**/\*WAP to implement Mid-Point Circle algorithm in Cpp.\*/**

**#include <iostream>//circle(x,y,r)**

**#include <cmath>**

**#include <graphics.h>**

**using namespace std;**

**int x,y;**

**float x\_n,y\_n,p\_n,r;**

**void draw\_circle()**

**{**

**if (p\_n<0) // if p\_n < 0 , x\_n=x\_n+1 , p\_n=p\_n+2\*x\_n+1**

**{**

**x\_n++; // change x\_n only**

**p\_n=p\_n+2\*x\_n+1;**

**}**

**else // if 0 < = p\_n , x\_n=x\_n+1 , y\_n=y\_n-1 , p\_n=p\_n+2\*y\_n+1**

**{**

**x\_n++; // change x\_n & y\_n**

**y\_n--;**

**p\_n=p\_n+2\*x\_n-2\*y\_n+1;**

**}**

**putpixel(x+x\_n,y+y\_n,GREEN); // 1st octant**

**putpixel(x+y\_n,y+x\_n,GREEN); // 2 nd octant**

**putpixel(x-y\_n,y+x\_n,GREEN); // 3 rd octant**

**putpixel(x-x\_n,y+y\_n,GREEN); // 4 th octant**

**putpixel(x-x\_n,y-y\_n,GREEN); // 5 th octant**

**putpixel(x-y\_n,y-x\_n,GREEN); // 6 th octant**

**putpixel(x+y\_n,y-x\_n,GREEN); // 7 th octant**

**putpixel(x+x\_n,y-y\_n,GREEN); // 8 th octant**

**}**

**int main()**

**{**

**int i;**

**while(1)**

**{**

**cout<<"\n\n\n\t\t\t\t\t1366\*768 ";**

**cout<<"\n\n\n\t\t Enter circle coordinates (x,y,r) with in range (0,0) to (1365,767)";**

**cout<<"\n\n Enter (x,y)";**

**cout<<"\n Enter x: ";**

**cin>>x;**

**cout<<" Enter y: ";**

**cin>>y;**

**cout<<"\n\n Enter r: ";**

**cin>>r;**

**x\_n=0;**

**y\_n=r;**

**p\_n=1.25-r; // p\_n = 5/4 -r**

**initwindow(1366,768);**

**for(i=0; i<=1365; i++) // creates white background**

**{**

**line(0,i,1365,i);**

**}**

**//setcolor(GREEN);**

**//circle(x,y,r+50);**

**putpixel(x,y,GREEN); // At center of circle**

**putpixel(x-r,y,GREEN); //At leftmost point**

**putpixel(x+r,y,GREEN); //At rightmost point**

**putpixel(x,y+r,GREEN); //At topmost point**

**putpixel(x,y-r,GREEN); //At bottom point**

**while (x\_n<=y\_n) // at 1st octant when angle = 45 degree x\_0 = y\_0**

**{**

**draw\_circle();**

**}**

**getch();**

**closegraph();**

**}**

**return 0;**

**}**

**/\*WAP to implement Mid-Point Circle algorithm in Cpp.\*/**

**#include<GL/gl.h>**

**#include<GL/glu.h>**

**#include<GL/glut.h>**

**//#include <bits/stdc++.h>**

**#include<iostream>**

**//for animation purpose**

**#include<vector>**

**using namespace std;**

**void display(); //display function**

**void reshape(int,int); //reshape the viewport**

**void timer(int); //for displaying no of frames in a sec**

**void getinfo(); //info from user**

**void drawCircle(); // drawing circle**

**int xc,yc,r,p;**

**void drawCircleAnimation(); //animation**

**void keyboard(unsigned char,int,int); //for animation keyboard input**

**int ax,ay,ar,ap; //for animation points**

**bool startAnimation=false;//for animation start**

**vector<int> point;//for animation**

**void init(){**

**glClearColor(0.1,0.1,0.1,1.0); //background color**

**}**

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

**getinfo();**

**glutInit(&argc,argv);**

**glutInitDisplayMode(GLUT\_RGB|GLUT\_DOUBLE);**

**glutInitWindowSize(500,500);**

**glutInitWindowPosition(200,200);**

**glutCreateWindow("Mid-Point-Circle");**

**glutDisplayFunc(display);**

**glutReshapeFunc(reshape);**

**glutSetKeyRepeat(GLUT\_KEY\_REPEAT\_OFF);**

**glutKeyboardFunc(keyboard);**

**glutTimerFunc(0,timer,0);**

**init();**

**glutMainLoop();**

**return 0;**

**}**

**void display(){**

**glClear(GL\_COLOR\_BUFFER\_BIT);**

**glLoadIdentity();**

**glColor3f(.7,.7,.7);//axis line color**

**glBegin(GL\_LINES);**

**glVertex2f(250,0);**

**glVertex2f(-250,0);**

**glVertex2f(0,250);**

**glVertex2f(0,-250);**

**glEnd();**

**glPointSize(3);**

**glBegin(GL\_POINTS);**

**glVertex2f(xc,yc);**

**glEnd();**

**glPointSize(1);**

**drawCircle();**

**drawCircleAnimation();**

**glutSwapBuffers();**

**}**

**void reshape(int w,int h){**

**glViewport(0,0,w,h);**

**glMatrixMode(GL\_PROJECTION);**

**glLoadIdentity;**

**gluOrtho2D(-250,250,-250,250);**

**glMatrixMode(GL\_MODELVIEW);**

**}**

**void timer(int){**

**glutPostRedisplay();**

**glutTimerFunc(1000/30,timer,0);**

**}**

**void getinfo(){**

**cout<<endl<<endl<<"\t Enter the following:"<<endl;**

**cout<<"\t Center x: ";**

**cin>>xc;**

**cout<<"\t Center y: ";**

**cin>>yc;**

**cout<<"\t radius r: ";**

**cin>>r;**

**//for animation**

**ax=0;**

**ay=r;**

**ar=r;**

**ap=1-r;**

**}**

**void drawCircle(){**

**int x,y;**

**p=1-r;**

**x=0;**

**y=r;**

**glColor3f(1,1,1);//circle color**

**glBegin(GL\_POINTS);**

**while(x<=y){**

**glVertex2f(xc+x,yc+y);**

**glVertex2f(xc+x,yc-y);**

**glVertex2f(xc-x,yc+y);**

**glVertex2f(xc-x,yc-y);**

**glVertex2f(xc+y,yc+x);**

**glVertex2f(xc+y,yc-x);**

**glVertex2f(xc-y,yc+x);**

**glVertex2f(xc-y,yc-x);**

**x+=1;**

**if(p<0)**

**p=p+2\*x+1;**

**else{**

**y=y-1;**

**p=p+2\*x-2\*y+1;**

**}**

**}**

**glEnd();**

**}**

**//For animation below here**

**void drawCircleAnimation(){**

**if(ax<=ay && startAnimation==true){**

**point.push\_back(xc+ax);**

**point.push\_back(yc+ay);**

**point.push\_back(xc+ax);**

**point.push\_back(yc-ay);**

**point.push\_back(xc-ax);**

**point.push\_back(yc+ay);**

**point.push\_back(xc-ax);**

**point.push\_back(yc-ay);**

**point.push\_back(xc+ay);**

**point.push\_back(yc+ax);**

**point.push\_back(xc+ay);**

**point.push\_back(yc-ax);**

**point.push\_back(xc-ay);**

**point.push\_back(yc+ax);**

**point.push\_back(xc-ay);**

**point.push\_back(yc-ax);**

**ax+=1;**

**if(ap<0)**

**ap=ap+2\*ax+1;**

**else{**

**ay=ay-1;**

**ap=ap+2\*ax-2\*ay+1;**

**}**

**}**

**if(ax<=ay)**

**glColor3f(1,0,0);**

**else**

**glColor3f(1,1,1);**

**glPointSize(1);**

**glBegin(GL\_POINTS);**

**for(int i=0;i<point.size();i+=2){**

**glVertex2f(point.at(i),point.at(i+1));**

**}**

**glEnd();**

**glPointSize(1);**

**}**

**void keyboard(unsigned char key,int x,int y){**

**if(key=='p')**

**startAnimation=true;**

**if(key=='o')**

**startAnimation=false;**

**}**