Lab Report

Lab Report No: 01

Course Code: CSE422

Course Title: Computer Graphics Lab

Lab Title: Basic Shape in OpenGL

Submitted To

Mr Tanim Ahmed
Lecturer
Department of CSE
Daffodil International University

Submitted By

Fazley Atif Maruf
ID: 191-15-2349
Section: Pc-C
Department of CSE, DIU

Date of Submission: 19-06-2022

Lab Name: Basic Shape in OpenGL

Lab Outcome: In this work I covered three basic shape these are gradually Triangle, Rectangle, Quadrilateral.

Functionalities: For drawing three basic we need some function like as GL_TRIANGLE, GL_QUADS.

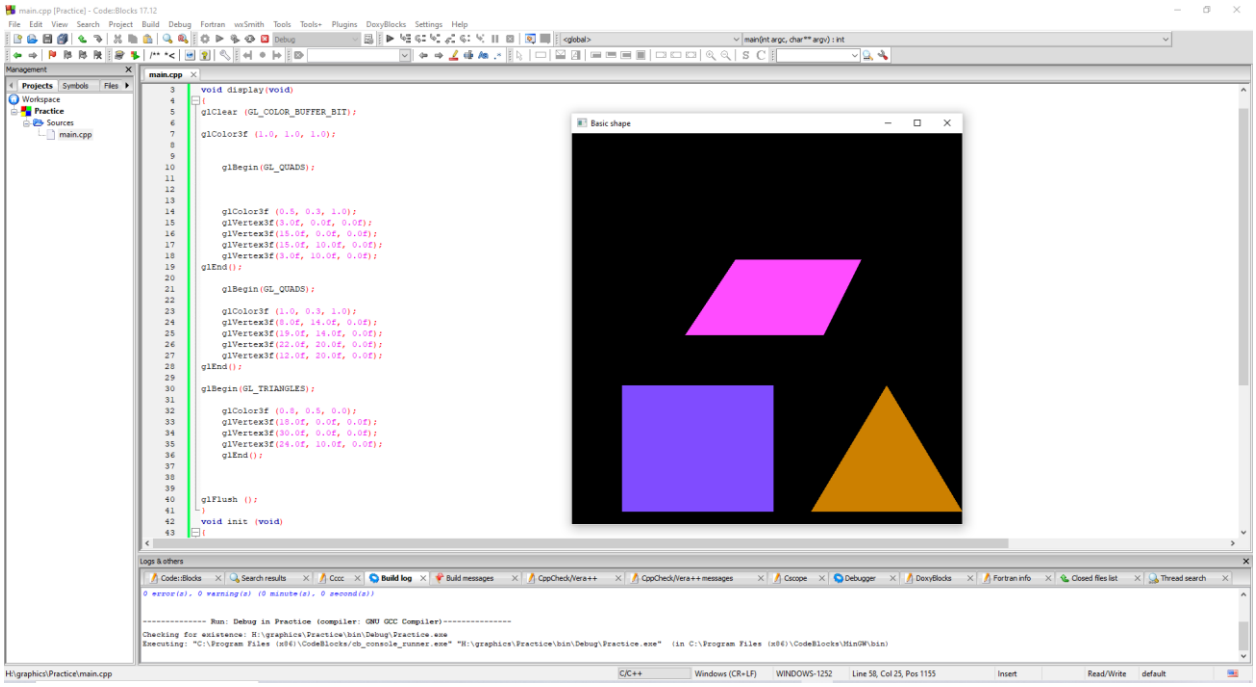
Source Code:

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

#include <GL/glut.h>

void display(void)

{

glClear (GL_COLOR_BUFFER_BIT);

glColor3f (1.0, 1.0, 1.0);

        glBegin(GL_QUADS);

        glColor3f (0.5, 0.3, 1.0);

        glVertex3f(3.0f, 0.0f, 0.0f);

        glVertex3f(15.0f, 0.0f, 0.0f);

        glVertex3f(15.0f, 10.0f, 0.0f);

   glVertex3f(3.0f, 10.0f, 0.0f);

glEnd();

        glBegin(GL_QUADS);

        glColor3f (1.0, 0.3, 1.0);

        glVertex3f(8.0f, 14.0f, 0.0f);

        glVertex3f(19.0f, 14.0f, 0.0f);

        glVertex3f(22.0f, 20.0f, 0.0f);
```

```
    glVertex3f(12.0f, 20.0f, 0.0f);
glEnd();

glBegin(GL_TRIANGLES);

    glColor3f (0.8, 0.5, 0.0);
        glVertex3f(18.0f, 0.0f, 0.0f);
        glVertex3f(30.0f, 0.0f, 0.0f);
        glVertex3f(24.0f, 10.0f, 0.0f);
        glEnd();
glFlush ();
}
void init (void)
{
/* select clearing (background) color */
glClearColor (0.0, 0.0, 0.0, 0.0);
/* initialize viewing values */
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glOrtho(-1.0, 30, -1.0, 30, -10.0, 10.0);
}

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

```
glutInit(&argc, argv);
glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize (600, 600);
glutInitWindowPosition (100, 100);
glutCreateWindow ("House");
init ();
glutDisplayFunc(display);
glutMainLoop();
return 0;
}
```

Output:



----END----

Lab Report

Lab Report No: 02

Course Code: CSE422

Course Title: Computer Graphics Lab

Lab Title: House Design using 2D Transformation in OpenGL

Submitted To

Mr Tanim Ahmed
Lecturer
Department of CSE
Daffodil International University

Submitted By

Fazley Atif Maruf
ID: 191-15-2349
Section: Pc-C
Department of CSE, DIU

Date of Submission: 19-06-2022

Lab Name: House Design using 2D Transformation in OpenGL

Lab Outcome: In this work I covered three basic shape these are gradually Triangle, Quadrilateral.

Functionalities: For drawing three basic we need some function like as GL_TRIANGLE, GL_QUADS.

Source Code:

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

#include <GL/glut.h>

void display(void)

{

glClear (GL_COLOR_BUFFER_BIT);


glColor3f (1.0, 1.0, 1.0);



        glBegin(GL_QUADS); //Begin quadrilateral coordinates


        //Trapezoid


        glColor3f (0.5, 0.3, 1.0);

        glVertex3f(0.0f, 0.0f, 0.0f);

        glVertex3f(15.0f, 0.0f, 0.0f);

        glVertex3f(15.0f, 10.0f, 0.0f);
```

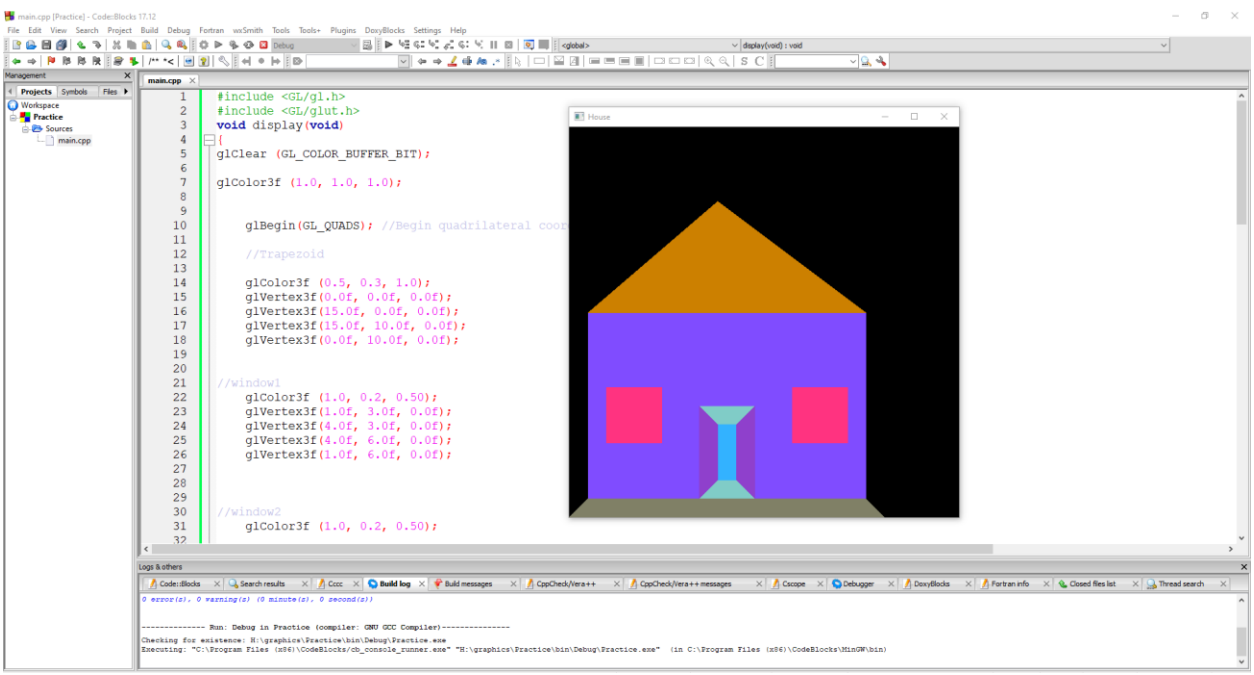
```
    glVertex3f(0.0f, 10.0f, 0.0f);

  //window1
    glColor3f (1.0, 0.2, 0.50);
    glVertex3f(1.0f, 3.0f, 0.0f);
          glVertex3f(4.0f, 3.0f, 0.0f);
          glVertex3f(4.0f, 6.0f, 0.0f);
    glVertex3f(1.0f, 6.0f, 0.0f);

  //window2
    glColor3f (1.0, 0.2, 0.50);

    glVertex3f(11.0f, 3.0f, 0.0f);
          glVertex3f(14.0f, 3.0f, 0.0f);
          glVertex3f(14.0f, 6.0f, 0.0f);
    glVertex3f(11.0f, 6.0f, 0.0f);

  //Door
    glColor3f (0.56, 0.25, 0.8);
    glVertex3f(6.0f, 0.0f, 0.0f);
          glVertex3f(9.0f, 0.0f, 0.0f);
          glVertex3f(9.0f, 5.0f, 0.0f);
    glVertex3f(6.0f, 5.0f, 0.0f);
```

```
glColor3f (0.20, 0.7, 1.0);
glVertex3f(7.0f, 1.0f, 0.0f);
    glVertex3f(8.0f, 1.0f, 0.0f);
    glVertex3f(8.0f, 4.0f, 0.0f);
glVertex3f(7.0f, 4.0f, 0.0f);

glColor3f (0.5, 0.8, 0.78);
glVertex3f(6.0f, 0.0f, 0.0f);
    glVertex3f(9.0f, 0.0f, 0.0f);
    glVertex3f(8.0f, 1.0f, 0.0f);
glVertex3f(7.0f, 1.0f, 0.0f);

glColor3f (0.5, 0.8, 0.78);
glVertex3f(6.0f, 5.0f, 0.0f);
    glVertex3f(7.0f, 4.0f, 0.0f);
    glVertex3f(8.0f, 4.0f, 0.0f);
glVertex3f(9.0f, 5.0f, 0.0f);

glColor3f (0.5, 0.5, 0.4);
glVertex3f(0.0f, 0.0f, 0.0f);
    glVertex3f(-1.0f, -1.0f, 0.0f);
    glVertex3f(16.0f, -1.0f, 0.0f);
glVertex3f(15.0f, 0.0f, 0.0f);
```

```
        glEnd(); //End quadrilateral coordinates

    glBegin(GL_TRIANGLES); //Begin triangle coordinates

    glColor3f (0.8, 0.5, 0.0);
        glVertex3f(0.0f, 10.0f, 0.0f);
        glVertex3f(15.0f, 10.0f, 0.0f);
        glVertex3f(7.0f, 16.0f, 0.0f);
        glEnd();

glFlush ();
}
void init (void)
{
/* select clearing (background) color */
glClearColor (0.0, 0.0, 0.0, 0.0);
/* initialize viewing values */
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glOrtho(-1.0, 20, -1.0, 20, -10.0, 10.0);
}
int main(int argc, char** argv)
{
```

```
glutInit(&argc, argv);
glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize (600, 600);
glutInitWindowPosition (100, 100);
glutCreateWindow ("House");
init ();
glutDisplayFunc(display);
glutMainLoop();
return 0;
}
```

Output:



----END----

Lab Report

Lab Report No: 05

Course Code: CSE422

Course Title: Computer Graphics Lab

Lab Title: Mid-Point Circle Drawing in OpenGL

Submitted To

Mr Tanim Ahmed
Lecturer
Department of CSE
Daffodil International University

Submitted By

Fazley Atif Maruf
ID: 191-15-2349
Section: Pc-C
Department of CSE, DIU

Date of Submission: 19-06-2022

Lab Name: Circle Drawing in OpenGL

Lab Outcome: In this work I covered how to draw a circle based on user input data.

Source Code:

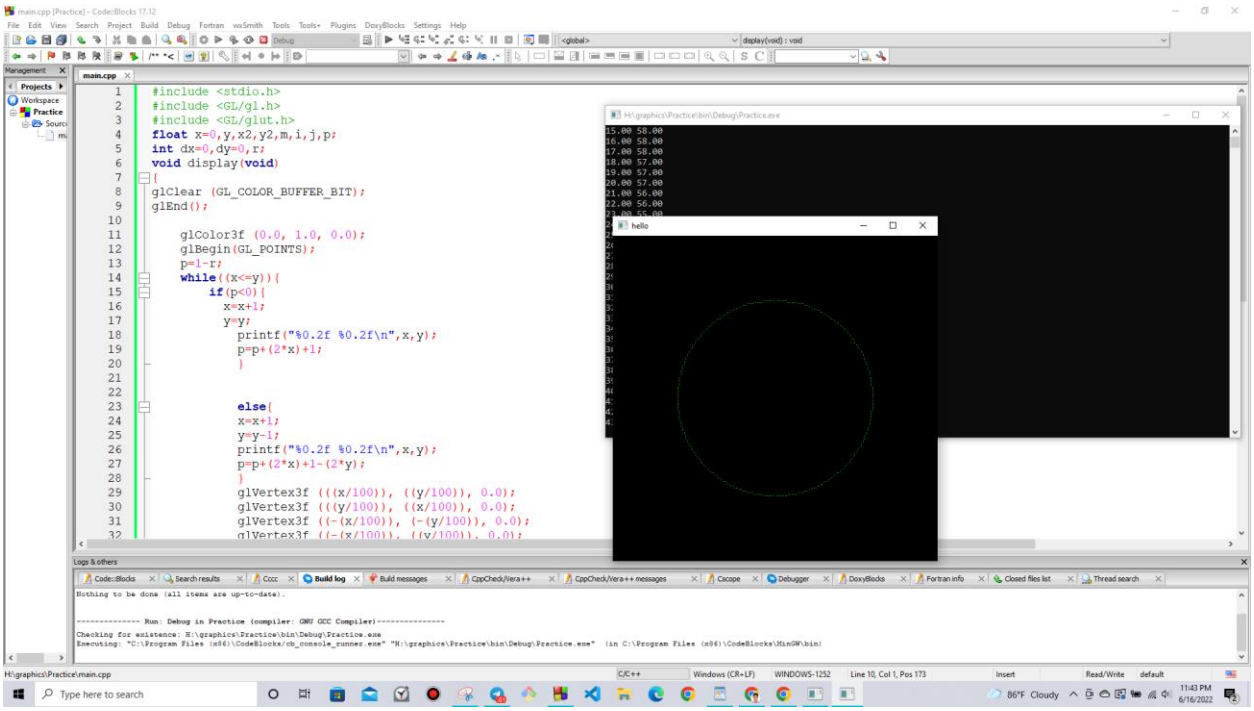
```
#include <stdio.h>

#include <GL/gl.h>

#include <GL/glut.h>

float x=0,y,x2,y2,m,i,j,p;

int dx=0,dy=0,r;

void display(void)

{

glClear (GL_COLOR_BUFFER_BIT);

glEnd();


   glColor3f (0.0, 1.0, 0.0);

   glBegin(GL_POINTS);

   p=1-r;


   while((x<=y)){

      if(p<0){

        x=x+1;

        y=y;
```

```
            printf("%0.2f %0.2f\n",x,y);
            p=p+(2*x)+1;
            }



            else{
            x=x+1;
            y=y-1;
            printf("%0.2f %0.2f\n",x,y);
            p=p+(2*x)+1-(2*y);
            }
            glVertex3f (((x/100)), ((y/100)), 0.0);
            glVertex3f (((y/100)), ((x/100)), 0.0);
            glVertex3f ((-(x/100)), (-(y/100)), 0.0);
            glVertex3f ((-(x/100)), ((y/100)), 0.0);
            glVertex3f (((x/100)), (-(y/100)), 0.0);
            glVertex3f (((y/100)), (-(x/100)), 0.0);
            glVertex3f ((-(y/100)), (-(x/100)), 0.0);
            glVertex3f ((-(y/100)), ((x/100)), 0.0);
        }
    glEnd();
    glFlush ();
```

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

int main(int argc, char** argv)
{
   printf("Enter radius: ");
   scanf("%d",&r);
   y=r;

glutInit(&argc, argv);
glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize (500, 500);
glutInitWindowPosition (100, 100);
glutCreateWindow ("hello");
init ();
glutDisplayFunc(display);
```

```
glutMainLoop();

return 0;

}
```

Output:



----END----

Lab Report

Lab Report No: 04

Course Code: CSE422

Course Title: Computer Graphics Lab

Lab Title: National Flag Drawing in OpenGL

Submitted To

Mr Tanim Ahmed
Lecturer
Department of CSE
Daffodil International University

Submitted By

Fazley Atif Maruf
ID: 191-15-2349
Section: Pc-C
Department of CSE, DIU

Date of Submission: 19-06-2022

Lab Name: National Flag Drawing in OpenGL

Lab Outcome: In this work I covered how to draw a national flag using OpenGL.

Functionalities: In this lab task have some functionalities like as GL_GUADS, GL_POLYGON and so on.

Source Code:

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

#include <GL/glut.h>

#include <math.h>

double points[45][45][3], r = 1.5, s = 0.00681817;

double pi = acos(-1);

void init(void)

{

   glClearColor(1.0, 1.0, 1.0, 1.0);

   glEnable(GL_DEPTH_TEST);

   glShadeModel(GL_SMOOTH);

   glDepthFunc(GL_LEQUAL);


   for (int x=0; x<45; x++) {

      for (int y=0; y<45; y++) {

                        points[x][y][0] = double((x / 3.0f) - 7.25f);

                        points[x][y][1] = double((y / 5.0f) - 4.5f);
```

```
                        points[x][y][2] = double(sin((((x / 5.0f) * 40.0f) / 360.0f) * pi *
2.0f));
                }
        }
}

void rectangle(void)
{
   int x, y;
        double double_x, double_y, double_xb, double_yb;

        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glLoadIdentity();

        glColor3f(0.0, 0.4, 0.3);
        glTranslatef(0.0f, 0.0f, -12.0f);

        glBegin(GL_QUADS);


        glColor3f(0.0, 0.4, 0.3);
        glVertex3f(-5.0f, -2.5f, 0.0f);
```

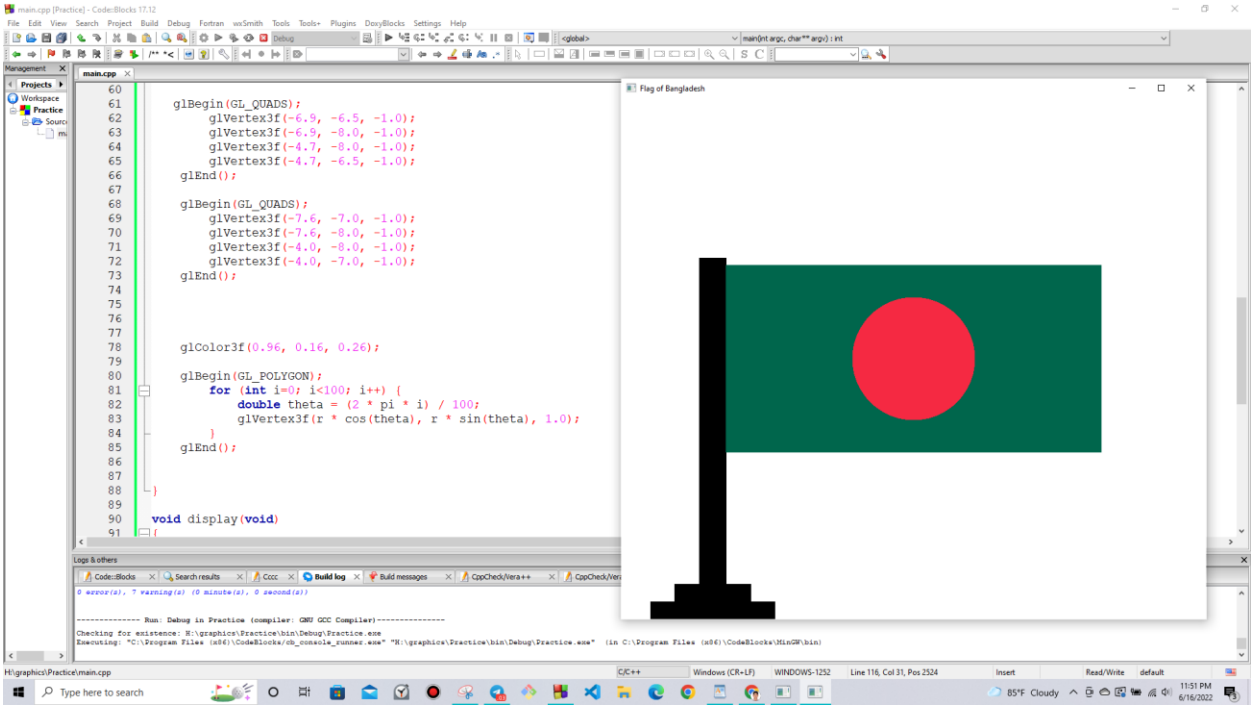
```
        glVertex3f(5.0f, -2.5f, 0.0f);
        glVertex3f(5.0f, 2.5f, 0.0f);
  glVertex3f(-5.0f, 2.5f, 0.0f);

        glEnd();

        glColor3f(0.0, 0.0, 0.0);
     glBegin(GL_QUADS);
     glVertex3f(-6.2, 2.9, -1.0);
     glVertex3f(-6.2, -8.0, -1.0);
     glVertex3f(-5.4, -8.0, -1.0);
     glVertex3f(-5.4, 2.9, -1.0);
  glEnd();

 glBegin(GL_QUADS);
     glVertex3f(-6.9, -6.5, -1.0);
     glVertex3f(-6.9, -8.0, -1.0);
     glVertex3f(-4.7, -8.0, -1.0);
     glVertex3f(-4.7, -6.5, -1.0);
  glEnd();

  glBegin(GL_QUADS);
```

```
        glVertex3f(-7.6, -7.0, -1.0);
        glVertex3f(-7.6, -8.0, -1.0);
        glVertex3f(-4.0, -8.0, -1.0);
        glVertex3f(-4.0, -7.0, -1.0);
    glEnd();





    glColor3f(0.96, 0.16, 0.26);


          glBegin(GL_POLYGON);
        for (int i=0; i<100; i++) {
           double theta = (2 * pi * i) / 100;
           glVertex3f(r * cos(theta), r * sin(theta), 1.0);
        }
          glEnd();
}
void display(void)
{
          rectangle();
          glutSwapBuffers();
```

```
}

void reshape(int w, int h)
{
    glViewport(0, 0, (GLsizei)w, (GLsizei)h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    gluPerspective(60.0, (GLfloat)w / (GLfloat)h, 1.0, 20.0);

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glutPostRedisplay();
}
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_RGB | GLUT_DEPTH | GLUT_DOUBLE);
    glutInitWindowSize(900, 800);
    glutInitWindowPosition(283, 100);
    glutCreateWindow("Flag of Bangladesh");
    init();
```

```
    glutDisplayFunc(display);

    glutReshapeFunc(reshape);

    glutMainLoop();


    return 0;

}
```

Output:



----END----

Lab Report

Lab Report No: 06

Course Code: CSE422

Course Title: Computer Graphics Lab

Lab Title: Bresenham Line Drawing in OpenGL

Submitted To

Mr Tanim Ahmed
Lecturer
Department of CSE
Daffodil International University

Submitted By

Fazley Atif Maruf
ID: 191-15-2349
Section: Pc-C
Department of CSE, DIU

Date of Submission: 19-06-2022

Lab Name: Bresenham Line Drawing in OpenGL

Lab Outcome: In this work I covered how to draw a line using Bresenham Line Algorithm. And It's totally work values of user input.

Source Code:

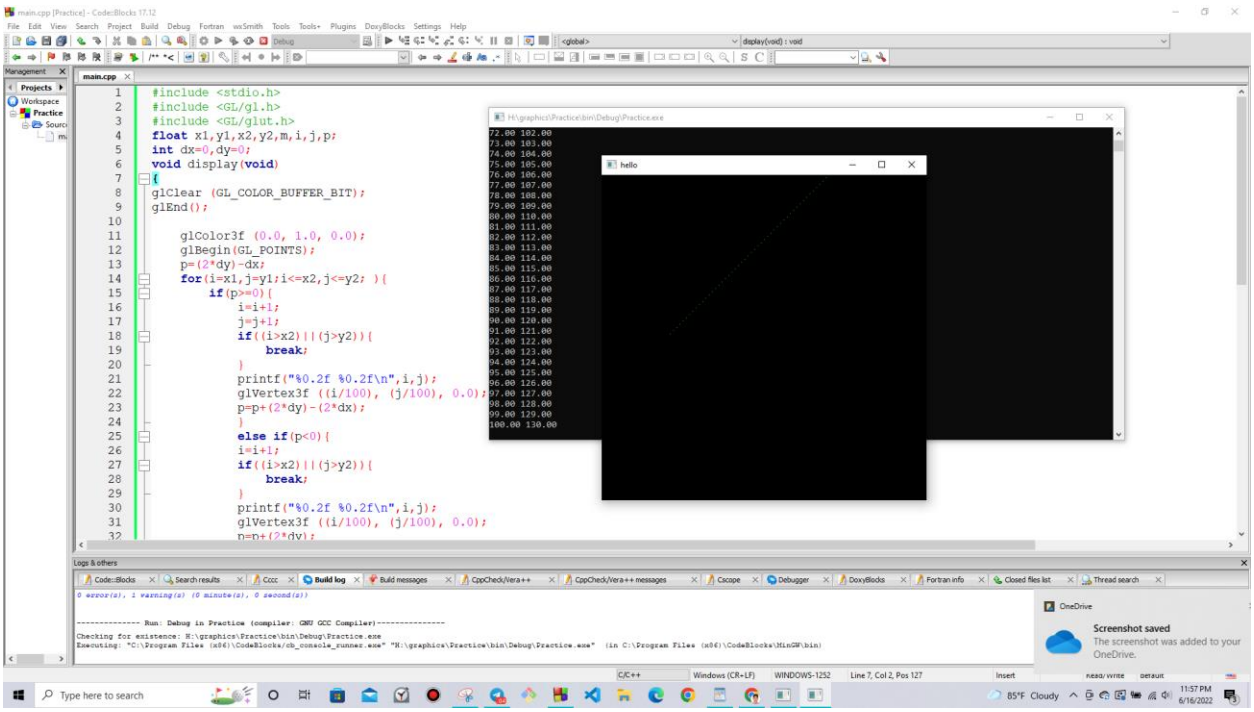
```
#include <stdio.h>

#include <GL/gl.h>

#include <GL/glut.h>

float x1,y1,x2,y2,m,i,j,p;

int dx=0,dy=0;

void display(void)

{

glClear (GL_COLOR_BUFFER_BIT);

glEnd();


   glColor3f (0.0, 1.0, 0.0);

   glBegin(GL_POINTS);

   p=(2*dy)-dx;

   for(i=x1,j=y1;i<=x2,j<=y2; ){

      if(p>=0){

         i=i+1;

         j=j+1;
```

```
            if((i>x2)||(j>y2)){
                break;
            }
            printf("%0.2f %0.2f\n",i,j);
            glVertex3f ((i/100), (j/100), 0.0);
            p=p+(2*dy)-(2*dx);
            }
            else if(p<0){
            i=i+1;
            if((i>x2)||(j>y2)){
                break;
            }
            printf("%0.2f %0.2f\n",i,j);
            glVertex3f ((i/100), (j/100), 0.0);
            p=p+(2*dy);
            }
        }
    glEnd();
    glFlush ();
    }
    void init (void)
    {
```

```
/* select clearing (background) color */
glClearColor (0.0, 0.0, 0.0, 0.0);
/* initialize viewing values */
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);
}

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

   printf("Enter first point: ");
   scanf("%f %f",&x1,&y1);
   printf("Enter second point: ");
   scanf("%f %f",&x2,&y2);
   dx=x2-x1;
   dy=y2-y1;
glutInit(&argc, argv);
glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize (500, 500);
glutInitWindowPosition (100, 100);
glutCreateWindow ("hello");
```

```
init ();

glutDisplayFunc(display);

glutMainLoop();

return 0; /* ISO C requires main to return int. */

}
```

Output:



----END----

Lab Report

Lab Report No: 03

Course Code: CSE422

Course Title: Computer Graphics Lab

Lab Title: Rotate Home in OpenGL

Submitted To

Mr Tanim Ahmed
Lecturer
Department of CSE
Daffodil International University

Submitted By

Fazley Atif Maruf
ID: 191-15-2349
Section: Pc-C
Department of CSE, DIU

Date of Submission: 19-06-2022

Lab Name: Rotate Home in OpenGL

Lab Outcome: In this work I covered how to draw a line using Bresenham Line Algorithm. And It's totally work values of user input.

Source Code:

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

#include <GL/glut.h>

//Initializes 3D rendering

void initRendering() {

	glEnable(GL_DEPTH_TEST);

	glEnable(GL_COLOR_MATERIAL); //Enable color

	glClearColor(0.5f, 0.5f, 0.0f, 0.0f); //Change the background to sky blue

}


//Called when the window is resized

void handleResize(int w, int h) {

	glViewport(0, 0, w, h);

	glMatrixMode(GL_PROJECTION);

	glLoadIdentity();

	gluPerspective(45.0, (double)w / (double)h, 1.0, 200.0);

}


float _angle = 30.0f;
```

```
float _cameraAngle = 0.0f;

//Draws the 3D scene
void drawScene() {
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

	glMatrixMode(GL_MODELVIEW);
	glLoadIdentity();
	glRotatef(-_cameraAngle, 0.0f, 1.0f, 0.0f);
	glTranslatef(0.0f, 0.0f, -9.0f);

	glPushMatrix();
	glTranslatef(0.0f, -1.0f, 0.0f);
	glRotatef(_angle, 0.0f, 0.0f, -1.0f);

	glBegin(GL_QUADS);
	glColor3f(0.5, 0.3, 0.2);
	glVertex3f(-1.0f, -1.5f, 0.0f);
	glVertex3f(0.0f, -1.5f, 0.0f);
	glVertex3f(0.0f, 0.5f, 0.0f);
	glVertex3f(-1.0f, 0.5f, 0.0f);
	glEnd();
```

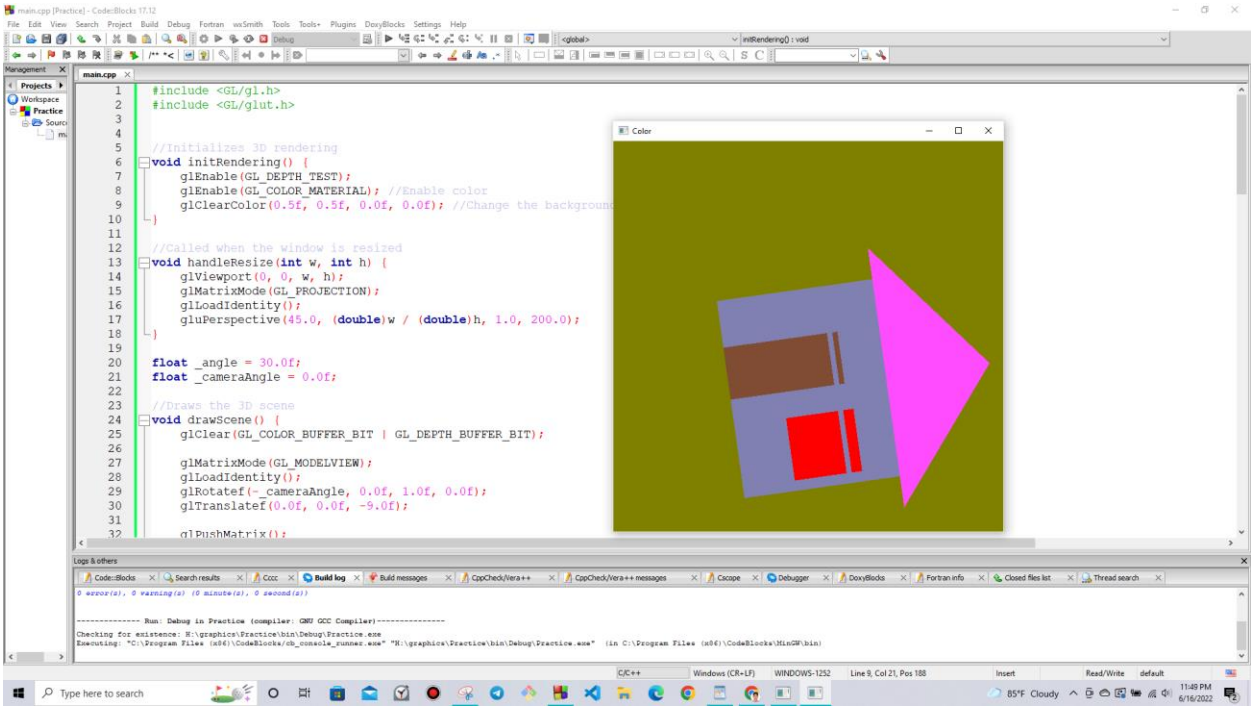
```
glBegin(GL_QUADS);
glColor3f(0.5, 0.3, 0.2);
glVertex3f(-1.0f, 0.6f, 0.0f);
glVertex3f(0.0f, 0.6f, 0.0f);
glVertex3f(0.0f, 0.7f, 0.0f);
glVertex3f(-1.0f, 0.7f, 0.0f);
glEnd();

glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.0f);
glVertex3f(0.5f, -0.5f, 0.0f);
glVertex3f(1.7f, -0.5f, 0.0f);
glVertex3f(1.7f, 0.5f, 0.0f);
glVertex3f(0.5f, 0.5f, 0.0f);
glEnd();

glBegin(GL_QUADS);
glColor3f(1.0f, 0.0f, 0.0f);
glVertex3f(0.5f, 0.6f, 0.0f);
glVertex3f(1.7f, 0.6f, 0.0f);
glVertex3f(1.7f, 0.8f, 0.0f);
```

```
glVertex3f(0.5f, 0.8f, 0.0f);
glEnd();

glBegin(GL_QUADS);
glColor3f(0.5, 0.5, 0.7);
glVertex3f(-1.9f, -1.5f, 0.0f);
glVertex3f(1.9f, -1.5f, 0.0f);
glVertex3f(1.9f, 1.5f, 0.0f);
glVertex3f(-1.9f, 1.5f, 0.0f);
glEnd();

glBegin(GL_TRIANGLES);
glColor3f(0.0f, 1.0f, 0.0f);
glEnd();
glBegin(GL_TRIANGLES);
//Triangle
glColor3f(1.0, 0.3, 1.0);
glVertex3f(-2.5f, 1.5f, 0.0f);
glVertex3f(2.5f, 1.5f, 0.0f);
glVertex3f(0.0f, 3.5f, 0.0f);

glEnd();
```

```
	glPopMatrix();

	glutSwapBuffers();
}

void update(int value) {
	_angle += 2.0f;
	if (_angle > 360) {
		_angle -= 360;
	}

	glutPostRedisplay(); ////Tell GLUT that the scene has changed
	glutTimerFunc(25, update, 0);
}

int main(int argc, char** argv) {
	//Initialize GLUT
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
	glutInitWindowSize(600, 600);

	//Create the window
```

```
    glutCreateWindow("Color");
    initRendering();
    //Set handler functions
    glutDisplayFunc(drawScene);
    glutReshapeFunc(handleResize);
    glutTimerFunc(25, update, 0); //Add a timer
    glutMainLoop();
    return 0;
}
```

Output:



----END----

Lab Report

Lab Report No: 07

Course Code: CSE422

Course Title: Computer Graphics Lab

Lab Title: 3D Lighting in OpenGL

Submitted To

Mr Tanim Ahmed
Lecturer
Department of CSE
Daffodil International University

Submitted By

Fazley Atif Maruf
ID: 191-15-2349
Section: Pc-C
Department of CSE, DIU

Date of Submission: 19-06-2022

Lab Name: 3D Lighting in OpenGL

Lab Outcome: In this work I presented a 3D Lighting object. Which is implemented by OpenGL.

Source Code:

```
#include <iostream>

#include <stdlib.h>

#include<windows.h>

#ifdef __APPLE__

#include <OpenGL/OpenGL.h>

#include <GLUT/glut.h>

#else

#include <GL/glut.h>

#endif

using namespace std;

void handleKeypress(unsigned char key, int x, int y) {

	switch (key) {

		case 27:

			exit(0);

	}

}

void initRendering() {

	glEnable(GL_DEPTH_TEST);
```

```
	glEnable(GL_COLOR_MATERIAL);
	glEnable(GL_LIGHTING);
	glEnable(GL_LIGHT0);
	glEnable(GL_LIGHT1);
	glEnable(GL_NORMALIZE);
}


//Called when the window is resized
void handleResize(int w, int h) {
	glViewport(0, 0, w, h);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	gluPerspective(45.0, (double)w / (double)h, 1.0, 200.0);
}

float _angle = -70.0f;


//Draws the 3D scene
void drawScene() {
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
```

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0.0f, 0.0f, -8.0f);

GLfloat ambientColor[] = {0.2f, 0.2f, 0.2f, 1.0f};
glLightModelfv(GL_LIGHT_MODEL_AMBIENT, ambientColor);


//Add positioned light
GLfloat lightColor0[] = {0.5f, 0.5f, 0.5f, 1.0f}; //Color (0.5, 0.5, 0.5)
GLfloat lightPos0[] = {4.0f, 0.0f, 8.0f, 1.0f}; //Positioned at (4, 0, 8)
glLightfv(GL_LIGHT0, GL_DIFFUSE, lightColor0);
glLightfv(GL_LIGHT0, GL_POSITION, lightPos0);



//Add directed light
GLfloat lightColor1[] = {0.5f, 0.2f, 0.2f, 1.0f};

GLfloat lightPos1[] = {-1.0f, 0.5f, 0.5f, 0.0f};
glLightfv(GL_LIGHT1, GL_DIFFUSE, lightColor1);
glLightfv(GL_LIGHT1, GL_POSITION, lightPos1);
```

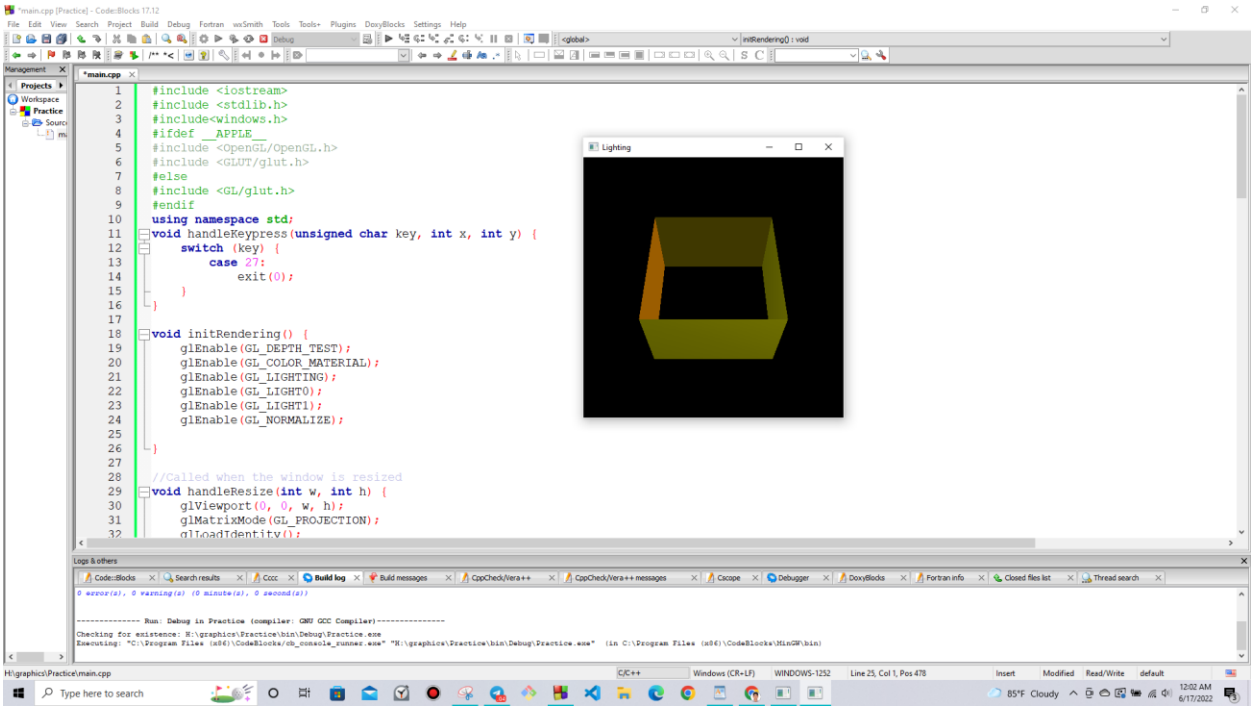
```
glRotatef(_angle, 1.0f, 0.0f, 0.0f);
glColor3f(1.0f, 1.0f, 0.0f);
glBegin(GL_QUADS);

glNormal3f(0.0f, 0.0f, 1.0f);
glVertex3f(-1.5f, -1.0f, 1.5f);
glVertex3f(1.5f, -1.0f, 1.5f);
glVertex3f(1.5f, 1.0f, 1.5f);
glVertex3f(-1.5f, 1.0f, 1.5f);


//Right
glNormal3f(1.0f, 0.0f, 0.0f);
glVertex3f(1.5f, -1.0f, -1.5f);
glVertex3f(1.5f, 1.0f, -1.5f);
glVertex3f(1.5f, 1.0f, 1.5f);
glVertex3f(1.5f, -1.0f, 1.5f);
```

```
	//Back
	glNormal3f(0.0f, 0.0f, -1.0f);
	glVertex3f(-1.5f, -1.0f, -1.5f);
	glVertex3f(-1.5f, 1.0f, -1.5f);
	glVertex3f(1.5f, 1.0f, -1.5f);
	glVertex3f(1.5f, -1.0f, -1.5f);

	//Left
	glNormal3f(-1.0f, 0.0f, 0.0f);
	glVertex3f(-1.5f, -1.0f, -1.5f);
	glVertex3f(-1.5f, -1.0f, 1.5f);
	glVertex3f(-1.5f, 1.0f, 1.5f);
	glVertex3f(-1.5f, 1.0f, -1.5f);
	glEnd();
	glutSwapBuffers();
}

void update(int value) {
	_angle += 1.5f;
	if (_angle > 360) {
		_angle -= 360;
	}
```

```
	glutPostRedisplay();
	glutTimerFunc(25, update, 0);
}

int main(int argc, char** argv) {
	//Initialize GLUT
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
	glutInitWindowSize(400, 400);

	//Create the window
	glutCreateWindow("Lighting ");
	initRendering();



	//Set handler functions
	glutDisplayFunc(drawScene);
	glutKeyboardFunc(handleKeypress);
	glutReshapeFunc(handleResize);
	glutTimerFunc(25, update, 0); //Add a timer
	glutMainLoop();
```

```
        return 0;

}
```

Output:



----END----