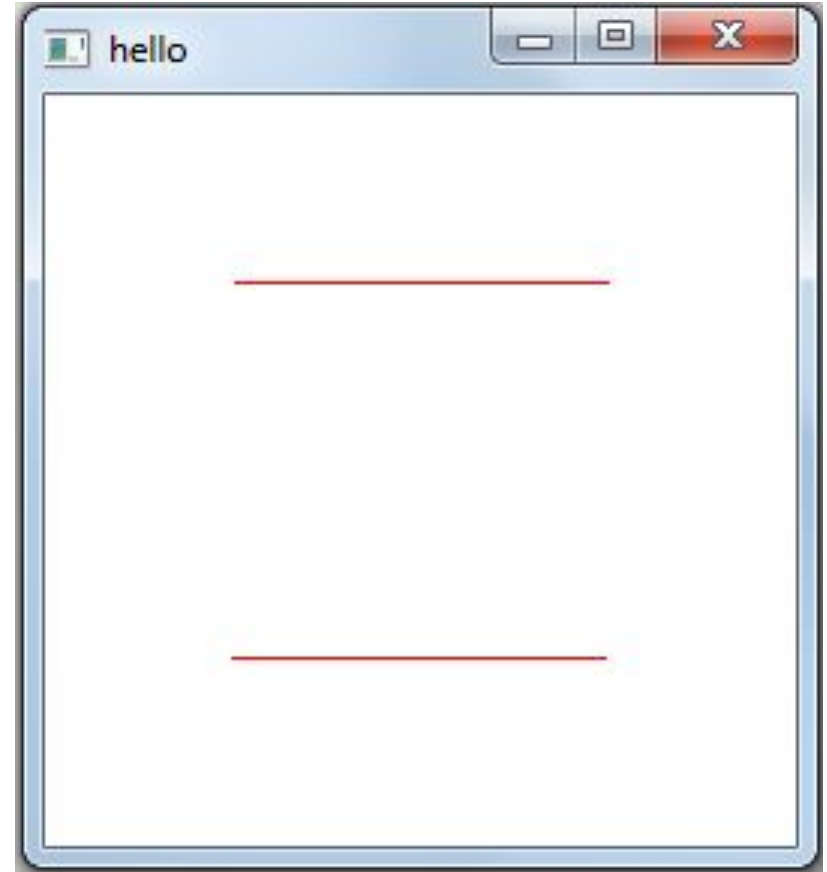
```
#include <GL/glut.h>
void init (void);
void display(void);
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize (250, 250);
    glutInitWindowPosition (100, 100);
    glutCreateWindow ("hello");
    init ();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

# Пример программы 2

```
void init (void)
{
    glClearColor (1.0, 1.0, 1.0, 0.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);
}
```

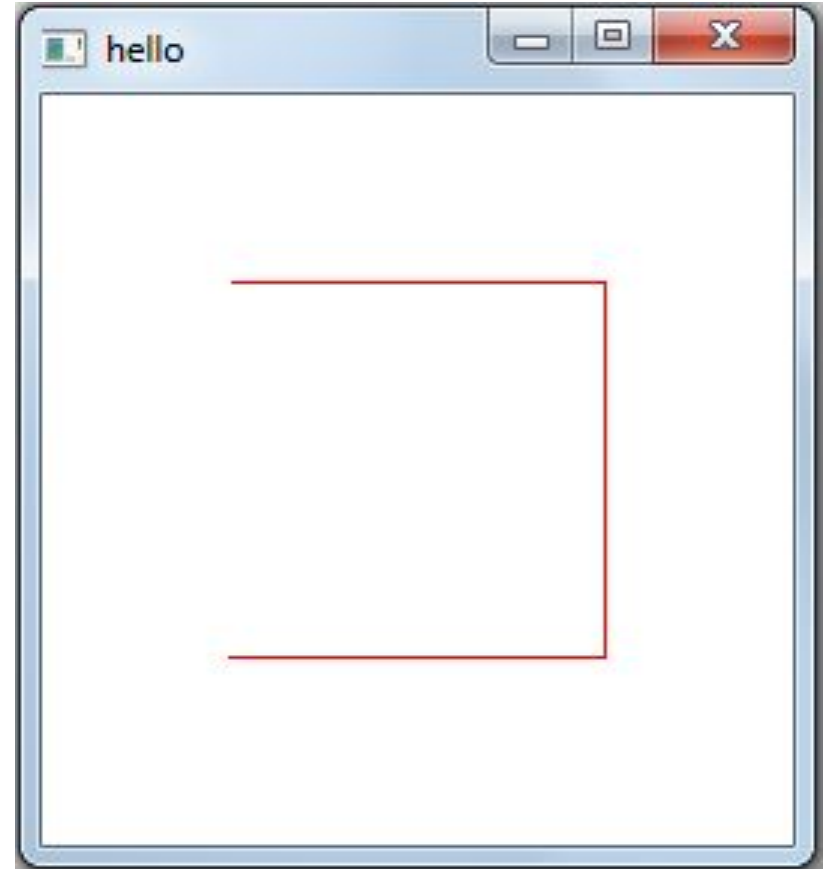
# Пример программы 2

```
void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glBegin(GL_LINES);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```



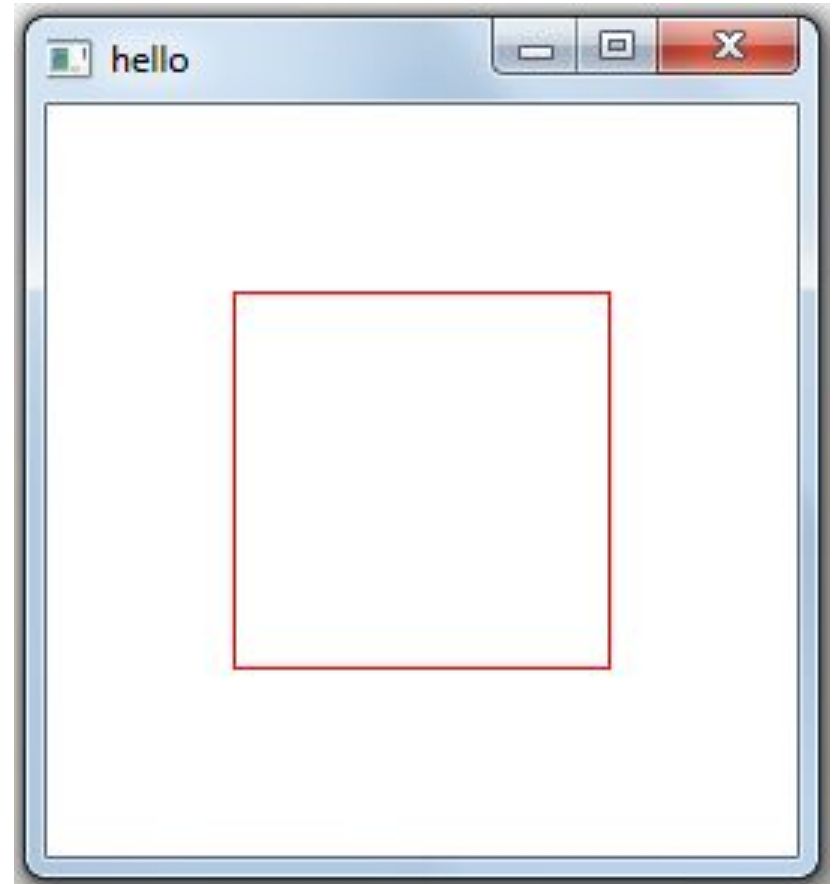
# Пример программы 2

```
void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glBegin(GL_LINE_STRIP);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```



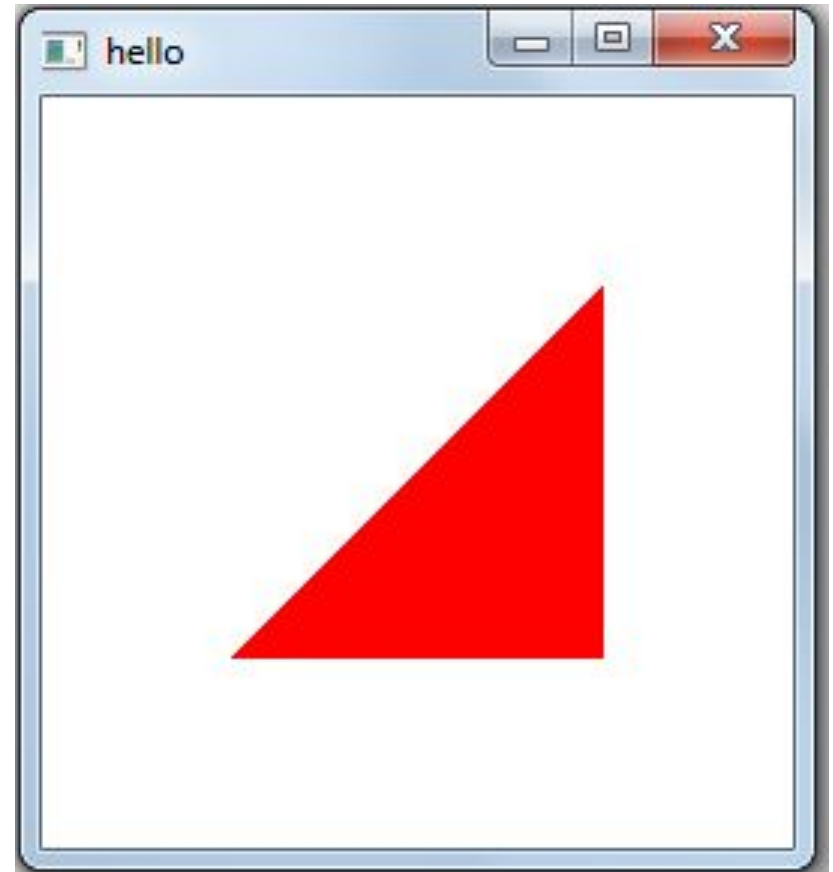
# Пример программы 2

```
void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glBegin(GL_LINE_LOOP);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```



# Пример программы 2

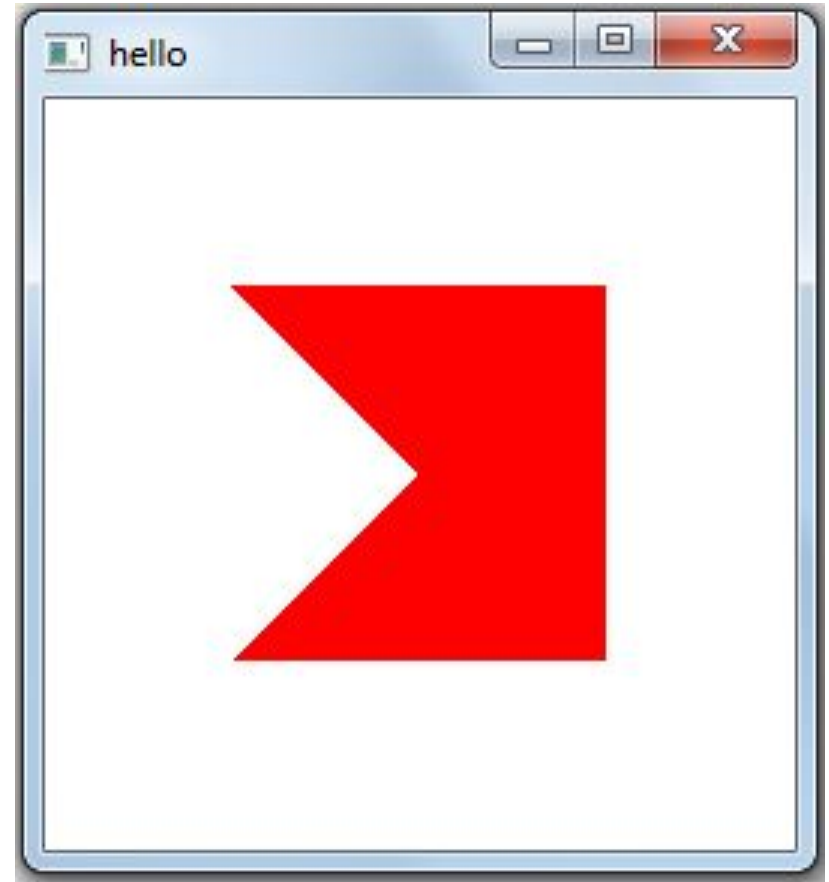
```
void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```





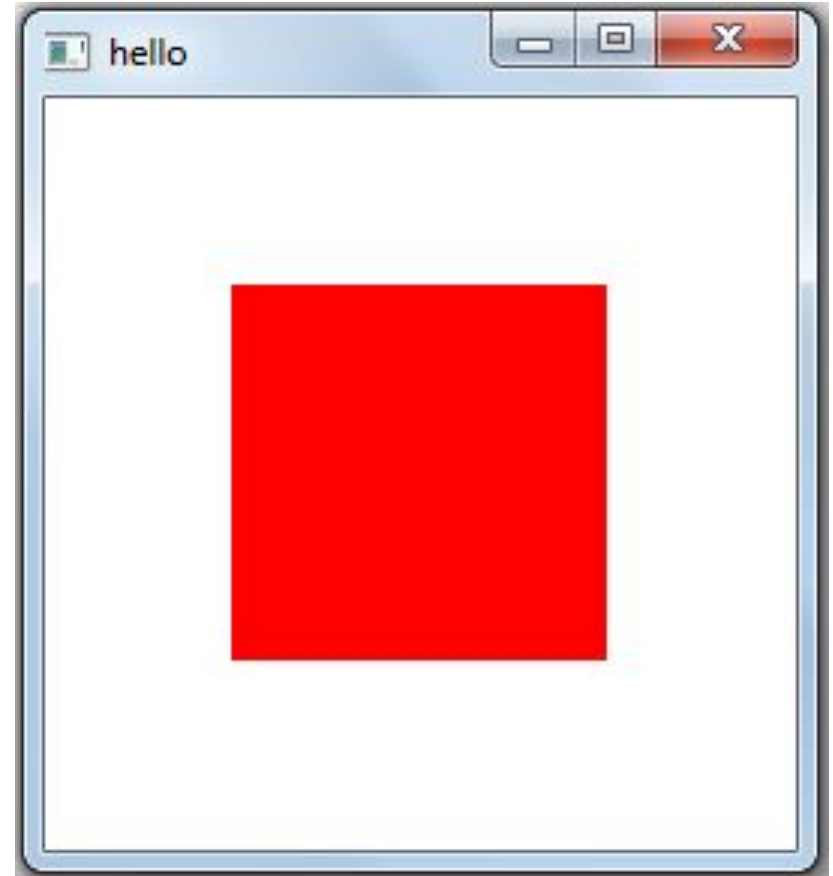
# Пример программы 2

```
void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glBegin(GL_TRIANGLE_STRIP);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```



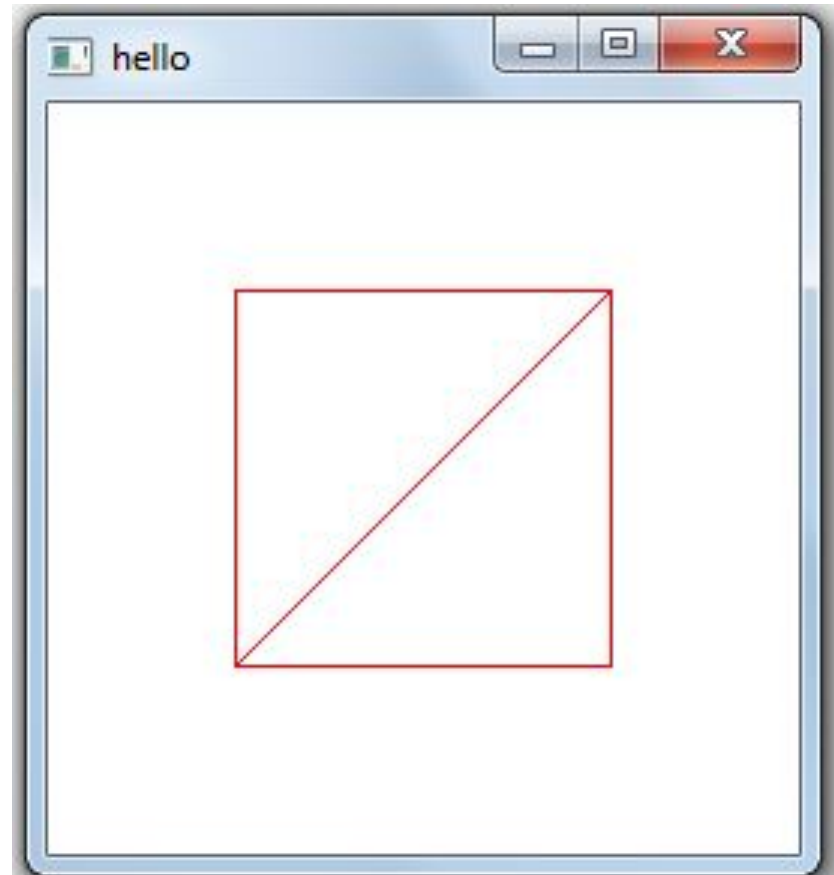
# Пример программы 2

```
void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glBegin(GL_TRIANGLE_FAN);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```



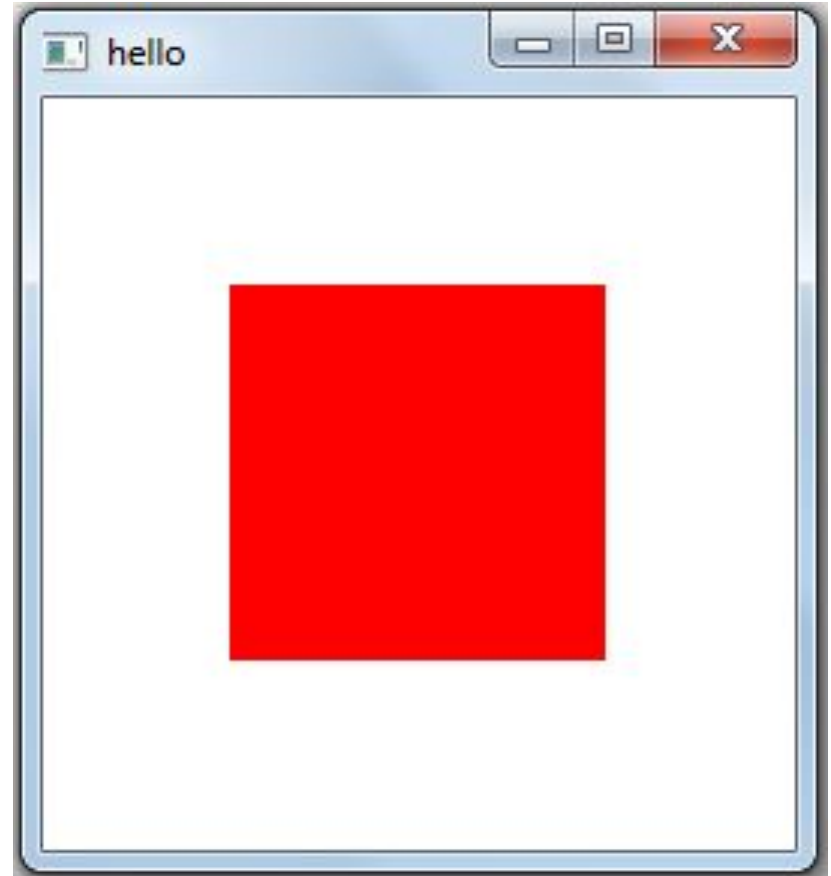
# Пример программы 2

```
void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glPolygonMode(GL_FRONT, GL_LINE);
    glBegin(GL_TRIANGLE_FAN);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```



# Пример программы 2

```
void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glBegin(GL_POLYGON);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```



# События

```
void glutDisplayFunc(void (*func)(void));  
void glutPostRedisplay(void);  
void glutIdleFunc(void (*func)(void));  
void glutReshapeFunc(void (*func)(int width, int height));  
void glutKeyboardFunc(void (*func)(unsigned char key, int x, int y));  
void glutSpecialFunc(void (*func)(int key, int x, int y));  
    GLUT_KEY_F1  
    ...  
    GLUT_KEY_F12  
    GLUT_KEY_LEFT  
    GLUT_KEY_UP,  
    GLUT_KEY_RIGHT  
    GLUT_KEY_DOWN  
    GLUT_KEY_PAGE_UP  
    GLUT_KEY_PAGE_DOWN  
    GLUT_KEY_HOME  
    GLUT_KEY_END  
    GLUT_KEY_INSERT
```

# События

```
void glutMouseFunc(void (*func)(int button, int state, int x, int y));
```

button:

GLUT\_LEFT\_BUTTON

GLUT\_MIDDLE\_BUTTON

GLUT\_RIGHT\_BUTTON

state:

GLUT\_UP

GLUT\_DOWN

```
void glutMotionFunc(void (*func)(int x, int y));
```

```
void glutPassiveMotionFunc(void (*func)(int x, int y));
```

# Пример программы 3

```
#include <GL/glut.h>
void init (void);
void display(void);
void keyboard(unsigned char, int, int);
void reshape (int, int);
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize (250, 250);
    glutInitWindowPosition (100, 100);
    glutCreateWindow ("hello");
    init ();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutKeyboardFunc(keyboard);
    glutMainLoop();
    return 0;
}
```

# Пример программы 3

```
void init (void)
{
    glClearColor (1.0, 1.0, 1.0, 0.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);
}

void display(void)
{
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f (1.0, 0.0, 0.0);
    glPolygonMode(GL_FRONT, GL_LINE);
    glBegin(GL_POLYGON);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}
```

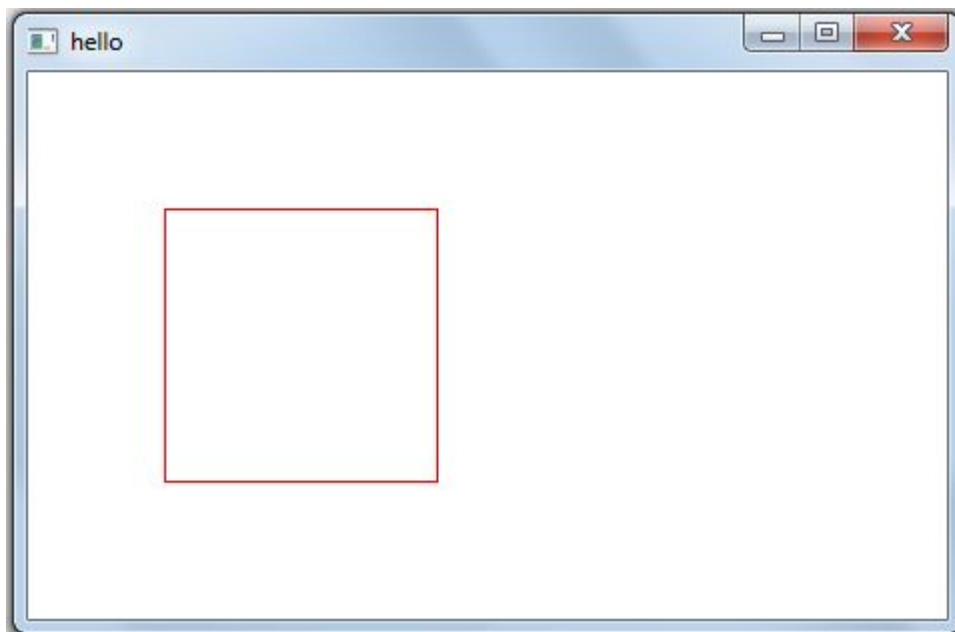
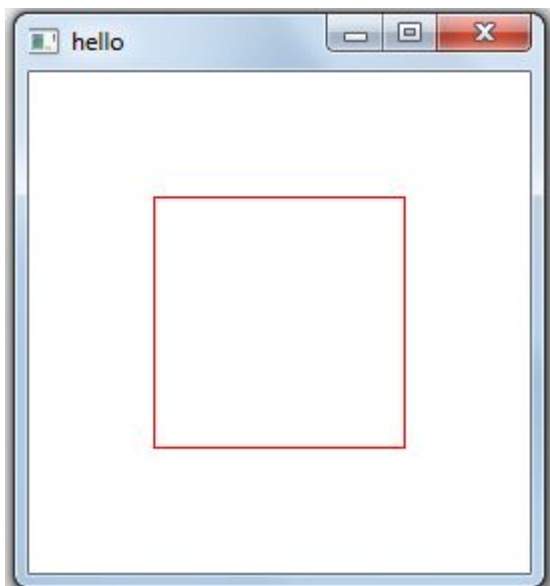


# Пример программы 3

```
void keyboard(unsigned char key, int x, int y)
{
    switch (key)
    {
        case 27:
            exit(0);
            break;
    }
}
```

```
void reshape (int w, int h)
{
    if (w / h < 1) glViewport(0, 0, w, w);
    else glViewport(0, 0, h, h);
}
```

# Пример программы 3



# Пример программы 4

```
#include <GL/glut.h>
#include <windows.h>
#include <GL/glut.h>
#define _USE_MATH_DEFINES
#include <math.h>
float R    = 150.0;
float angle = 0.0;
float X0   = 0.0;
float Y0   = 0.0;
int  n     = 3;
int  width = 640;
int  height= 480;
```

# Пример программы 4

```
void figure(void)
{
    glBegin(GL_LINE_LOOP);
    for(int i = 0; i < n; i++)
    {
        float x = X0 + R * cos(2 * M_PI * i / n + angle);
        float y = Y0 + R * sin(2 * M_PI * i / n + angle);
        glVertex2f(x, y);
    }
    glEnd();
}

void init(void)
{
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-320, 320, -240, 240);
}
```

# Пример программы 4

```
void display(void)
{
    glClearColor(1.0, 1.0, 1.0, 0.0);
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0, 0.0, 0.0);
    figure();
    glFlush();
}

void keyboard(unsigned char key, int x, int y)
{
    float min = ((width < height) ? width : height) / 2;
    if(key == '+' && R < min) R++;
    if(key == '-' && R > 0) R--;
    glutPostRedisplay();
}
```

# Пример программы 4

```
void special(int key, int x, int y)
{  switch(key)
   { case GLUT_KEY_F1:    n = 3; break;
     case GLUT_KEY_F2:    n = 4; break;
     case GLUT_KEY_F3:    n = 5; break;
     case GLUT_KEY_F4:    n = 6; break;
     case GLUT_KEY_F5:    n = 7; break;
     case GLUT_KEY_F6:    n = 8; break;
     case GLUT_KEY_F7:    n = 9; break;
     case GLUT_KEY_F8:    n = 10; break;
     case GLUT_KEY_F9:    n = 11; break;
     case GLUT_KEY_F10:   n = 12; break;
     case GLUT_KEY_F11:   n = 13; break;
     case GLUT_KEY_F12:   n = 14; break;
     case GLUT_KEY_LEFT:  if(X0 > -width / 2) X0--; break;
     case GLUT_KEY_RIGHT: if(X0 < width / 2) X0++; break;
     case GLUT_KEY_UP:    if(Y0 < height / 2) Y0++; break;
     case GLUT_KEY_DOWN:  if(Y0 > -height / 2) Y0--; break;
   }
  glutPostRedisplay();
}
```

# Пример программы 4

```
void reshape(int w, int h)
{
    width = w;
    height = h;
    glViewport(0, 0, w, h);
    glutPostRedisplay();
}

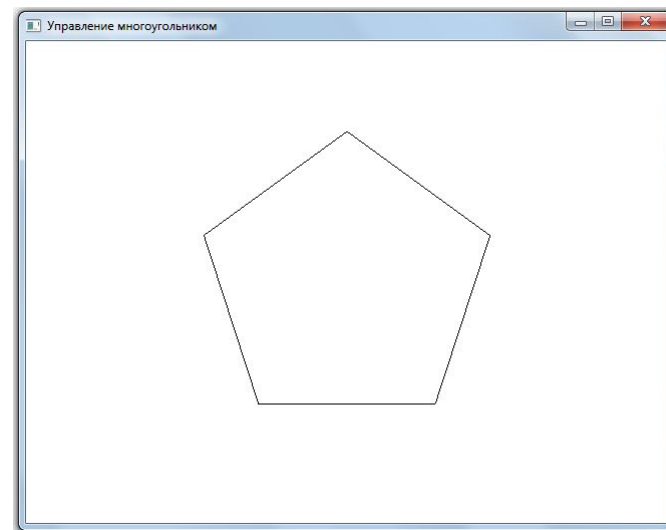
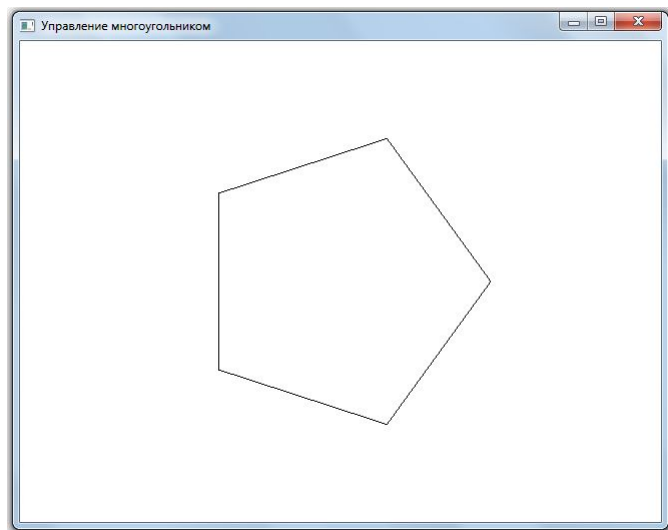
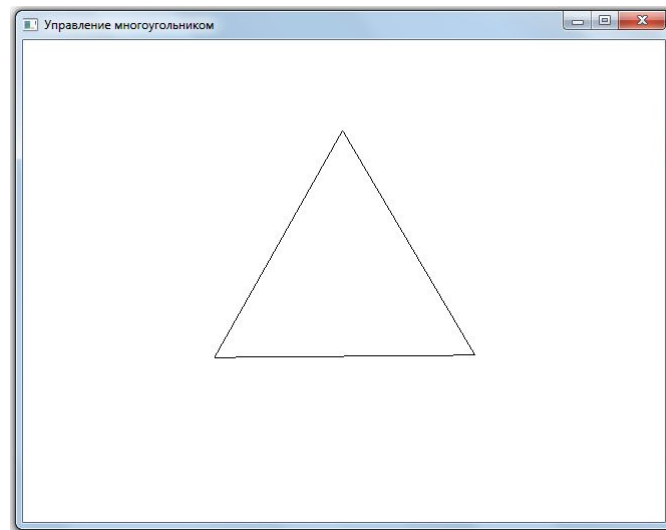
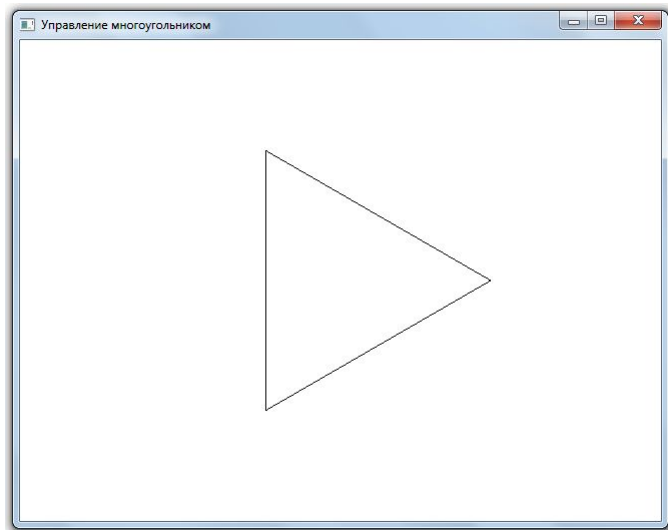
void mouse(int button, int state, int x, int y)
{
    if(button == GLUT_LEFT_BUTTON && state == GLUT_UP)
    {
        X0 = 640.0 / width * ( x - width / 2);
        Y0 = 480.0 / height * (-y + height / 2);
    }
    if(button == GLUT_RIGHT_BUTTON && state == GLUT_UP)
    {
        angle += 0.01;
        if(angle >= 360.0) angle = 0.0;
    }
    glutPostRedisplay();
}
```

# Пример программы 4

```
void main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(640, 480);
    glutInitWindowPosition(0,0);
    glutCreateWindow("Управление многоугольником");
    glutDisplayFunc(display);
    glutKeyboardFunc(keyboard);
    glutSpecialFunc(special);
    glutMouseFunc(mouse);
    glutReshapeFunc(reshape);
    init();
    glutMainLoop();
}
```



# Пример программы 3



# OpenTK

<http://www.opentk.com>

## Windows.Form

# Пример программы 5

```
using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Windows.Forms;  
  
using OpenTK.Graphics;  
using OpenTK.Graphics.OpenGL;
```

# Пример программы 5

```
namespace ExampleOpenTK
{
    public partial class OpenTK_Form : Form
    {
        public OpenTK_Form()
        {
            InitializeComponent();
        }

        private void glControl1_Load(object sender, EventArgs e)
        {
            GL.ClearColor(1.0f, 1.0f, 0.0f, 0.0f);
            GL.MatrixMode(MatrixMode.Projection);
            GL.LoadIdentity();
            GL.Ortho(0, this.ClientSize.Width, 0, this.ClientSize.Height, -1, 1);
            GL.MatrixMode(MatrixMode.Modelview);
            GL.LoadIdentity();
        }
    }
}
```

# Пример программы 5

```
private void glControl1_Paint(object sender, PaintEventArgs e)
{
    GL.Clear(ClearBufferMask.ColorBufferBit);
    GL.Viewport(0, 0, this.ClientSize.Width, this.ClientSize.Height);
    GL.Color3(1.0, 0.0, 0.0);
    GL.Begin(BeginMode.LineStrip);
    float x, y;
    for (x = 0; x < this.Width; x++)
    {
        y = this.ClientSize.Height / 2 * (1 + (float)Math.Sin(0.05 * x) *
            (float)Math.Exp(-0.005 * x));
        GL.Vertex2(x, y);
    }
    GL.End();
    glControl1.SwapBuffers();
}
```

# Пример программы 5

```
private void glControl1_Resize(object sender, EventArgs e)
{
    GL.ClearColor(1.0f, 1.0f, 0.0f, 0.0f);
    GL.MatrixMode(MatrixMode.Projection);
    GL.LoadIdentity();
    GL.Ortho(0, this.ClientSize.Width, 0, this.ClientSize.Height, -1, 1);
    GL.MatrixMode(MatrixMode.Modelview);
    GL.LoadIdentity();
}
}
```

# Пример программы 5

