EXPERIMENT 2:

DDA LINE:

```
/*
#include <GL/glut.h>
#include <iostream>

#include <math.h>
using namespace std;
float r, g, b, x, y;
float x_1, x_2, y_1, y_2;
float xin, yin, length;
bool flag = true;
//int counter = 0;



void mouse(int button, int state,
   int mousex, int mousey)
{
   if (button == GLUT_LEFT_BUTTON
      && state == GLUT_DOWN) {
      flag = true;
      x = mousex;
      y = 480 - mousey;
   }

   // Change color of circle



      // Redisplay
     //  glutPostRedisplay();
```

```
}

int sgn(float a) {

   if (a == 0) {

      return 0;
   }
   if (a < 0) {

      return -1;
   }
   else
      return 1;



}

void Line() {
   cout << "x_1=" << x_1 << " y_1=" << y_1;
   cout << "x_2=" << x_2 << " y_2=" << y_2;

   float dy, dx, length;
   x_2 = x;
   y_2 = y;
   dy = y_2 - y_1;
   dx = x_2 - x_1;

   if (abs(dx) >= abs(dy)) {
```

```
        length = abs(dx);

    }
    else {
        length = abs(dy);
    }

    float xin, yin;

    xin = (x_2 - x_1) / length;
    yin = (y_2 - y_1) / length;

    float x, y;

    x = x_1 + 0.5 * sgn(xin);
    y = y_1 + 0.5 * sgn(yin);

    int i = 0;
    while (i <= length) {

        glBegin(GL_POINTS);
        glVertex2i(x, y);
        glEnd();

        x = x + xin;
        y = y + yin;
        i++;
    }

    glFlush();
}
```

```
void init(void)
{
   glClearColor(0, 0, 0, 0);
   glColor3f(1.0, 1.0, 0.0);
   gluOrtho2D(0, 640, 0, 480);
   glClear(GL_COLOR_BUFFER_BIT);
}

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

   cout << " Enter x1, y1 point";
   //cin >> x_1 >> y_1;
   cout << "\n Enter x2, y2 point";
   //cin >> x_2 >> y_2;

   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
   glutInitWindowSize(0, 600);
   glutCreateWindow("DDA Line");
   init();
   glutMouseFunc(mouse);
   glutDisplayFunc(Line);
   glutMainLoop();
   //cout << "x_1=" << x_1 << " y_1=" << y_1;
   //cout << "x_2=" << x_2 << " y_2=" << y_2;
   return 0;
}
*/
```

BRESENHAM LINE:

```
/*DDA Line Drawing Algorithm Implementation*/
```

```
#include<GL/glut.h>
#include<iostream>
#include<math.h>

using namespace std;

float r, g, b, x, y;
float x_1, x_2, y_1, y_2;
float xin, yin, length;
bool flag = true;

void init(void)
{
	glClearColor(1.0, 1.0, 0.0, 0.0);
	glColor3f(1.0, 0.0, 0.0);
	gluOrtho2D(0, 640, 0, 480);
	glClear(GL_COLOR_BUFFER_BIT);
}

void mouse(int button, int state, int mousex, int mousey)
{
	if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
	{
		flag = true;
		x = mousex;
		y = 480 - mousey;
	}
	cout << "\n\n mousex = " << x;
	cout << "mousey = " << y;
```

```
}


int sgn(float d)
{

	if (d == 0)
	{
		return 0;
	}
	else if (d < 0)
	{
		return -1;
	}
	else
	{
		return 1;
	}
}


void Line() {

	float dy, dx, length,G;
	x_2 = x;
	y_2 = y;
	dx = x_2 - x_1;
	dy = y_2 - y_1;

	G = (2 * dy) - dx;
```

```
if (abs(dx) >= abs(dy))
{
	length = abs(dx);
}
else
{
	length = abs(dy);
}
int j = 0;
x = x_1;
y = y_1;



while (j <= length)
{

	if (abs(dx) >= abs(dy))
	{
		x = x + 1;
		if (G >= 0)
		{
			y = y + 1;
			G = G + 2 * (dy - dx);
		}
		else
		{
			G = G + (2 * dy);
		}

	}
	else
```

```
		{
			y = y + 1;
			if (G >= 0)
			{
				x = x + 1;
				G = G + 2 * (dy - dx);
			}
			else
			{
				G = G + (2 * dy);
			}
		}
		cout << "\n x = " << x;
		cout << " y = " << y;
		x = x + 1; // to show dotted line we skipped alternate points
		y = y + 1; // to show dotted line we skipped alternate points
		glBegin(GL_POINTS);
		glVertex2i(x, y);
		glEnd();

		j++;
	}
	glFlush();


}

int main(int argc, char** argv)
{
	cout << "Enter the first point of a line\n";
	cin >> x_1 >> y_1;
```

```
	//cout << "Enter the last point of a line\n";
	//cin >> x_2 >> y_2;
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowSize(0, 600);
	glutCreateWindow("Bresenhams Line Drawing");
	init();
	glutMouseFunc(mouse);
	glutDisplayFunc(Line);
	glutMainLoop();
	return 0;
}
```