

**Computer Graphics Lab**

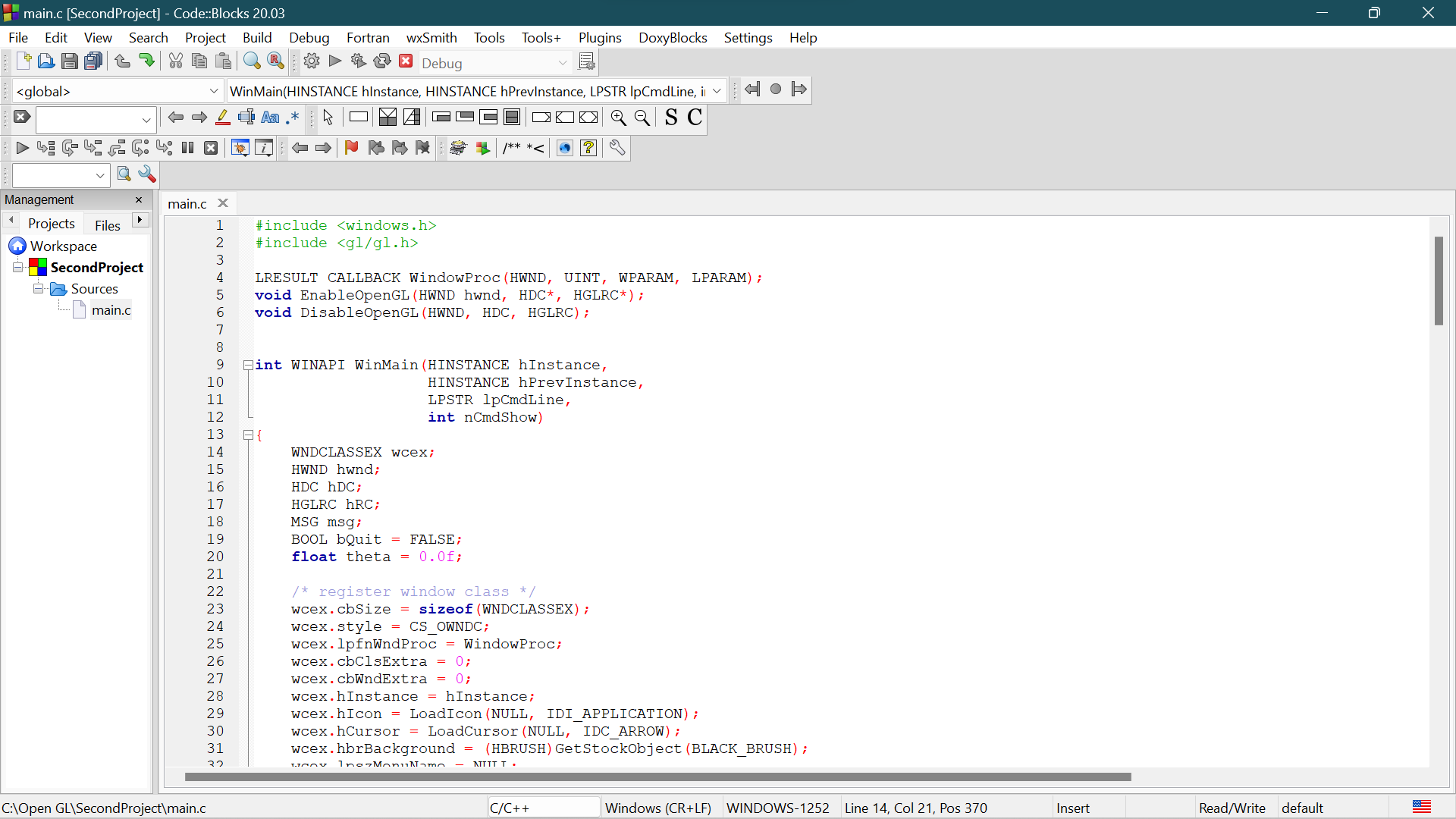
**Name:** Pratham Kandari

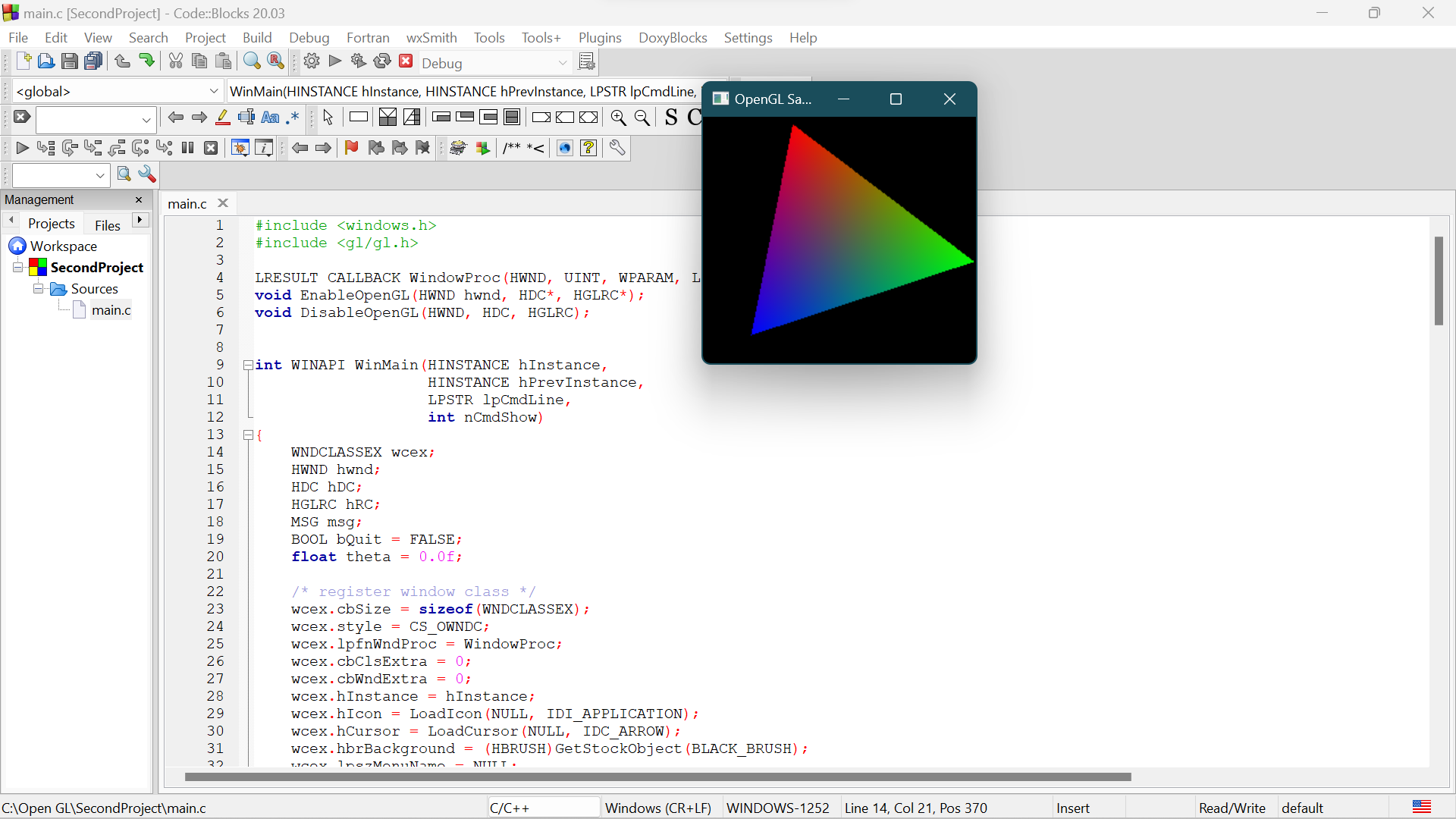
**Sap Id:** 500097663

**Batch:** 7

**Experiment -2**

1. Configuration screen of OpenGL.





1. WAP to display the window RED, GREEN, and BLUE.

Ans: Program for RED

#include<GL/freeglut.h>

void init(){

glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //Line C

glutInitWindowSize(640,480);

glutInitWindowPosition(1000,200);

glutCreateWindow("Simple Window");

}

void display()

{

glClearColor(1.0,0.0,0.0,0.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

// gluOrtho2D(0.0,100.0,0,100.0);

glFlush();

}

int main(int argc,char \*\*argv)

{

glutInit(&argc,argv); //Line A

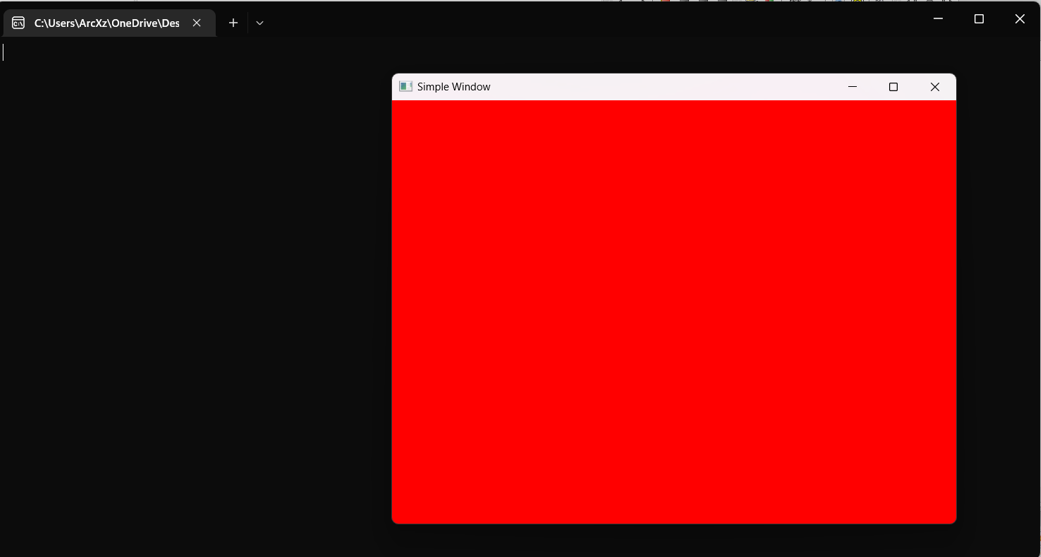
init(); //Line B

glutDisplayFunc(display);

glutMainLoop();

return 0;

}



Program for GREEN

#include<GL/freeglut.h>

void init(){

glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //Line C

glutInitWindowSize(640,480);

glutInitWindowPosition(1000,200);

glutCreateWindow("Simple Window");

}

void display()

{

glClearColor(0.0,1.0,0.0,0.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

// gluOrtho2D(0.0,100.0,0,100.0);

glFlush();

}

int main(int argc,char \*\*argv)

{

glutInit(&argc,argv); //Line A

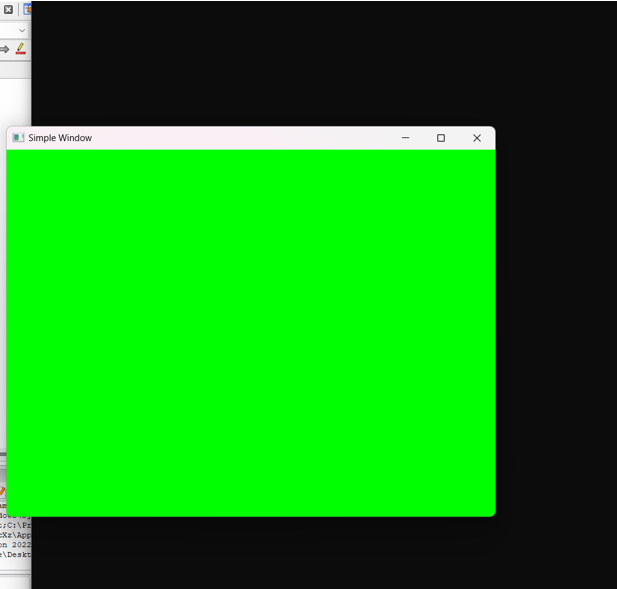
init(); //Line B

glutDisplayFunc(display);

glutMainLoop();

return 0;

}



Program for BLUE

#include<GL/freeglut.h>

void init(){

glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //Line C

glutInitWindowSize(640,480);

glutInitWindowPosition(1000,200);

glutCreateWindow("Simple Window");

}

void display()

{

glClearColor(0.0,0.0,1.0,0.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

// gluOrtho2D(0.0,100.0,0,100.0);

glFlush();

}

int main(int argc,char \*\*argv)

{

glutInit(&argc,argv); //Line A

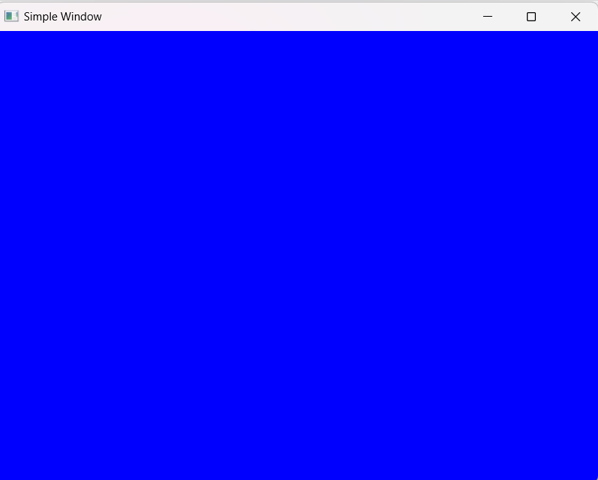
init(); //Line B

glutDisplayFunc(display);

glutMainLoop();

return 0;

}



1. WAP to display the Polygon.

Ans:

#include <GL/gl.h>

#include <GL/glut.h> //

void display() {

glClearColor( 0, 0, 0, 1 );

glClear( GL\_COLOR\_BUFFER\_BIT );

glBegin(GL\_POLYGON);

glColor3f( 0.0f, 0.0f, 1.0f );

glVertex2f(-0.3f, 0.3f);

glVertex2f(0.3f, 0.3f);

glVertex2f(0.6f, 0.0f);

glVertex2f(0.3f, -0.3f);

glVertex2f(-0.3f, -0.3f);

glVertex2f(0.6f, 0.0f);

glEnd();

glutSwapBuffers();

}

int main( int argc, char\*\* argv )

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE);

glutInitWindowSize(500,500);

glutInitWindowPosition(100,100);

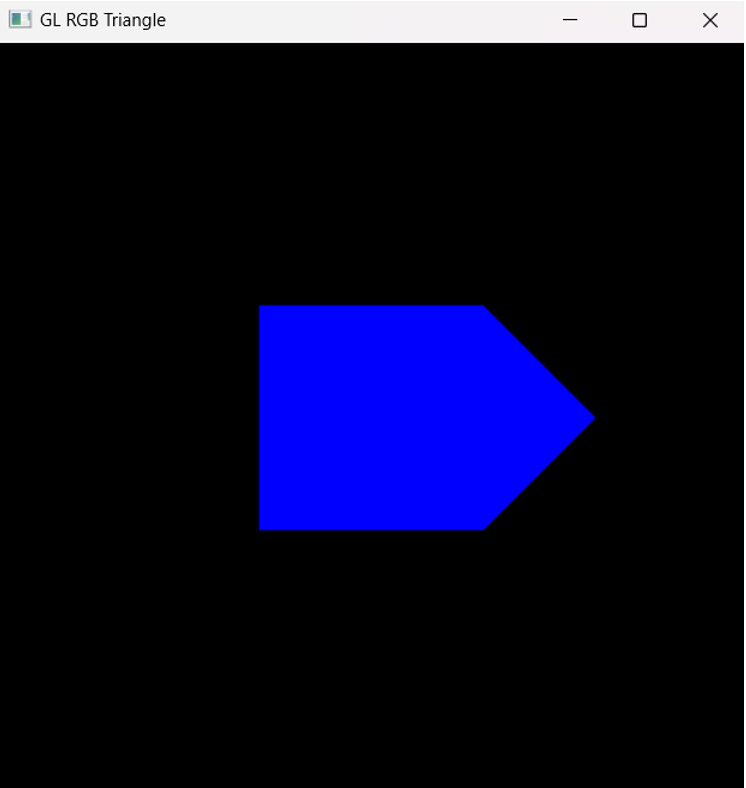
glutCreateWindow("GL RGB Triangle");

glutDisplayFunc(display);

glutMainLoop();

return 0;

}



1. WAP to display triangle.

Ans:

#include <GL/gl.h>

#include <GL/glut.h> //

void display() {

glClearColor( 0, 0, 0, 1 );

glClear( GL\_COLOR\_BUFFER\_BIT );

glBegin(GL\_TRIANGLES);

glColor3f( 1, 0, 0 );

glVertex2f( -0.8, -0.8 );

glColor3f( 0, 1, 0 );

glVertex2f( 0.8, -0.8 );

glColor3f( 0, 0, 1 );

glVertex2f( 0, 0.9 );

glEnd();

glutSwapBuffers();

}

int main( int argc, char\*\* argv ) {

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE);

glutInitWindowSize(500,500);

glutInitWindowPosition(100,100);

glutCreateWindow("GL RGB Triangle");

glutDisplayFunc(display);

glutMainLoop();

return 0;

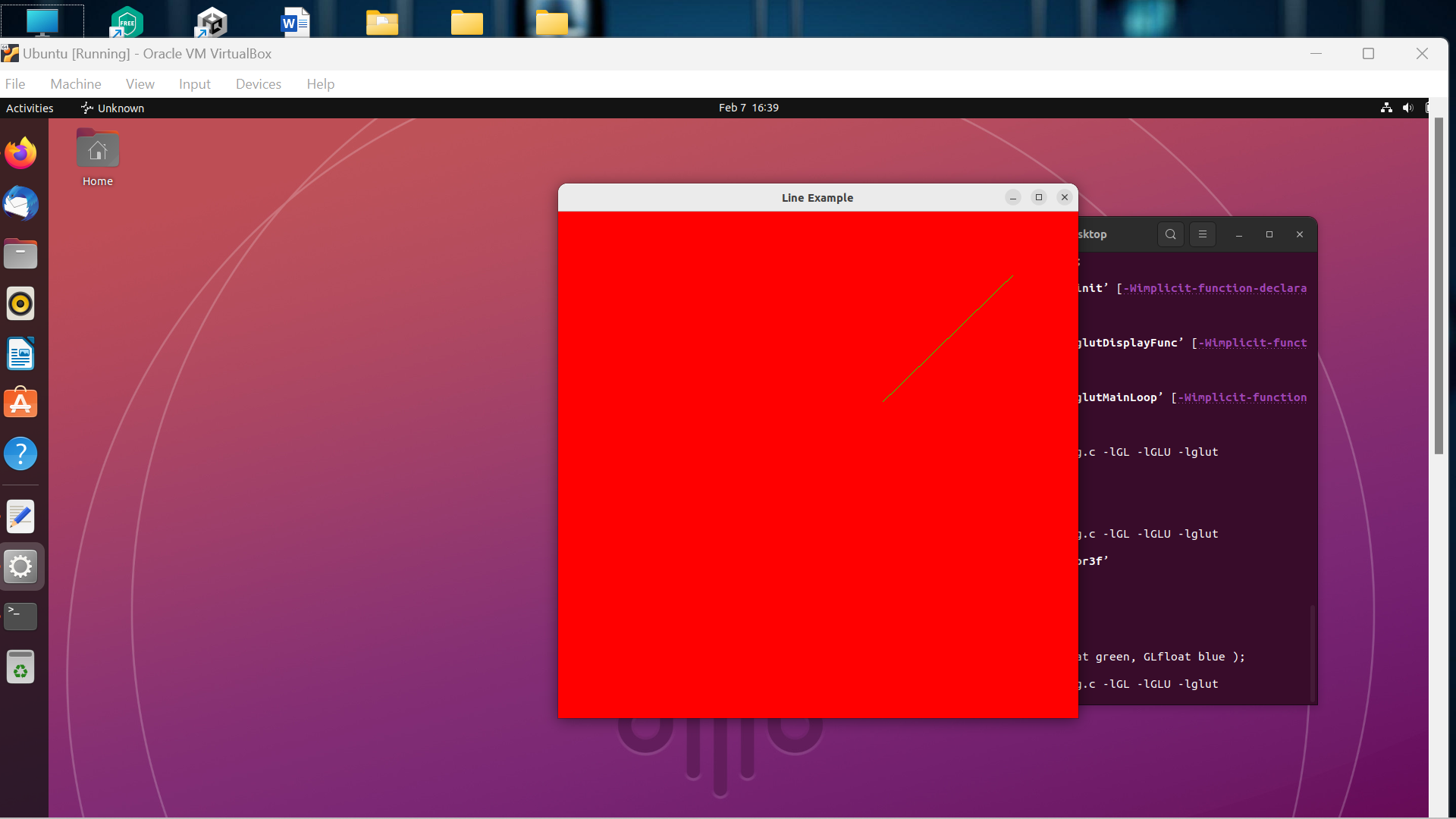
}

****

1. WAP to present the line segment.

Ans:

#include <GL/glut.h>  
#include <stdio.h>  
  
void display() {  
  glClear(GL\_COLOR\_BUFFER\_BIT);  
  glClearColor(1.0, 0.0, 0.0, 0.0);  
  glColor3f(0.0, 1.0, 0.0);  
  glBegin(GL\_LINES);  
    glVertex2f(0.25, 0.25);  
    glVertex2f(0.75, 0.75);  
  glEnd();  
  glFlush();  
}  
  
int main(int argc, char\*\* argv) {  
  glutInit(&argc, argv);  
  glutCreateWindow("Line Example");  
  glutDisplayFunc(display);  
  glutMainLoop();  
  return 0;  
}



1. The sample program, GLUT CallBack Function, Register CallBack Function, Event Queue, OPENGL Common Format.

Ans:

/\*

\* GL01Hello.cpp: Test OpenGL/GLUT C/C++ Setup

\* Tested under Eclipse CDT with MinGW/Cygwin and CodeBlocks with MinGW

\* To compile with -lfreeglut -lglu32 -lopengl32

\*/

#include <windows.h> // for MS Windows

#include <GL/glut.h> // GLUT, include glu.h and gl.h

/\* Handler for window-repaint event. Call back when the window first appears and

whenever the window needs to be re-painted. \*/

void display() {

glClearColor(0.0f, 0.0f, 0.0f, 1.0f); // Set background color to black and opaque

glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)

// Draw a Red 1x1 Square centered at origin

glBegin(GL\_QUADS); // Each set of 4 vertices form a quad

glColor3f(1.0f, 0.0f, 0.0f); // Red

glVertex2f(-0.5f, -0.5f); // x, y

glVertex2f( 0.5f, -0.5f);

glVertex2f( 0.5f, 0.5f);

glVertex2f(-0.5f, 0.5f);

glEnd();

glFlush(); // Render now

}

/\* Main function: GLUT runs as a console application starting at main() \*/

int main(int argc, char\*\* argv) {

glutInit(&argc, argv); // Initialize GLUT

glutCreateWindow("OpenGL Setup Test"); // Create a window with the given title

glutInitWindowSize(320, 320); // Set the window's initial width & height

glutInitWindowPosition(50, 50); // Position the window's initial top-left corner

glutDisplayFunc(display); // Register display callback handler for window re-paint

glutMainLoop(); // Enter the event-processing loop

return 0;

}