# Practical No 10

Title: - Write C++ program to draw the following pattern. Use DDA line and Bresenham drawing algorithm. Apply the concept of encapsulation.

Name:- Sattyam Sagar Chavan Roll No:- 73

Class:-SE AIDS Sub:-OOPL & CGL

# Input:

#include <iostream>

# include <graphics.h> # include <stdlib.h> using namespace std; class dcircle

{

private: int x0, y0; public:

dcircle()

{ x0=0; y0=0;

}

void setoff(int xx, int yy)

{

x0=xx; y0=yy;

}

void drawc(int x1, int y1, int r)

{

float d; int x,y; x=0;

y=r;

d=3-2\*r; do

{

putpixel(x1+x0+x, y0+y-y1, 15); putpixel(x1+x0+y, y0+x-y1,15); putpixel(x1+x0+y, y0-x-y1,15); putpixel(x1+x0+x,y0-y-y1,15); putpixel(x1+x0-x,y0-y-y1,15); putpixel(x1+x0-y, y0-x-y1,15); putpixel(x1+x0-y, y0+x-y1,15); putpixel(x1+x0-x, y0+y-y1,15); if (d<=0)

{

d = d+4\*x+6;

}

else

{

d=d+4\*(x-y)+10;

y=y-1;

}

x=x+1;

}

while(x<y);

}

};

class pt

{

protected: int xco, yco,color; public:

pt()

{

xco=0,yco=0,color=15;

}

void setco(int x, int y)

{

xco=x; yco=y;

}

void setcolor(int c)

{

color=c;

}

void draw()

{

putpixel(xco,yco,color);

}

};

class dline:public pt

{

private: int x2, y2; public:

dline():pt()

{ x2=0; y2=0;

}

void setline(int x, int y, int xx, int yy)

{

pt::setco(x,y); x2=xx; y2=yy;

}

void drawl( int colour)

{

float x,y,dx,dy,length; int i; pt::setcolor(colour); dx= abs(x2-xco); dy=abs(y2-yco);

if(dx>=dy)

{

length= dx;

}

else

{

length= dy;

}

dx=(x2-xco)/length; dy=(y2-yco)/length; x=xco+0.5; y=yco+0.5;

i=1;

while(i<=length)

{

pt::setco(x,y);

pt::draw(); x=x+dx; y=y+dy; i=i+1;

}

pt::setco(x,y);

pt::draw();

}

};

int main()

{

int gd=DETECT, gm; initgraph(&gd, &gm, NULL);

int x,y,r, x1, x2, y1, y2, xmax, ymax, xmid, ymid, n, i; dcircle c;

cout<<"\nenter coordinates of centre of circle : "; cout<<"\n enter the value of x : ";

cin>>x;

cout<<"\nenter the value of y : "; cin>>y;

cout<<"\nenter the value of radius : "; cin>>r;

xmax= getmaxx(); ymax=getmaxy(); xmid=xmax/2; ymid=ymax/2; setcolor(1); c.setoff(xmid,ymid);

line(xmid, 0, xmid, ymax); line(0,ymid,xmax,ymid); setcolor(15); c.drawc(x,y,r);

pt p1; p1.setco(100,100); p1.setcolor(14);

dline l;

l.setline(x1+xmid, ymid-y1, x2+xmid, ymid-y2); cout<<"Enter Total Number of lines : ";

cin>>n; for(i=0;i<n;i++)

{

cout<<"Enter co-ordinates of point x1 : "; cin>>x1;

cout<<"enter coordinates of point y1 : "; cin>>y1;

cout<<"Enter co-ordinates of point x2 : "; cin>>x2;

cout<<"enter coordinates of point y2 : "; cin>>y2;

l.setline(x1+xmid, ymid-y1, x2+xmid, ymid-y2); l.drawl(15);

}

cout<<"\nEnter coordinates of centre of circle : ";

cout<<"\n Enter the value of x : "; cin>>x;

cout<<"\nEnter the value of y : "; cin>>y;

cout<<"\nEnter the value of radius : "; cin>>r;

setcolor(5); c.drawc(x,y,r); getch(); delay(200); closegraph(); return 0;

}

# Output:

