SET-2

Problem-1

A gaming company is designing a **2D racing game** where cars move on roads represented on a pixel grid. To display the race track on the screen, the game engine needs to draw straight paths between checkpoints. Since the track must be shown at the pixel level, the engine has to calculate which pixels should be illuminated to display the road segment.

Write a program using Bresenham's Line Drawing Algorithm to draw the path between two points **(30, 20)** and **(80, 90)**.

Solution-1

```
#include <GL/glut.h>
#include <bits/stdc++.h>
#include <windows.h>
using namespace std;

void init(void) {
   glClearColor(1, 1, 1, 1);
   glMatrixMode(GL_PROJECTION);
   gluOrtho2D(-100, 100, -100, 100);
}


void bresenham(int x0, int y0, int x1, int y1) {
   if (x0 > x1 || y0 > y1) {
      swap(x0, x1);
      swap(y0, y1);
   }

   int dx = abs(x1 - x0);
   int dy = abs(y1 - y0);
   float m = dy / (float)dx;

   int x = x0;
   int y = y0;

   int x_inc = (x1 >= x0) ? 1 : -1;
   int y_inc = (y1 >= y0) ? 1 : -1;

   glBegin(GL_POINTS);

   if (m < 1) {
      int p = 2 * dy - dx;
```

```
        while (x <= x1) {
            glVertex2i(x, y);

            x += x_inc;

            if (p < 0) {
                p += 2 * dy;
            }
            else {
                y += y_inc;
                p += 2 * dy - 2 * dx;
            }
        }
    }
    else {
        int p = 2 * dx - dy;

        while (x <= x1) {
            glVertex2i(x, y);

            y += y_inc;

            if (p < 0) {
                p += 2 * dx;
            }
            else {
                x += x_inc;
                p += 2 * dx - 2 * dy;
            }
        }
    }

    glEnd();
}


void display() {
    glClear(GL_COLOR_BUFFER_BIT);


    glBegin(GL_LINES);
    glColor3f(0, 0, 0);
    glVertex2f(100, 0);  glVertex2f(-100, 0);
    glVertex2f(0, 100);  glVertex2f(0, -100);
    glEnd();
```

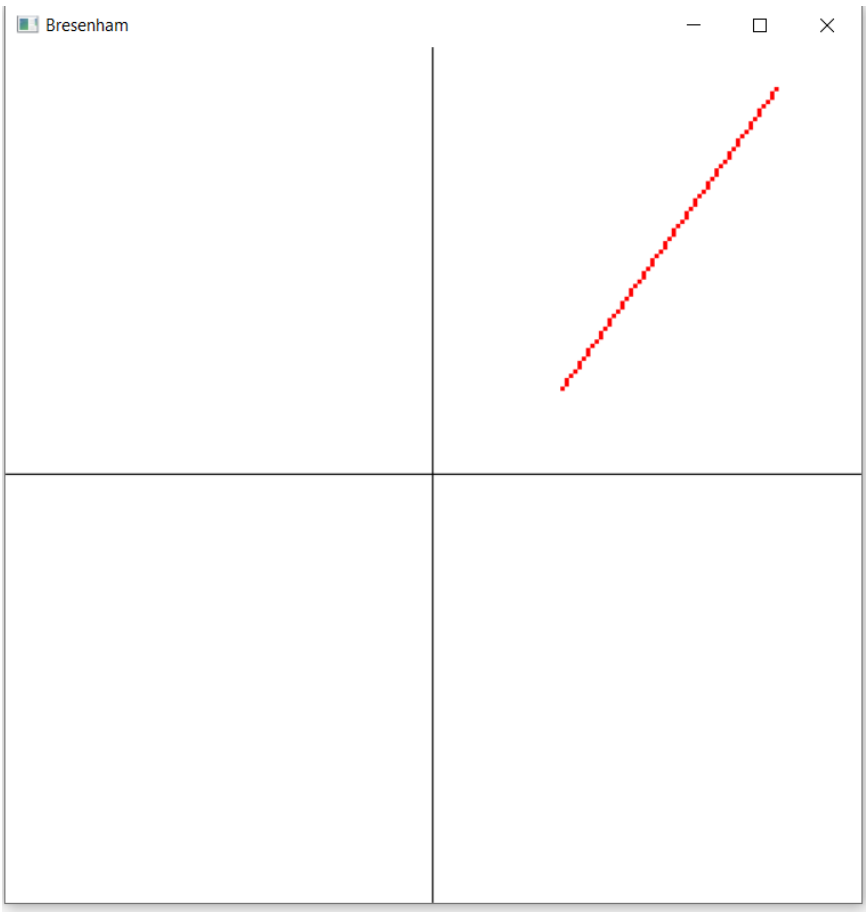
```
    glColor3f(1, 0, 0);
    glPointSize(3);
    bresenham(30, 20, 80, 90);

    glFlush();
}


int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(600, 600);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Bresenham");

    init();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

Output



Problem-2

Draw the given shape using **(GL_LINES)**. The window size should be **640 × 580**.

Solution-2

```
#include <windows.h>
#include <GL/glut.h>
#include <iostream>
using namespace std;

void display() {
   glClear(GL_COLOR_BUFFER_BIT);

   glBegin(GL_LINES);
   glVertex2f(0, 1);
   glVertex2f(0, -1);

   glVertex2f(-1, 0);
   glVertex2f(1, 0);
   glEnd();

   glBegin(GL_LINES);
   glVertex2f(-0.7, 0.7);
   glVertex2f(-0.5, 0.7);
   glVertex2f(-0.5, 0.7);
   glVertex2f(-0.3, 0.5);
   glVertex2f(-0.3, 0.5);
   glVertex2f(-0.3, 0.2);
   glVertex2f(-0.3, 0.2);
   glVertex2f(-0.7, 0.2);
   glVertex2f(-0.7, 0.2);
   glVertex2f(-0.7, 0.7);

   glEnd();


   glFlush();
}

int main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutCreateWindow("SHAPE");
   glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE | GLUT_DEPTH);
   glutInitWindowPosition(100, 100);
```

```
    glutInitWindowSize(640, 580);

    glutDisplayFunc(display);
    glutMainLoop();

    return 0;
}
```

Output