```
/*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;

    }
```