**Pune Institute Of Computer Technology Dhankawadi,**

**Pune – 43.**

Assignment No. 3

Computer Graphics

**SE-IT-10 ACADEMIC YEAR :- 2020-2021**

**Name :- Diptesh Ravindra Varule Roll No :- 23277**

**Topic Name**:

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

Code :

#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("Circle");

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:

