BRESENHAMS LINE DRAWING ALGORITHM

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

int x1, y1, x2, y2;

void myInit() {
   glClear(GL_COLOR_BUFFER_BIT);
   glClearColor(0.0, 0.0, 0.0, 1.0);
   glMatrixMode(GL_PROJECTION);
   glOrtho(0.0, 500.0, 0.0, 250.0, -250.0, 250.0);
}

void draw_pixel(int x, int y) {
   glBegin(GL_POINTS);
   glVertex2i(x, y);
   glEnd();
}

void draw_line(int x1, int x2, int y1, int y2) {
   int dx, dy, i, e;
   int incx, incy, inc1, inc2;
   int x, y;

   dx = x2 - x1;
   dy = y2 - y1;

   if (dx < 0) dx = -dx;
   if (dy < 0) dy = -dy;
   incx = (x2 < x1) ? -1 : 1;
```

```
incy = (y2 < y1) ? -1 : 1;
x = x1;
y = y1;

if (dx > dy) {
   draw_pixel(x, y);
   e = 2 * dy - dx;
   inc1 = 2 * (dy - dx);
   inc2 = 2 * dy;
   for (i = 0; i < dx; i++) {
      if (e >= 0) {
         y += incy;
         e += inc1;
      } else {
         e += inc2;
      }
      x += incx;
      draw_pixel(x, y);
   }
} else {
   draw_pixel(x, y);
   e = 2 * dx - dy;
   inc1 = 2 * (dx - dy);
   inc2 = 2 * dx;
   for (i = 0; i < dy; i++) {
      if (e >= 0) {
         x += incx;
         e += inc1;
      } else {
         e += inc2;
      }
```

```
            y += incy;
            draw_pixel(x, y);
        }
    }
}

void myDisplay() {
    draw_line(x1, x2, y1, y2);
    glFlush();
}

int main(int argc, char **argv) {
    cout << "Enter (x1, y1, x2, y2): ";
    cin >> x1 >> y1 >> x2 >> y2;

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("Bresenham's Line Drawing");
    myInit();
    glutDisplayFunc(myDisplay);
    glutMainLoop();

    return 0;
}
```

BRESENHAMS CIRCLE DRAWING ALGORITHM

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

int radius;
void MyInit()
{
   glClearColor(1,1,1,1);
   glColor3f(1,0,0);
   glOrtho(-500,500,500,-500,0,1);
}

void drawPoints(int x,int y)
{
   glBegin(GL_POINTS);
      glVertex2i(x,y);
      glVertex2i(-x,y);
      glVertex2i(x,-y);
      glVertex2i(-x,-y);

      glVertex2i(y,x);
      glVertex2i(-y,x);
      glVertex2i(y,-x);
      glVertex2i(-y,-x);
   glEnd();
}

void Circle()
```

```
{
   MyInit();

   int x=0, y=radius;
   drawPoints(x,y);

   int d = 3-2*radius;



   while(x<=y)
   {
      if(d < 0)
      {
         x = x+1;
         d = d + 4*x +6;
      }
      else if(d>=0)
      {
         x = x+1;
         y = y-1;
         d = d + 4*(x-y) + 10;
      }

      drawPoints(x,y);
   }

   glFlush();
}

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

```
    glutInit(&argc,argv);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(100,100);
    glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE);
    glutCreateWindow("Bresenham Circle");

    cout<<"Enter the radius of Circle:";
    cin>>radius;
    glutDisplayFunc(Circle);

    glutMainLoop();
    return 0;
}
```