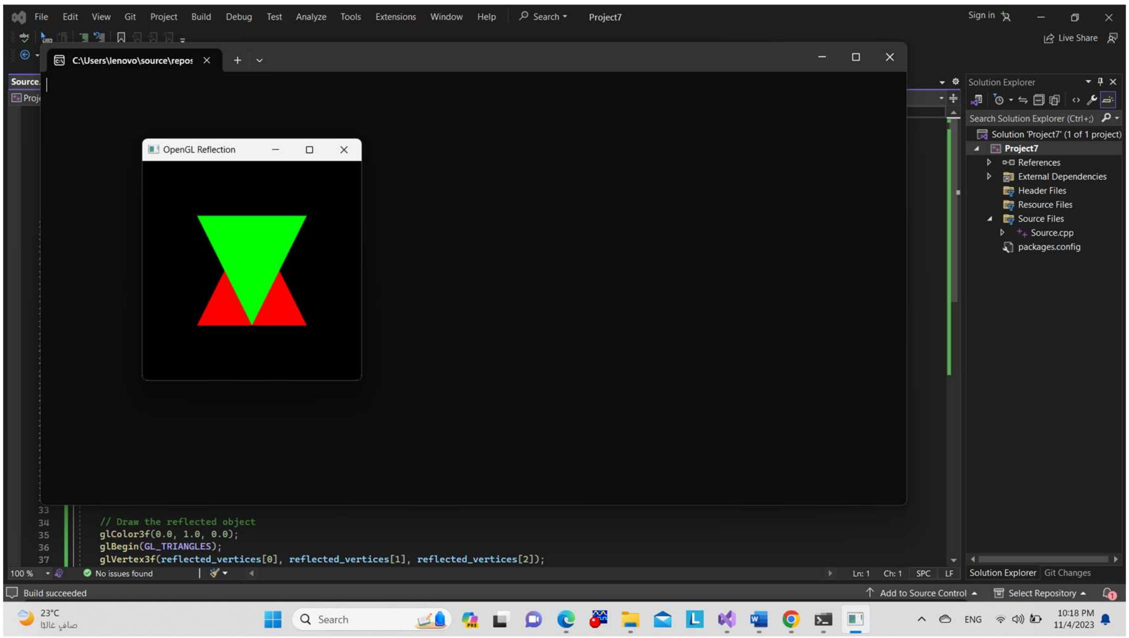
```
#include <windows.h>  // for MS Windows
#include <GL/glut.h>  // GLUT, include glu.h and gl.h


void display() {
    glClear(GL_COLOR_BUFFER_BIT);

    // Define the vertices of the object you want to reflect
    GLfloat vertices[] = { -0.5, -0.5, 0.0, 0.5, -0.5, 0.0, 0.0, 0.5, 0.0 };

    // Define the reflection matrix for the line y = 0
    GLfloat reflection_matrix[] = { 1, 0, 0, 0, -1, 0, 0, 0, 1 };

    // Multiply the reflection matrix by the vertices of the object to obtain
the reflected vertices
    GLfloat reflected_vertices[9];
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            reflected_vertices[i * 3 + j] = 0;
            for (int k = 0; k < 3; k++) {
                reflected_vertices[i * 3 + j] += vertices[i * 3 + k] *
reflection_matrix[k * 3 + j];
            }
        }
    }

    // Draw the original object
    glColor3f(1.0, 0.0, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex3f(vertices[0], vertices[1], vertices[2]);
    glVertex3f(vertices[3], vertices[4], vertices[5]);
    glVertex3f(vertices[6], vertices[7], vertices[8]);
    glEnd();

    // Draw the reflected object
    glColor3f(0.0, 1.0, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex3f(reflected_vertices[0], reflected_vertices[1],
reflected_vertices[2]);
    glVertex3f(reflected_vertices[3], reflected_vertices[4],
reflected_vertices[5]);
    glVertex3f(reflected_vertices[6], reflected_vertices[7],
reflected_vertices[8]);
    glEnd();

    glFlush();
}

int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutCreateWindow("OpenGL Reflection");
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```



```
    // Draw the reflected object
    glColor3f(0.0, 1.0, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex3f(reflected_vertices[0], reflected_vertices[1], reflected_vertices[2]);
```

No issues found

Build succeeded