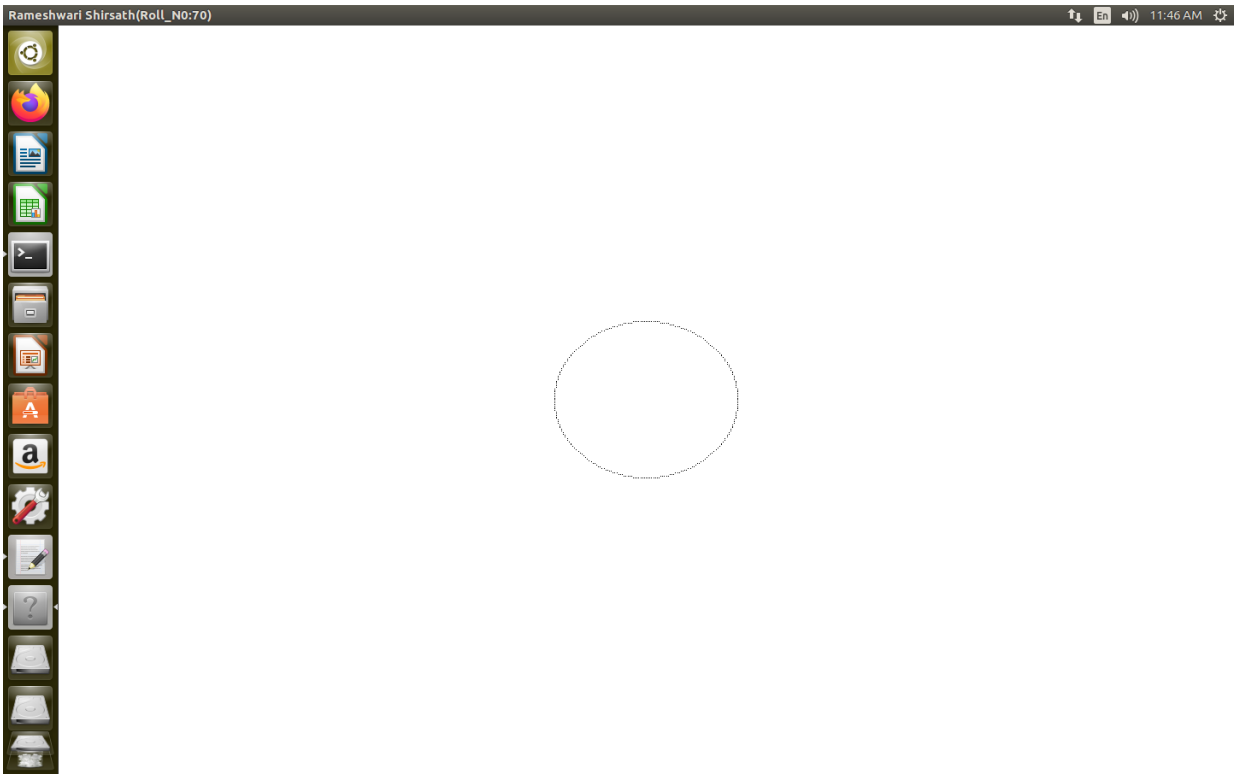
# Practical -3 CRICLE-1

**Implement Bresenham circle drawing algorithm to draw any object. The object should be displayed in all the quadrants with respect to center and radius**

```
#include<iostream>
#include<math.h>
#include<GL/glut.h>
using namespace std;
int xc = 320, yc = 240;

void plot_point(int x, int y)
{
      glBegin(GL_POINTS);
      glVertex2i(xc+x, yc+y);
      glVertex2i(xc+x, yc-y);
      glVertex2i(xc+y, yc+x);
      glVertex2i(xc+y, yc-x);
      glVertex2i(xc-x, yc-y);
      glVertex2i(xc-y, yc-x);
      glVertex2i(xc-x, yc+y);
      glVertex2i(xc-y, yc+x);
      glEnd();
}
void bresenham_circle(int r)
{
      int x=0,y=r;
      float d=3-2*r;
      plot_point(x,y);
      int k;
      while(x < y)
      {
            //x = x + 1;
            if(d < 0)
                  d=d+4*x+6;
            else
            {
                        y = y - 1;
                  d=d+4*(x-y)+10;
            }
            x=x+1;
            plot_point(x,y);
      }
      glFlush();
}

void display()
{
      int radius1=50;
      glClear(GL_COLOR_BUFFER_BIT);
      bresenham_circle(radius1);
}

void Init()
{
      /* Set clear color to white */
      glClearColor(1.0,1.0,1.0,0);
```

```
	/* Set fill color to black */
	glColor3f(0.0,0.0,0.0);
	gluOrtho2D(0 , 640 , 0 , 480);
}

int main(int argc, char **argv)
{
	glutInit(&argc,argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowPosition(0,0);
	glutInitWindowSize(640,480);
	glutCreateWindow("Rameshwari Shirsath Roll No:70");
	Init();
	glutDisplayFunc(display);
	glutMainLoop();
}
```

**OUTPUT:**




## Practical -3 CRICLE-2

Implement Bresenham circle drawing algorithm to draw any object. The object should be
displayed in all the quadrants with respect to center and radius.

```
#include <iostream>
#include <GL/glut.h>
#include <math.h>
using namespace std;
//Defalut radius of circle
int cx=300,cy=300,R=70; bool flag=1;
//Color struct
struct color{
GLubyte r,g,b;
};
//init function for init.
void init()
{
glClearColor(1,1,1,0);
glClear(GL_COLOR_BUFFER_BIT);
gluOrtho2D(0,600,0,600);
glColor3f(0,0,0);
}
//ploat the pixel (x,y)
void plotpixel(int x,int y)
{
glPointSize(1.5);
glBegin(GL_POINTS);
glVertex2i(x,y);
glEnd();
glFlush();
}
//ploat the points using the circle sym.
void octant(int xc,int yc,int x,int y)
{
plotpixel(xc+x,yc+y);
plotpixel(xc+y,yc+x);
plotpixel(xc+y,yc-x);
plotpixel(xc+x,yc-y);
plotpixel(xc-x,yc-y);
plotpixel(xc-y,yc- x);
plotpixel(xc-y,yc+x);
plotpixel(xc- x,yc+y);
}
//mid point circle drawing
void circleMP(int xc,int yc,int r)
{
int p=1-r,x=0,y=r;
//loop til the x become y equal to radius (r,r)
while(x<y)
{
octant(xc,yc,x,y);
x++;
if(p>0){ //if p>0 decrement the y and 2(x-y)+1
y--;
p+=2*(x-y)+1;
}
else{ //if p<=0 add 2x+1 to p
p+=2*x+1;
```

```
}
}
}
//convert the rad to deg
double ang(int q){

return (double)q*3.142/180;
}
void plottofill(int x,int y,color c)
{
glPointSize(1.0);
glColor3ub(c.r,c.g,c.b);
glBegin(GL_POINTS);
glVertex2i(x,y);
glEnd();
glFlush();
}
void seedfill(int x,int y,color oc,color nc)
{
color c;
glReadPixels(x,y,1,1,GL_RGB,GL_UNSIGNED_BYTE,&c);
if(c.r==oc.r&&c.b==oc.b&&c.g==oc.g)
{
plottofill(x,y,nc);
seedfill(x+1,y,oc,nc);
seedfill(x- 1,y,oc,nc);
seedfill(x,y+1,oc,nc);
seedfill(x,y-1,oc,nc);
}
}
//Draw all the Cirlces
void drawcircles(int x,int y,int r)
{
circleMP(x,y,r);
circleMP(x+2*r,y,r);
circleMP(x- 2*r,y,r);
circleMP(x+2*r*cos(ang(60)),y+2*r*sin(ang(60)),r);
circleMP(x-2*r*cos(ang(60)),y+2*r*sin(ang(60)),r);
circleMP(x-2*r*cos(ang(60)),y-2*r*sin(ang(60)),r);
circleMP(x+2*r*cos(ang(60)),y-2*r*sin(ang(60)),r);
circleMP(x,y,3*r);
circleMP(x,y,(float)2*r-r*(0.20));
}
//Display Function
void draw() {

}

//Clear the whole screen
void clear_screen(){

glClearColor(1,1,1,0);
glClear(GL_COLOR_BUFFER_BIT);
}
//Mouse click function
void mouseClick(int button,int state,int x,int y)
{
cout<<"Mouse Clicked"<<endl;
```

```
//First point to get the xc,yc
if(flag&&button==GLUT_LEFT_BUTTON&&state==GLUT_DOWN)
{
cout<<"Center Found"<<endl;
cx=x,cy=600-y;
glPointSize(5.0);
glColor3f(1,0,0);
glBegin(GL_POINTS);
glVertex2i(x,600-y);
glEnd();
glFlush();
flag=0;
}
//find the radius of the circle
else if (!flag&&button==GLUT_LEFT_BUTTON&&state==GLUT_DOWN)
{
cout<<"Ohhho !!, I got a radius"<<endl;
glColor3f(0,0,1);
glPointSize(1.0);
glBegin(GL_POINTS);
glVertex2i(x,600-y);
glEnd();
glFlush();
R=abs(x-cx);
flag=1;
}
}
//Menu function
void menu(int ch) {
color oc={255,255,255}; color nc={255,0,0};
switch(ch)
{
case 1:
      drawcircles(cx,cy,R);
      break;
case 2:
      clear_screen();
      break;
case 3:

      cout<<"Fill the Centered Circle"<<endl;
       seedfill(cx+5,cy,oc,nc);
      break;
case 4:
      exit(0);
      break;
}
}
int main(int agrc,char ** agrv)
{
glutInit(&agrc,agrv);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutInitWindowPosition(0,0);
glutInitWindowSize(600,600);
glutCreateWindow("Rameshwari Shirsath Roll No:70");
init();
glutDisplayFunc(draw);
glutCreateMenu(menu);
```

```
glutAddMenuEntry("Draw",1);
glutAddMenuEntry("Clear",2);
glutAddMenuEntry("Color Fill",3);
glutAddMenuEntry("Exit",4);
glutAttachMenu(GLUT_RIGHT_BUTTON);
glutMouseFunc(mouseClick);
glutMainLoop();
}
```

**OUTPUT:**