

```

#include <GL/glut.h>
#include<bits/stdc++.h>
using namespace std;

int xc = 250, yc = 250;

void init() {
    glClearColor(0,0,0,1);
    glColor3f(1,1,1);
    glPointSize(1);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0,500,0,500);
}

void drawPoint(pair<int,int>pixel){
    // add xc and yc to avoid negative co-ordinates
    glVertex2i(xc + pixel.first, yc + pixel.second);
}

void circle(){
    float r,p,x,y;
    cout<<"Enter Radius : "; cin>>r;
    p = 1 - r;
    x = 0, y = r;

    while(x<=y){
        drawPoint({x,y});
        drawPoint({y,x});

        drawPoint({x,-y});
        drawPoint({-y,x});

        drawPoint({-x,-y});
        drawPoint({-y,-x});

        drawPoint({-x,y});
        drawPoint({y,-x});

        if(p<0)
            p+=1+2*(x+1);
        else {
            p+=1+2*(x+1)-2*(y-1);

            y--;
        }
        x++;
    }
}

```

```
    }  
}
```

```
void display() {  
    glClear(GL_COLOR_BUFFER_BIT);  
    glBegin(GL_POINTS);  
  
    circle();  
  
    glEnd();  
    glFlush();  
}
```

```
int main(int argc, char** argv) {  
    glutInit(&argc, argv);  
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);  
    glutInitWindowSize(500, 500);  
    glutInitWindowPosition(1100, 600);  
    glutCreateWindow("Draw Circle");  
    init();  
    glutDisplayFunc(display);  
    glutMainLoop();  
    return 0;  
}
```



```
#include <GL/glut.h>
#include<bits/stdc++.h>
using namespace std;

void init() {
    glClearColor(0,0,0,1);
    glColor3f(1,1,1);
    glPointSize(1);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0,500,0,500);
}

void drawPoint(pair<int,int>pixel){
    glVertex2i(pixel.first,pixel.second);
}

void circle(int r,int xc,int yc){
    float p,x,y;

    p=1-r;
    x=0,y=r;
```

```

while(x<=y){
    drawPoint({ xc+x,yc+y });
    drawPoint({ xc+y,yc+x });

    drawPoint({ xc+x,yc-y });
    drawPoint({ xc-y,yc+x });

    drawPoint({ xc-x,yc-y });
    drawPoint({ xc-y,yc-x });

    drawPoint({ xc-x,yc+y });
    drawPoint({ xc+y,yc-x });

    if(p<0)
        p+=1+2*(x+1);
    else{
        p+=1+2*(x+1)-2*(y-1);
        y--;
    }
    x++;
}
}

```

```

void display() {
    float x,y,r;
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POINTS);

    cout<<"Enter radius : ";cin>>r;
    cout<<"Enter center ";cin>>x>>y;

    circle(r,x,y);

    glEnd();
    glFlush();
}

```

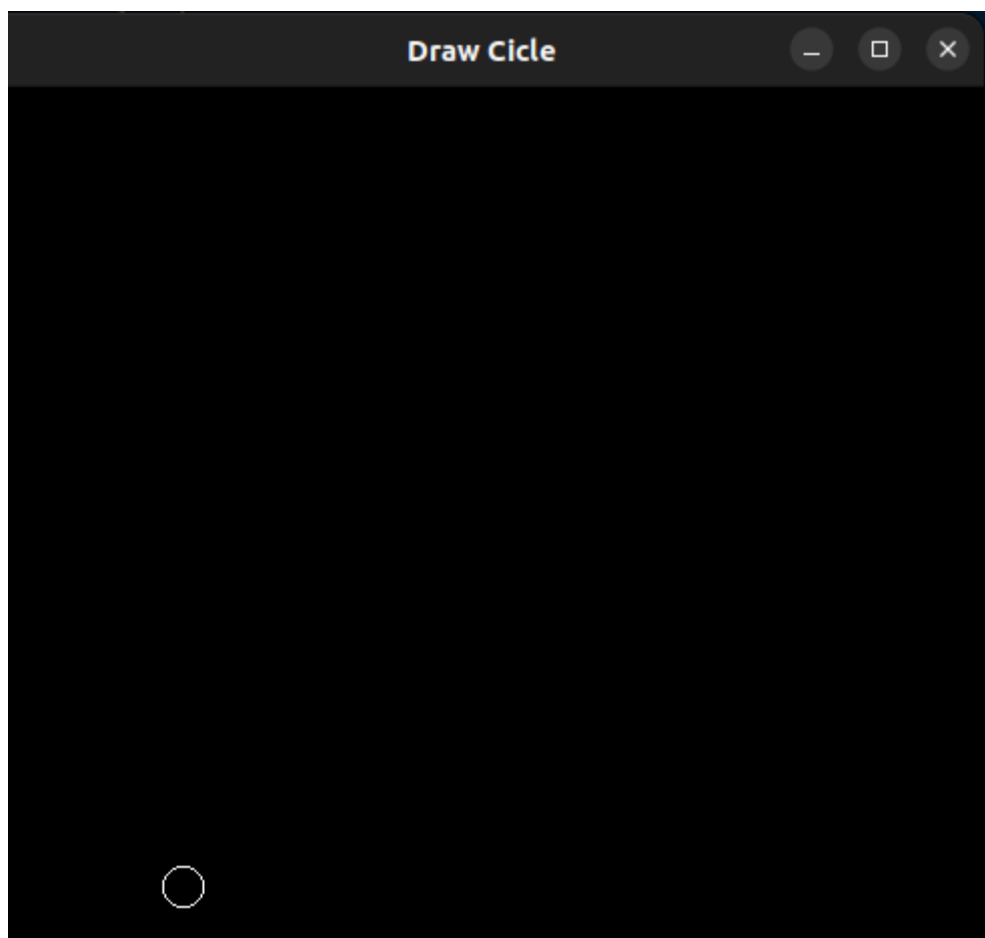
```

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

    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(1100,600);
    glutCreateWindow("Draw Cicle");
    init();
    glutDisplayFunc(display);
    glutMainLoop();
}

```

```
}    return 0;
```



```

#include <GL/glut.h>
#include<bits/stdc++.h>
using namespace std;

void init() {
    glClearColor(0,0,0,1);
    glColor3f(1,1,1);
    glPointSize(1);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0,500,0,500);
}

void drawPoint(pair<int,int>pixel){
    glVertex2i(pixel.first,pixel.second);
}

void concentricCircle(int r1,int r2){
    float p,x,y;

    p=1-r1;
    x=0,y=r1;

    while(x<=y){
        drawPoint({x+250,y+250});
        drawPoint({y+250,x+250});

        drawPoint({x+250,250-y});
        drawPoint({250-y,x+250});

        drawPoint({250-x,250-y});
        drawPoint({250-y,250-x});

        drawPoint({250-x,y+250});
        drawPoint({y+250,250-x});

        if(p<0)
            p+=1+2*(x+1);
        else{
            p+=1+2*(x+1)-2*(y-1);
            y--;
        }
        x++;
    }

    p=1-r2;

```

```

x=0,y=r2;

while(x<=y){
    drawPoint({x+250,y+250});
    drawPoint({y+250,x+250});

    drawPoint({x+250,250-y});
    drawPoint({250-y,x+250});

    drawPoint({250-x,250-y});
    drawPoint({250-y,250-x});

    drawPoint({250-x,y+250});
    drawPoint({y+250,250-x});

    if(p<0)
        p+=1+2*(x+1);
    else{
        p+=1+2*(x+1)-2*(y-1);
        y--;
    }
    x++;
}
}

```

```

void display() {
    float r1,r2;
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POINTS);

    cout<<"Enter radius 1 : ";cin>>r1;
    cout<<"Enter radius 2 : ";cin>>r2;

    concentricCircle(r1,r2);

    glEnd();
    glFlush();
}

```

```

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

    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(1100,600);
    glutCreateWindow("Draw star");
    init();
}

```

```
    glutDisplayFunc(display);  
    glutMainLoop();  
    return 0;  
}
```



```

#include <GL/glut.h>
#include<bits/stdc++.h>
using namespace std;

void init() {
    glClearColor(0,0,0,1);
    glColor3f(1,1,1);
    glPointSize(1);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0,500,0,500);
}

void drawPoint(pair<int,int>pixel){
    glVertex2i(pixel.first,pixel.second);
}

void circle(int r,int xc,int yc){
    float p,x,y;

    p=1-r;
    x=0,y=r;

    while(x<=y){
        drawPoint({ xc+x,yc+y });
        drawPoint({ xc+y,yc+x });

        drawPoint({ xc+x,yc-y });
        drawPoint({ xc-y,yc+x });

        drawPoint({ xc-x,yc-y });
        drawPoint({ xc-y,yc-x });

        drawPoint({ xc-x,yc+y });
        drawPoint({ xc+y,yc-x });

        if(p<0)
            p+=1+2*(x+1);
        else{
            p+=1+2*(x+1)-2*(y-1);
            y--;
        }
        x++;
    }
}

```

```

void display() {
    float x,y,r,c;

    vector<int>radius;
    vector<pair<int,int>>centre;

    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POINTS);

    cout<<"Enter number of circles : ";cin>>c;

    cout<<"Enter radius : ";
    for(int i=0;i<c;i++){
        cin>>r;
        radius.push_back(r);
    }

    cout<<"Enter centre : ";
    for(int i=0;i<c;i++){
        cin>>x>>y;
        centre.push_back({x,y});
    }

    for(int i=0;i<c;i++)
        circle(radius[i],centre[i].first,centre[i].second);

    glEnd();
    glFlush();
}

```

```

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

    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(1100,600);
    glutCreateWindow("Draw star");
    init();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}

```

```

#include <GL/glut.h>
#include<bits/stdc++.h>
using namespace std;

void init() {
    glClearColor(0,0,0,1);

    glPointSize(2);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0,500,0,500);
}

void drawPoint(pair<int,int>pixel){
    glVertex2i(pixel.first,pixel.second);
}

void circle(int r,int xc,int yc){
    float p,x,y;

    p=1-r;
    x=0,y=r;

    while(x<=y){
        drawPoint({xc+x,yc+y});
        drawPoint({xc+y,yc+x});

        drawPoint({xc+x,yc-y});
        drawPoint({xc-y,yc+x});

        drawPoint({xc-x,yc-y});
        drawPoint({xc-y,yc-x});

        drawPoint({xc-x,yc+y});
        drawPoint({xc+y,yc-x});

        if(p<0)
            p+=1+2*(x+1);
        else{
            p+=1+2*(x+1)-2*(y-1);
            y--;
        }
        x++;
    }
}

```

```
void display() {  
  
    glClear(GL_COLOR_BUFFER_BIT);  
    glBegin(GL_POINTS);  
  
    glColor3f(0,0,1);  
    circle(35,170,250);  
  
    glColor3f(1,1,1);  
    circle(35,250,250);  
  
    glColor3f(1,0,0);  
    circle(35,330,250);  
  
    glColor3f(1,1,0);  
    circle(35,210,220);  
  
    glColor3f(0,1,0);  
    circle(35,290,220);  
  
    glEnd();  
    glFlush();  
}
```

```
int main(int argc,char** argv) {  
  
    glutInit(&argc,argv);  
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);  
    glutInitWindowSize(500,500);  
    glutInitWindowPosition(1100,600);  
    glutCreateWindow("Draw star");  
    init();  
    glutDisplayFunc(display);  
    glutMainLoop();  
    return 0;  
}
```

