## Sample codes

# First OpenGL code

```
•  #include<GL/freeglut.h>
• #include<GL/gl.h>


• int main(int argc, char** argv)
• {
•     glutInit(&argc, argv);
•     glutInitDisplayMode(GLUT_SINGLE|GLUT_RGBA);
•     glutInitWindowSize(500,500);
•     glutInitWindowPosition(100,100);
•     glutCreateWindow("OpenGL - First window demo");
```

## Draw in the window

```
#include<GL/freeglut.h>
#include<GL/gl.h>

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGBA);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(100,100);
    glutCreateWindow("OpenGL - First window demo");

    glBegin(GL_POLYGON);
       glVertex2f(-0.5,-0.5);
       glVertex2f(-0.5,0.5);
       glVertex2f(0.5,0.5);
       glVertex2f(0.5,-0.5);
    glEnd();

    glFlush();
    glutMainLoop();
    return 0;
}
```

```
#include<GL/freeglut.h>
#include<GL/gl.h>
void PolygonDisplay()
{
glBegin(GL_POLYGON);
      glVertex2f(-0.5,-0.5);
      glVertex2f(-0.5,0.5);
      glVertex2f(0.5,0.5);
      glVertex2f(0.5,-0.5);
   glEnd();
   glFlush();
}
```

```
int main(int argc, char** argv)
{   glutInit(&argc, argv);

glutInitDisplayMode(GLUT_SINGLE|GLUT_RGBA)
;
    glutInitWindowSize(500,500);
    glutInitWindowPosition(100,100);
    glutCreateWindow("OpenGL - First window
demo");
    glutDisplayFunc(PolygonDisplay);

    glutMainLoop();
    Return 0; }
```

```
#include<GL/gl.h>
void PolygonDisplay()
{
glClearColor(0.0,0.0,0.0,0.0);
glClear(GL_COLOR_BUFFER_BIT);
glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);
      glVertex2f(-0.5,-0.5);
      glVertex2f(-0.5,0.5);
      glVertex2f(0.5,0.5);
      glVertex2f(0.5,-0.5);
   glEnd();
   glFlush();
}
```

## glClearColor

glClearColor specifies the red, green, blue, and alpha values used by glClear to clear the color buffers. Values specified by glClearColor are clamped to the range 0 - 1 .

Syntax

```
void glClearColor(GLclampf red, GLclampf green, GLclampf blue, GLclampf alpha);
```

Usage

```
glClearColor(0.0, 0.0, 0.0, 0.0);
```

## glClear

glClear sets the bitplane area of the window to values previously selected by glClearColor, glClearDepthf, and glClearStencil.

Syntax

```
void glClear(GLbitfield mask);
```

mask

Bitwise OR of masks that indicate the buffers to be cleared. The three masks are GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT, and GL_STENCIL_BUFFER_BIT.

Usage

```
glClear(GL_COLOR_BUFFER_BIT);
```

## glColor3f

Sets the current color of the pen

Syntax

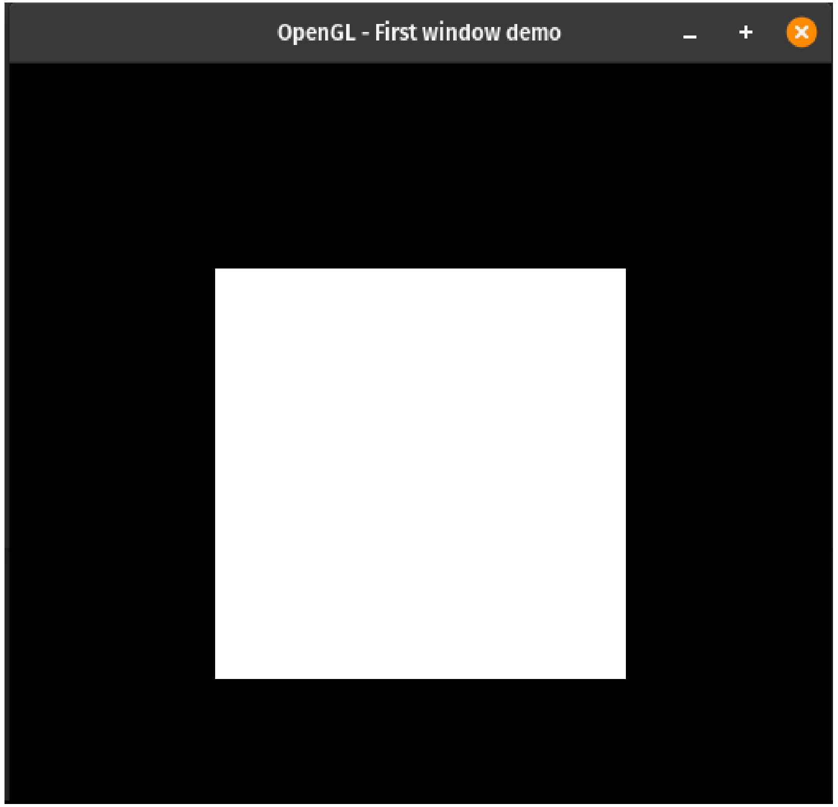
```
void glColor3f(GLfloat red, GLfloat green, GLfloat blue);
```

Usage

```
glColor3f(1.0, 1.0, 1.0);
```

## Example

```
glBegin(GL_POLYGON);
      glVertex2f(-0.5, -0.5);

       glVertex2f(-0.5, 0.5);

       glVertex2f(0.5, 0.5);

       glVertex2f(0.5, -0.5);

glEnd();
```



## Standard primitives

GL_POINTS

GL_LINES

GL_LINE_STRIP

GL_LINE_LOOP

GL_QUAD_STRIP

GL_TRIANGLES

GL_TRIANGLE_STRIP

GL_TRIANGLE_FAN

GL_QUADS

GL_POLYGON

## glFlush

void glFlush(void);

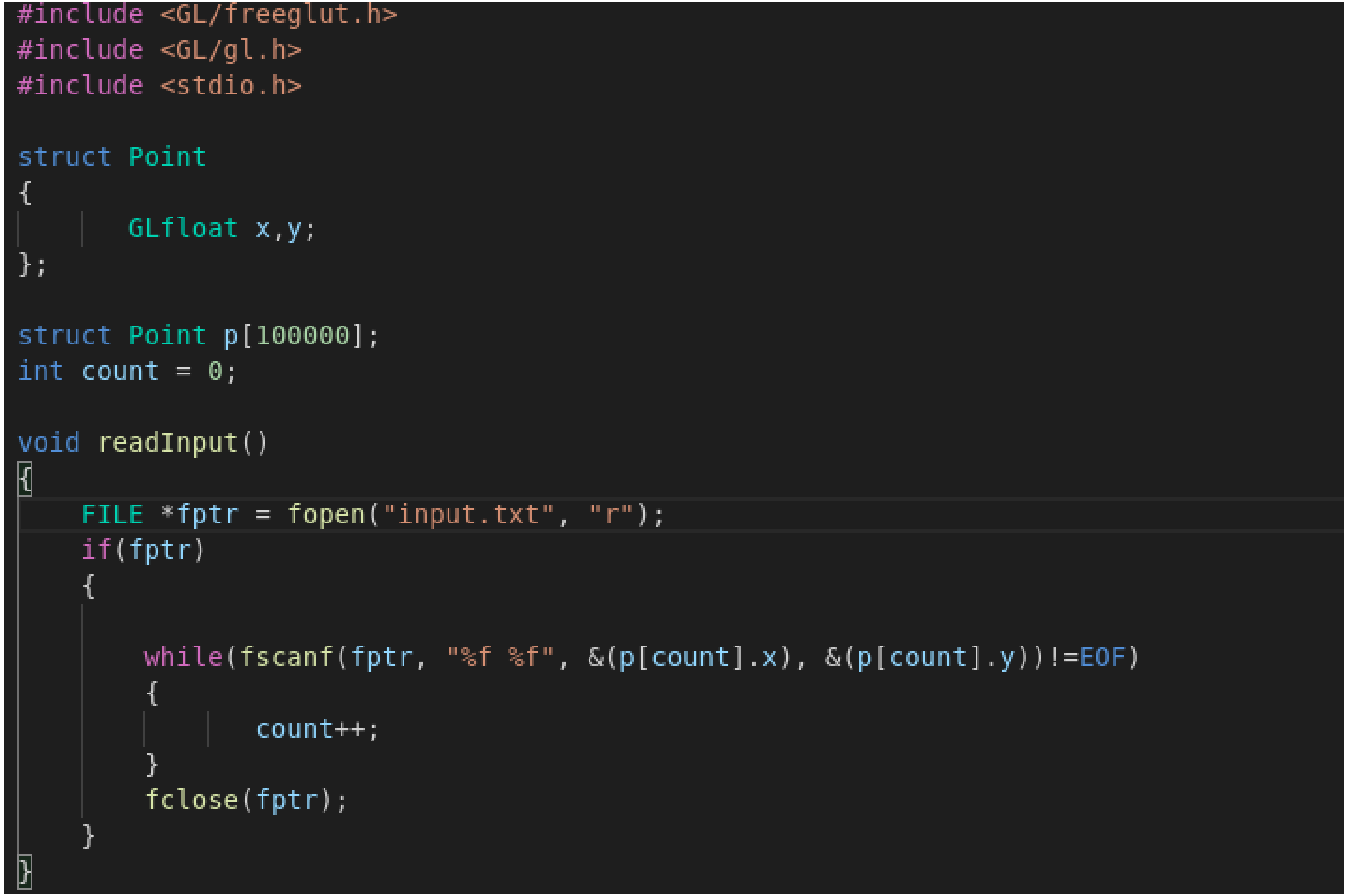
Forces execution of GL commands in finite time.

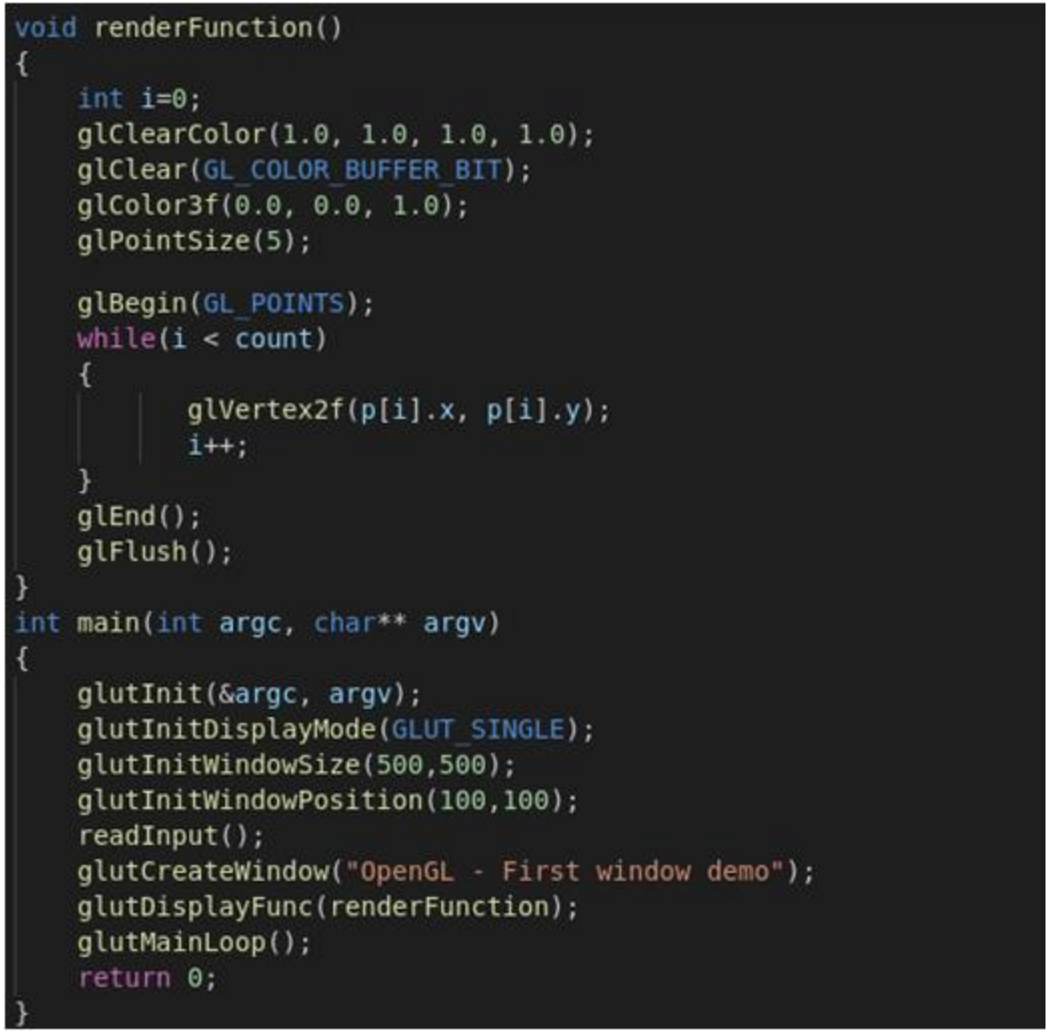
Different GL implementations buffer commands in several different locations, including network buffers and the graphics accelerator itself. glFlush empties all of these buffers, causing all issued commands to be executed as quickly as they are accepted by the actual rendering engine.

Command Suffixes and Argument Data Types

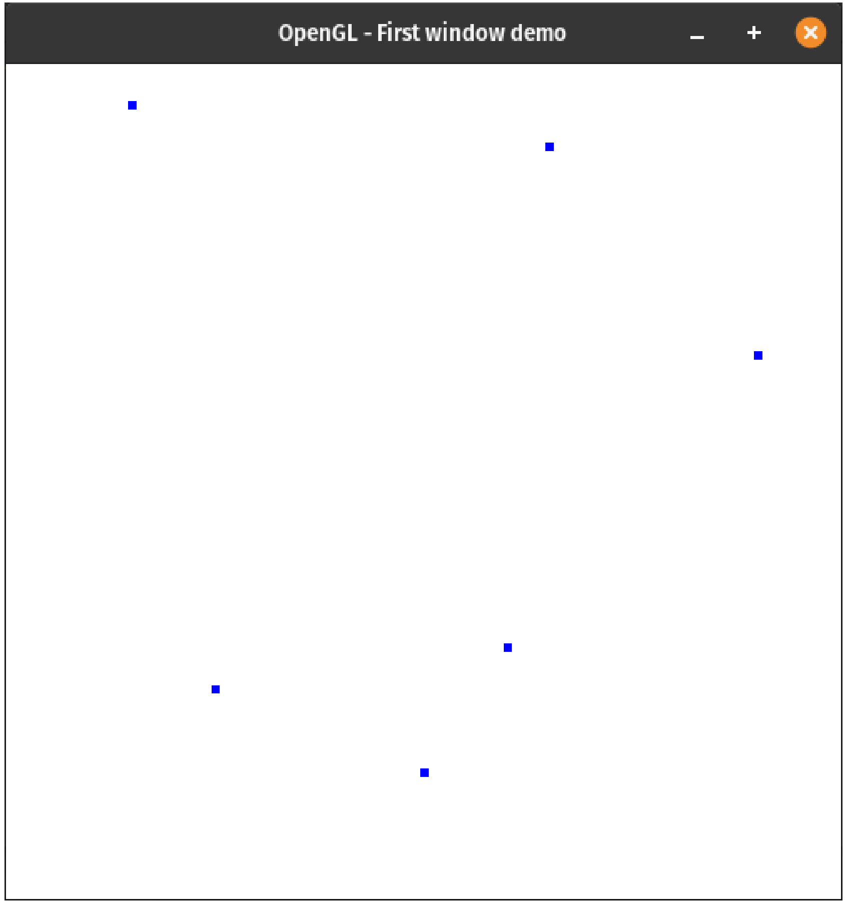
| Suffix | Data Type | Typical Corresponding C-Language Type | OpenGL Type Definition |
|---|---|---|---|
| b | 8-bit integer | signed char | GLbyte |
| s | 16-bit integer | short | GLshort |
| i | 32-bit integer | long | GLint, GLsizei |
| f | 32-bit floating-point | float | GLfloat, GLclampf |
| d | 64-bit floating-point | double | GLdouble, GLclampd |
| ub | 8-bit unsigned integer | unsigned char | GLubyte, GLboolean |
| us | 16-bit unsigned integer | unsigned short | GLushort |
| ui | 32-bit unsigned integer | unsigned long | GLuint, GLenum, GLbitfield |

## Demo 1 - Plotting some points read from a file


```
#include <GL/freeglut.h>
#include <GL/gl.h>
#include <stdio.h>

struct Point
{
        GLfloat x,y;
};

struct Point p[100000];
int count = 0;

void readInput()
{
    FILE *fptr = fopen("input.txt", "r");
    if(fptr)
    {

        while(fscanf(fptr, "%f %f", &(p[count].x), &(p[count].y))!=EOF)
        {
                count++;
        }
        fclose(fptr);
    }
}
```



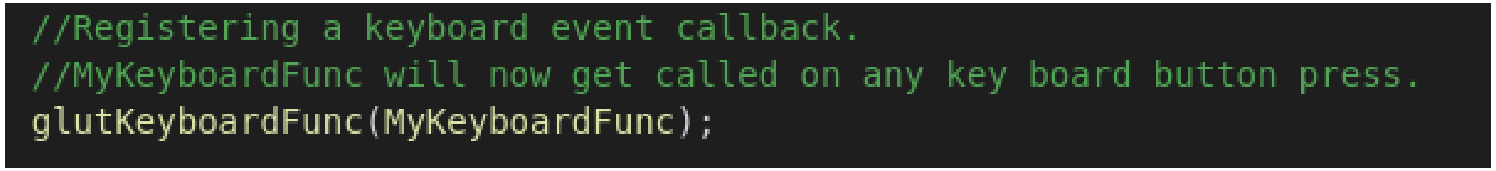
```
void renderFunction()
{
    int i=0;
    glClearColor(1.0, 1.0, 1.0, 1.0);
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0, 0.0, 1.0);
    glPointSize(5);

    glBegin(GL_POINTS);
    while(i < count)
    {
            glVertex2f(p[i].x, p[i].y);
            i++;
    }
    glEnd();
    glFlush();
}
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(100,100);
    readInput();
    glutCreateWindow("OpenGL - First window demo");
    glutDisplayFunc(renderFunction);
    glutMainLoop();
    return 0;
}
```


OpenGL - First window demo

## Demo 2 - Event handling in OpenGL

In the following program we try to clear the screen using a color depending on the key we pressed on the keyboard.

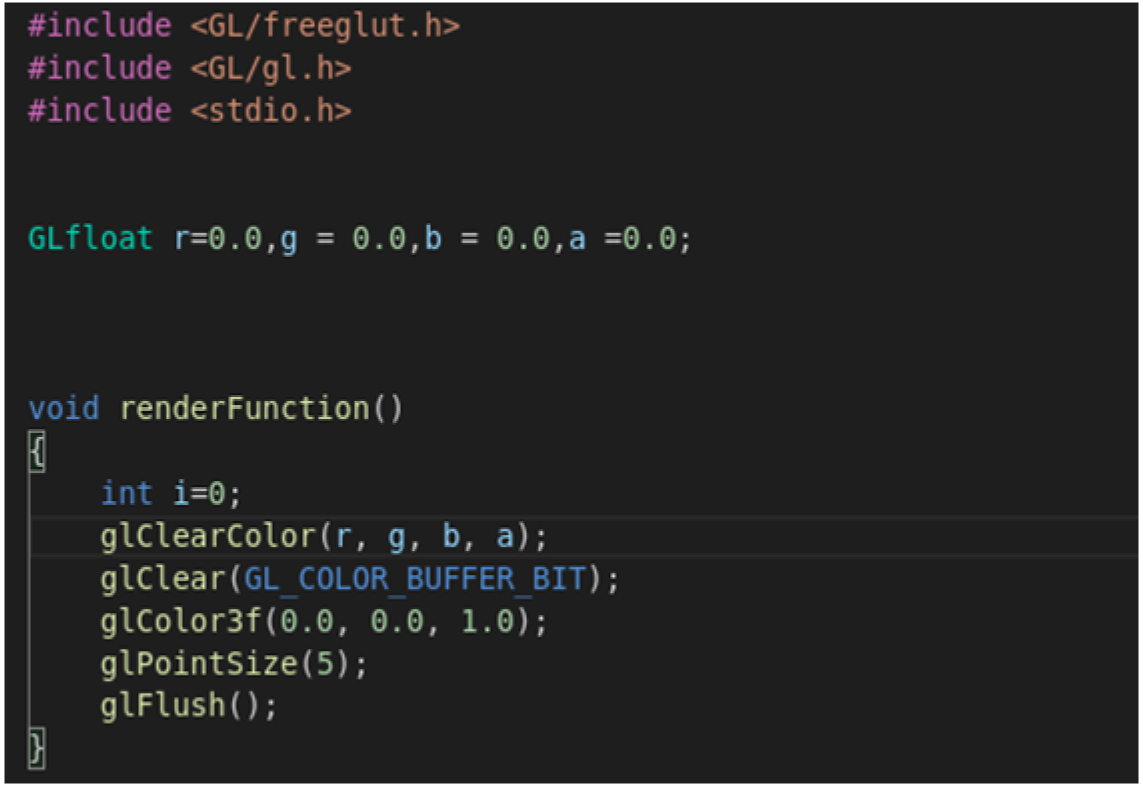
To do this we need to register a callback corresponding to the keypress event in the initialization part.



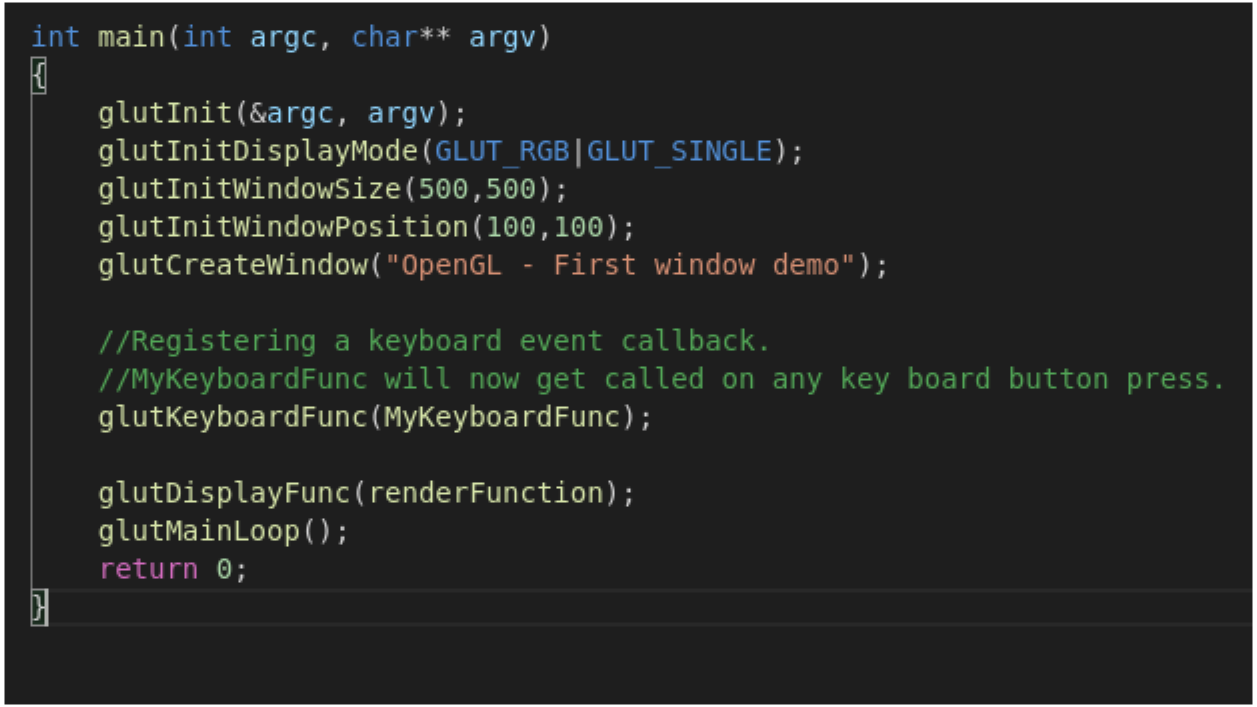
```
//Registering a keyboard event callback.
//MyKeyboardFunc will now get called on any key board button press.
glutKeyboardFunc(MyKeyboardFunc);
```

We also need to define what MyKeyboardFunc does.

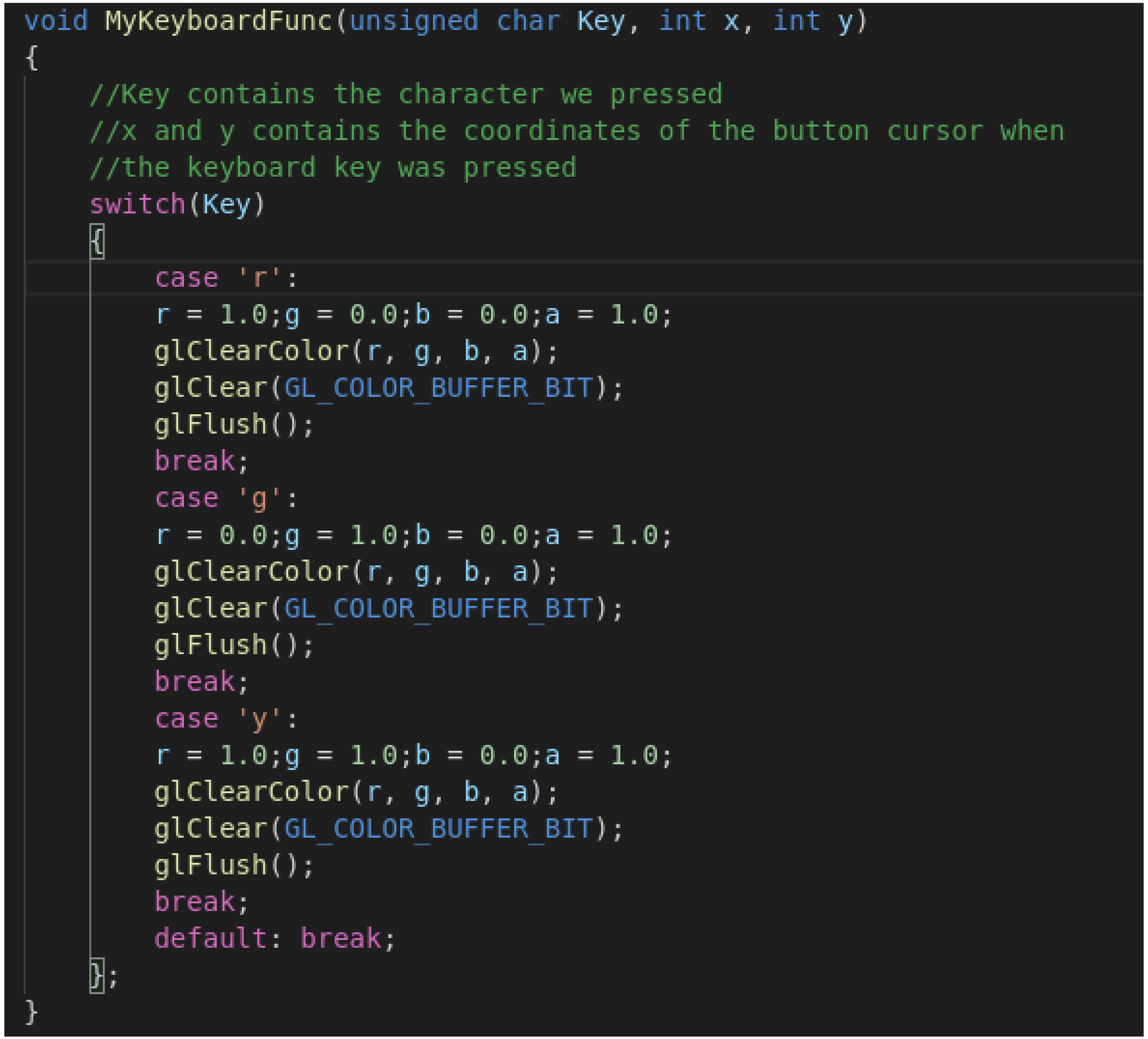
Check out the following slides for the entire program.



```
#include <GL/freeglut.h>
#include <GL/gl.h>
#include <stdio.h>


GLfloat r=0.0,g = 0.0,b = 0.0,a =0.0;



void renderFunction()
{
    int i=0;
    glClearColor(r, g, b, a);
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0, 0.0, 1.0);
    glPointSize(5);
    glFlush();
}
```

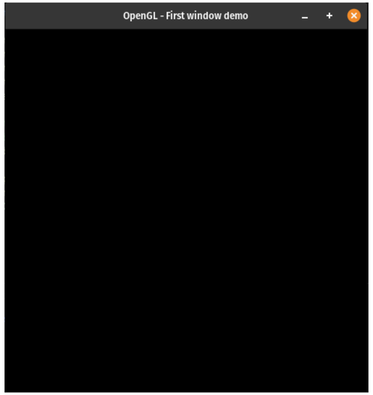


```
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_RGB|GLUT_SINGLE);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(100,100);
    glutCreateWindow("OpenGL - First window demo");

    //Registering a keyboard event callback.
    //MyKeyboardFunc will now get called on any key board button press.
    glutKeyboardFunc(MyKeyboardFunc);

    glutDisplayFunc(renderFunction);
    glutMainLoop();
    return 0;
}
```

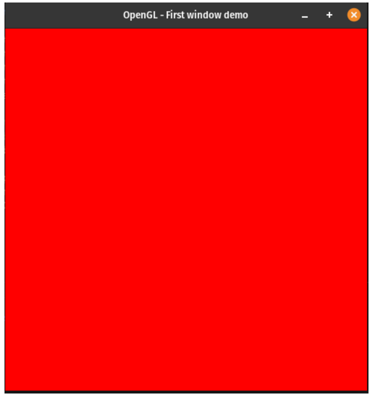


```c
void MyKeyboardFunc(unsigned char Key, int x, int y)
{
    //Key contains the character we pressed
    //x and y contains the coordinates of the button cursor when
    //the keyboard key was pressed
    switch(Key)
    {
        case 'r':
        r = 1.0;g = 0.0;b = 0.0;a = 1.0;
        glClearColor(r, g, b, a);
        glClear(GL_COLOR_BUFFER_BIT);
        glFlush();
        break;
        case 'g':
        r = 0.0;g = 1.0;b = 0.0;a = 1.0;
        glClearColor(r, g, b, a);
        glClear(GL_COLOR_BUFFER_BIT);
        glFlush();
        break;
        case 'y':
        r = 1.0;g = 1.0;b = 0.0;a = 1.0;
        glClearColor(r, g, b, a);
        glClear(GL_COLOR_BUFFER_BIT);
        glFlush();
        break;
        default: break;
    };
}
```

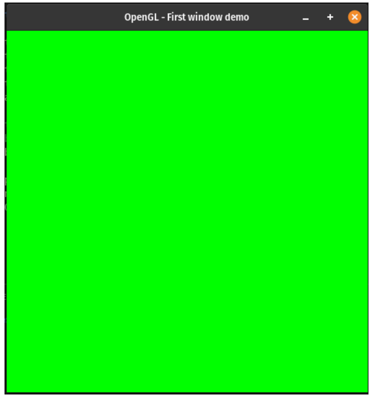
## Output



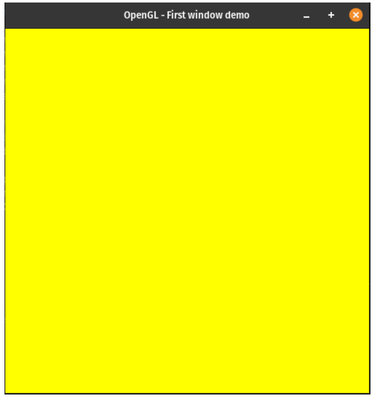
Initially



Pressed r



Pressed g



Pressed y

## Practice Questions

1. Write a program to read RGBA values of a color from 'input.txt' and set the background to this color.

1. Write a program to read the coordinates of the points given in the order in which they should be visited and draw a line strip between them. Make the background white in color.

1. Write a program to read the vertex coordinates of a convex polygon from a file and draw the polygon and all its diagonals. Make the background white in color.

## Practice Questions

4. Write a program to read the vertex coordinates of a rectangular strip from a file and plot the strip in red color on the screen. Make the background black. Remember in OpenGL -z axis goes into the screen, +z axis comes out of the screen.

4. Write a program in which you can make strokes by holding down the mouse left button, dragging and releasing. The path (stroke) traced out by the mouse pointer between the events of mouse's left button click and the eventual release of the left click needs to be plotted. There can be any number of strokes. Using this concept write your name on the screen. Make the background white and the strokes black in color.

## Assignment-1 and Quiz-1

Assignment 1 will be posted soon

Test will be conducted 6th February 2024, Tuesday.

- Topics – the basics of OpenGL, practice questions and assignment 1.
- Marks – 15

## References

[1].*https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwi4ydrc7LHsAhU94HMBHViVDxcQFjADegQIBRAC&url=https%3A%2F%2Fwww.cs.utexas.edu%2Fusers%2Ffussell%2Fcourses%2Fcs354%2Fhandouts%2FAddison.Wesley.OpenGL.Programming.Guide.8th.Edition.Mar.2013.ISBN.0321773039.pdf&usg=AOvVaw0eSL-A754ij_wV_sq03JvS*

[2].*OpenGL Architecture Review Board, OpenGL Reference Manual: The Official Reference Document for OpenGL, Release 1, Addison-Wesley, Reading, Massachusetts, 1992 (ISBN 0-201-63276-4).*

[3].*http://www.cs.toronto.edu/~kyros/courses/418/Notes/Visibility.pdf*

[4].*https://www.youtube.com/watch?v=pQcC2CqReSA https://flylib.com/books/en/2.789.1.32/1/*

[5].*http://www.opengl-tutorial.org/beginners-tutorials/tutorial-1-opening-a-window/*

Thank you

## First OpenGL code

```
#include<GL/freeglut.h>
#include<GL/gl.h>


int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGBA);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(100,100);
    glutCreateWindow("OpenGL - First window demo");
    glutMainLoop();
    return 0;
}
```