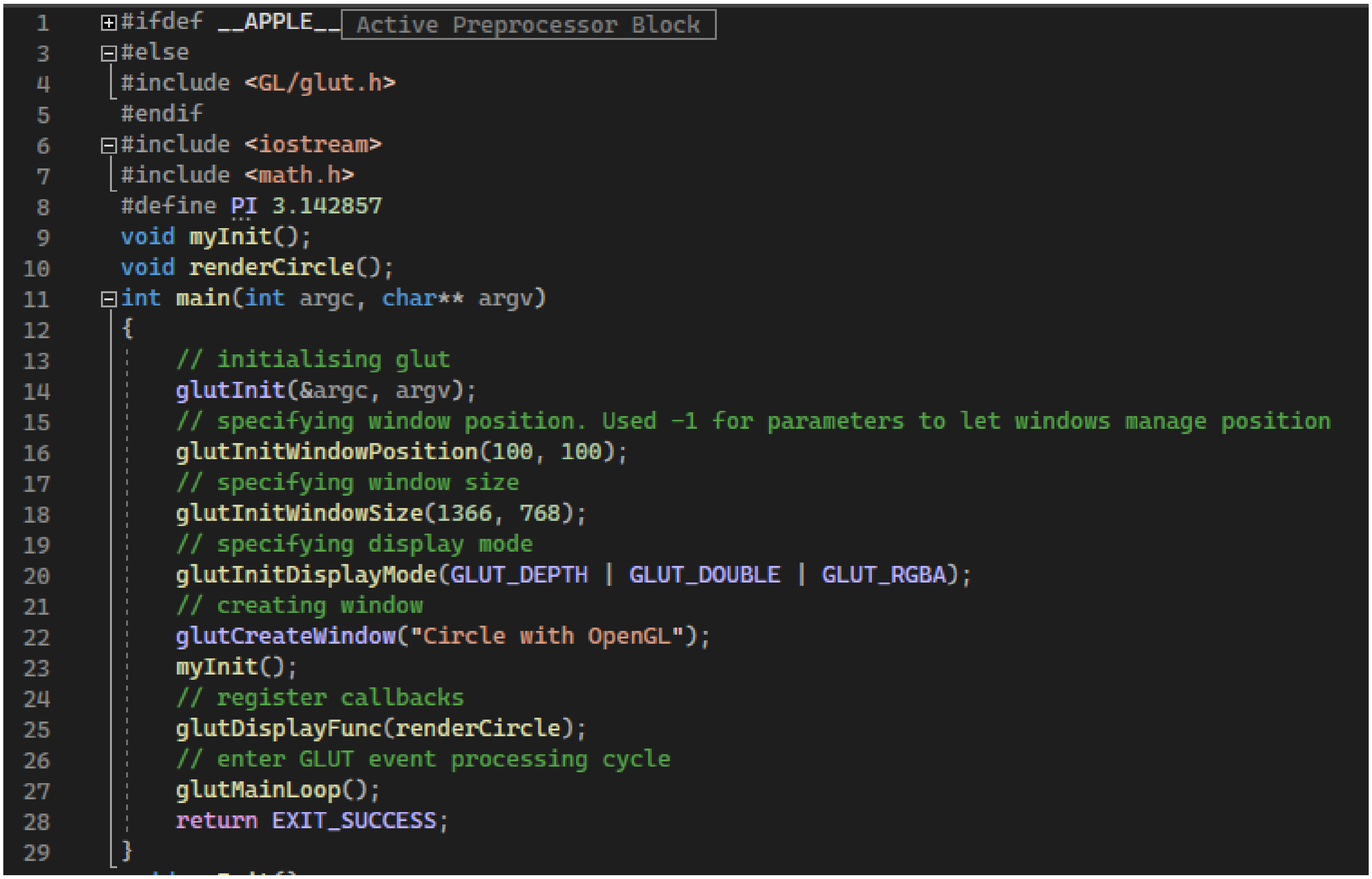
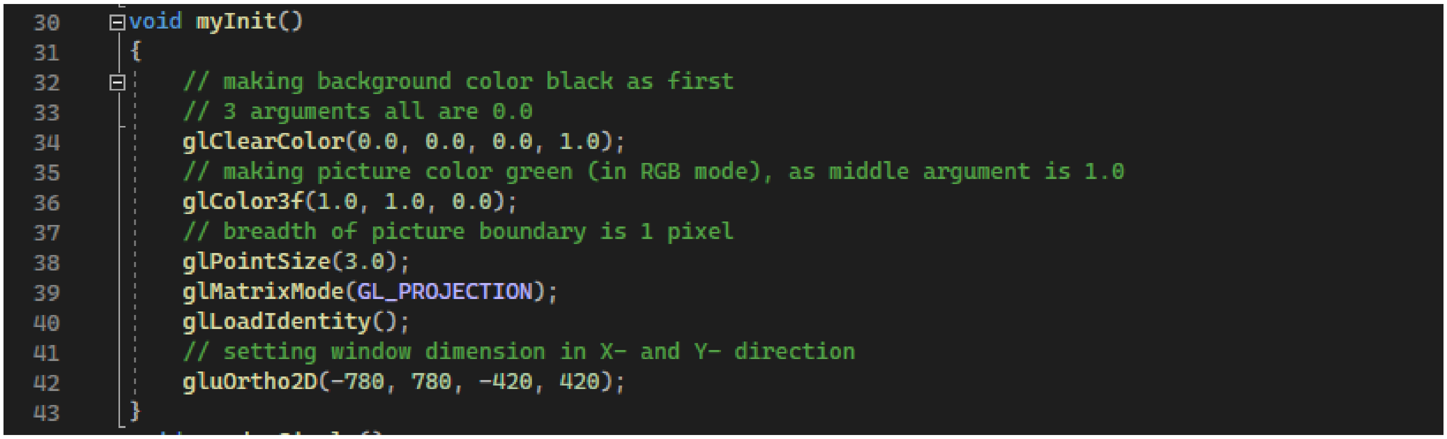
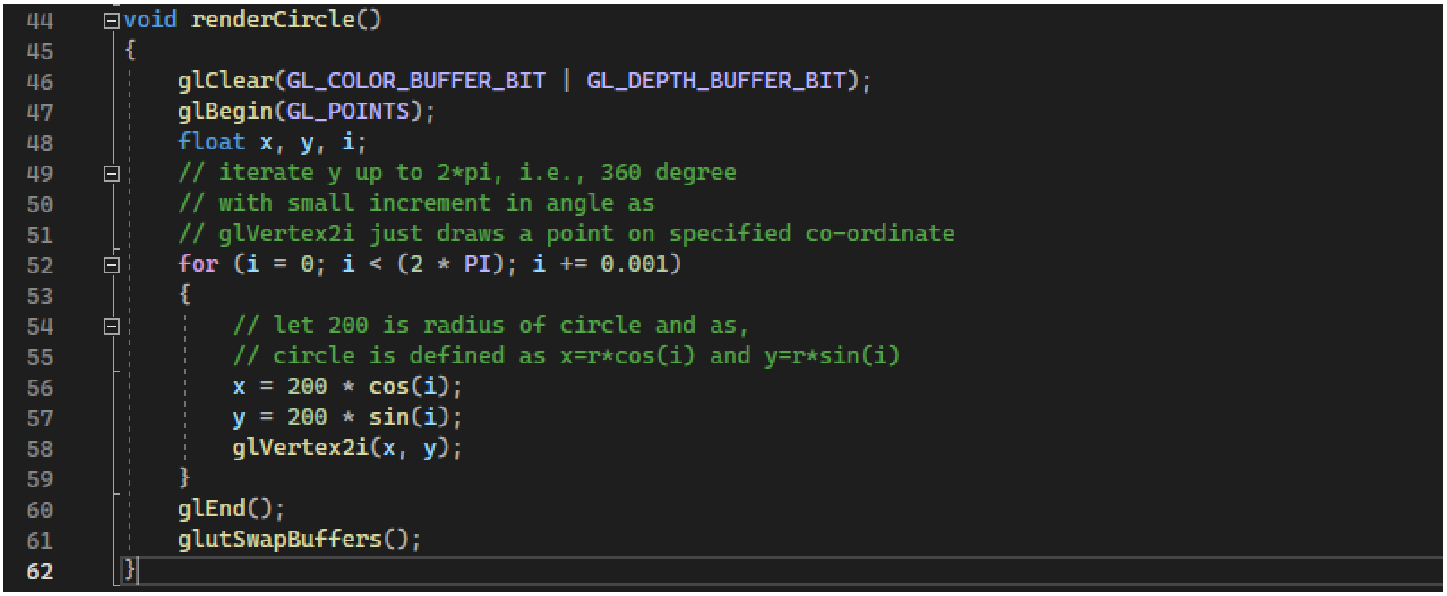
Task: Create a circle using free glut library:



```
#ifdef __APPLE__ Active Preprocessor Block
#else
#include <GL/glut.h>
#endif
#include <iostream>
#include <math.h>
#define PI 3.142857
void myInit();
void renderCircle();
int main(int argc, char** argv)
{
    // initialising glut
    glutInit(&argc, argv);
    // specifying window position. Used -1 for parameters to let windows manage position
    glutInitWindowPosition(100, 100);
    // specifying window size
    glutInitWindowSize(1366, 768);
    // specifying display mode
    glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGBA);
    // creating window
    glutCreateWindow("Circle with OpenGL");
    myInit();
    // register callbacks
    glutDisplayFunc(renderCircle);
    // enter GLUT event processing cycle
    glutMainLoop();
    return EXIT_SUCCESS;
}
```

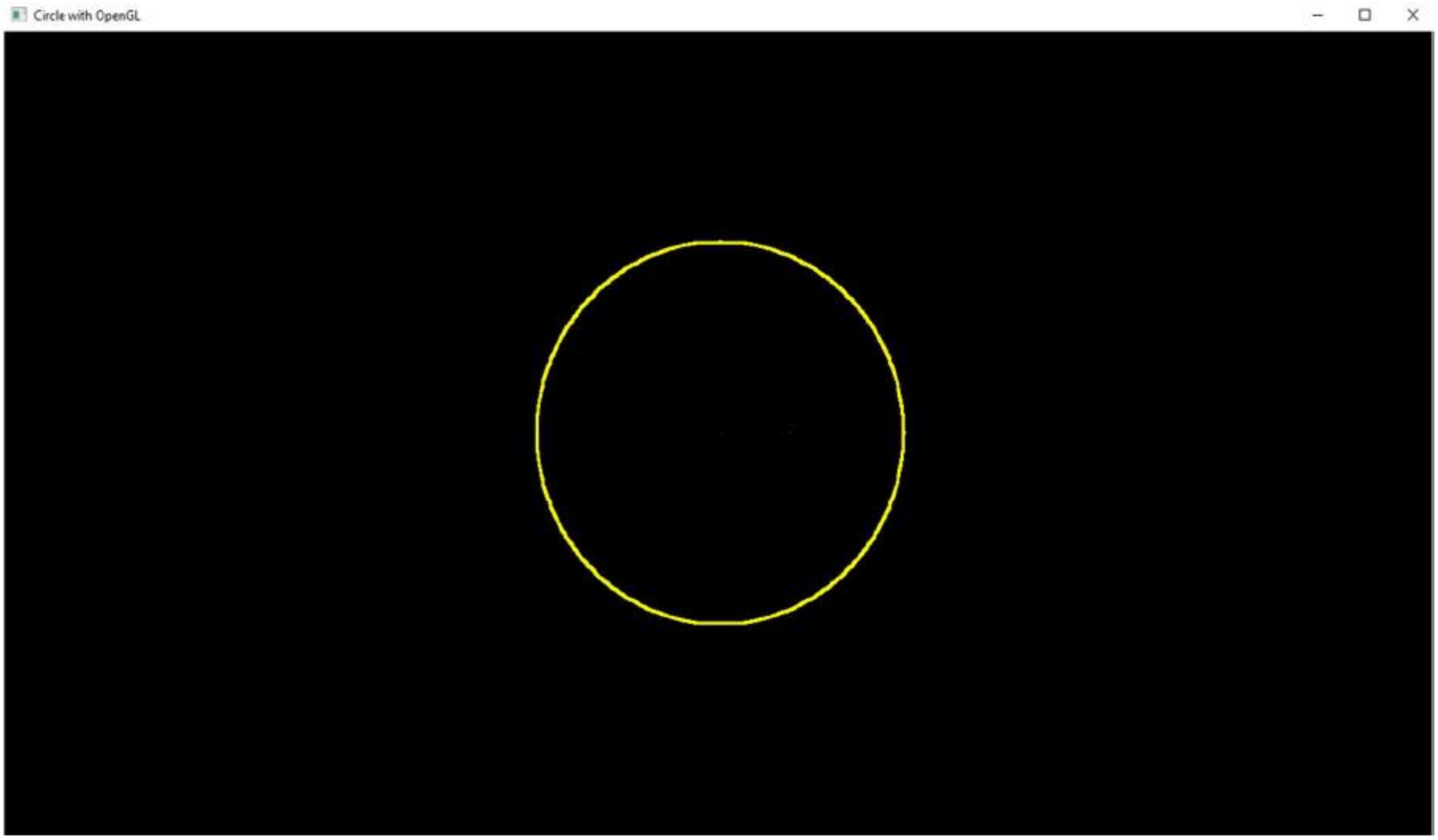


```
void myInit()
{
    // making background color black as first
    // 3 arguments all are 0.0
    glClearColor(0.0, 0.0, 0.0, 1.0);
    // making picture color green (in RGB mode), as middle argument is 1.0
    glColor3f(1.0, 1.0, 0.0);
    // breadth of picture boundary is 1 pixel
    glPointSize(3.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    // setting window dimension in X- and Y- direction
    gluOrtho2D(-780, 780, -420, 420);
}
```



```
void renderCircle()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glBegin(GL_POINTS);
    float x, y, i;
    // iterate y up to 2*pi, i.e., 360 degree
    // with small increment in angle as
    // glVertex2i just draws a point on specified co-ordinate
    for (i = 0; i < (2 * PI); i += 0.001)
    {
        // let 200 is radius of circle and as,
        // circle is defined as x=r*cos(i) and y=r*sin(i)
        x = 200 * cos(i);
        y = 200 * sin(i);
        glVertex2i(x, y);
    }
    glEnd();
    glutSwapBuffers();
}
```

Output:


Circle with OpenGL