SSN COLLEGE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UCS1712 – GRAPHICS AND MULTIMEDIA LAB

EX NO: 4 – Midpoint Circle Drawing Algorithm

Name : Likhitha Verma A
Reg : 185001084
Date : 17/08/2021

1. AIM :

To write a C++ program using OpenGL to implement the Midpoint Circle drawing algorithm with radius and a centre given as user input.

ALGORITHM:

1. Pass coordinates of centre (xc, yc) and radius as parameters to midpoint function
2. Initialize x=0, y=r, p=5/4 – r
3. Plot (x + xc, y + yc)
4. While (y > x):
    a. If p < 0:
        i. Increment x by 1
        ii. p += 2*x + 1
    b. else:
        i. Decrement y by 1
        ii. Increment x by 1
        iii. p += 2 * (x - y) + 1
    c. Plot the points (x + xc, y + yc) for 7 different symmetric segments

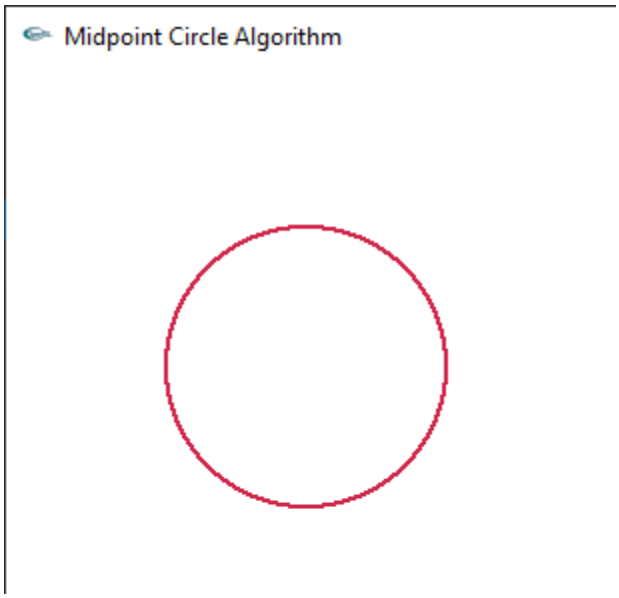
CODE :

```
#include<glut.h>
#include<math.h>
#include<stdio.h>
void myInit() {
      glClearColor(1.0, 1.0, 1.0, 0.0);
      glColor3f(0.82, 0.15, 0.29);
      glMatrixMode(GL_PROJECTION);
      glLoadIdentity();
```

```
	glPointSize(2);
	gluOrtho2D(0.0, 500.0, 0.0, 500.0);
}
void midPointCircleAlgo(int xc, int yc, int r)
{
	int x = 0;
	int y = r;
	float p = 5 / 4 - r;
	glBegin(GL_POINTS);
	glVertex2d(x+xc,y+yc);
	while (y > x)
	{
		if (p < 0)
		{
			x++;
			p += 2 * x + 1;
		}
		else
		{
			y--;
			x++;
			p += 2 * (x - y) + 1;
		}
		glVertex2d(x + xc, y + yc);
		glVertex2d(x + xc, -y + yc);
		glVertex2d(-x + xc, y + yc);
		glVertex2d(-x + xc, -y + yc);
		glVertex2d(y + xc, x + yc);
		glVertex2d(-y + xc, x + yc);
		glVertex2d(y + xc, -x + yc);
		glVertex2d(-y + xc, -x + yc);
	}
	glEnd();
}
void myDisplay() {
	glClear(GL_COLOR_BUFFER_BIT);
	midPointCircleAlgo(250, 250, 70);
	glFlush();
}
```

```
int main(int argc, char* argv[]) {
      glutInit(&argc, argv);
      glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
      glutInitWindowSize(500, 500);
      glutCreateWindow("Midpoint Circle Algorithm");
      glutDisplayFunc(myDisplay);
      myInit();
      glutMainLoop();
      return 1;
}
```

OUTPUT :



RESULT :

Thus a C++ program using OpenGL was written to implement the Midpoint Circle drawing algorithm successfully.

2. AIM :

To write a C++ program using OPENGL to replicate any circular object with the help of the Midpoint Circle algorithm. Use the necessary colours and elements to show details

ALGORITHM :

1. Pass coordinates of centre (xc, yc) and radius as parameters to midpoint function
2. Initialize x=0, y=r, p=5/4 – r
3. Plot (x + xc, y + yc)
4. While (y > x):
    a. If p < 0:
        i. Increment x by 1
        ii. p += 2*x + 1
    b. else:
        i. Decrement y by 1
        ii. Increment x by 1
        iii. p += 2 * (x - y) + 1
    c. Plot the points (x + xc, y + yc) for 7 different symmetric segments
5. To draw the olympic rings fix the center coordinates and radius and call the midpoint algorithm 5 times.

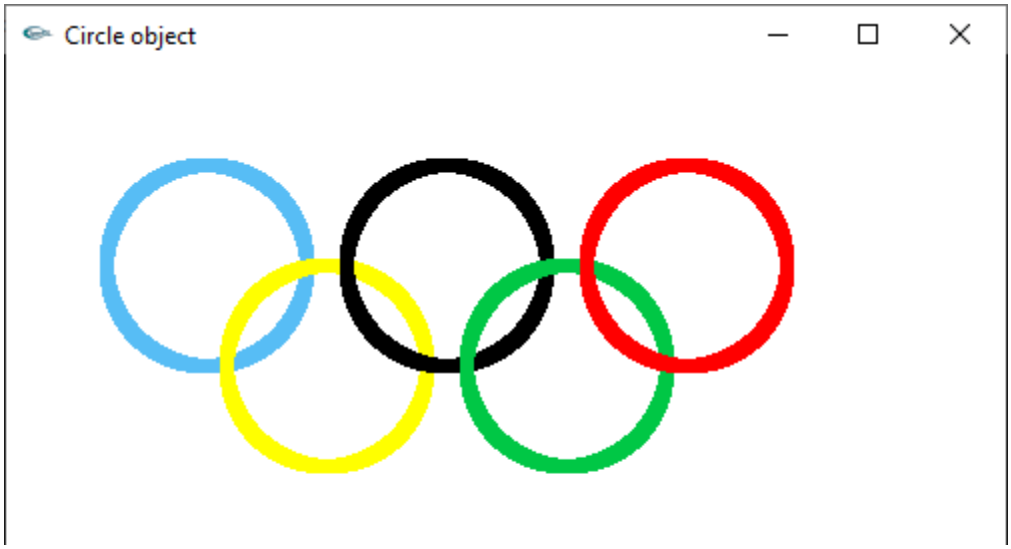
CODE :

```
#include<glut.h>
#include<math.h>
#include<stdio.h>
void myInit() {
	glClearColor(1.0, 1.0, 1.0, 0.0);
	glColor3f(0.34, 0.74, 0.96);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	glPointSize(7);
	gluOrtho2D(0.0, 500.0, 0.0, 500.0);
}
void midPointCircleAlgo(int xc, int yc, int r)
{
	int x = 0;
	int y = r;
	float p = 5 / 4 - r;
	glBegin(GL_POINTS);
	glVertex2d(x + xc, y + yc);

	while (y > x){
		if (p < 0){
			x++;
```

```
                p += 2 * x + 1;
            }
            else{
                y--;
                x++;
                p += 2 * (x - y) + 1;
            }
            glVertex2d(x + xc, y + yc);
            glVertex2d(x + xc, -y + yc);
            glVertex2d(-x + xc, y + yc);
            glVertex2d(-x + xc, -y + yc);
            glVertex2d(y + xc, x + yc);
            glVertex2d(-y + xc, x + yc);
            glVertex2d(y + xc, -x + yc);
            glVertex2d(-y + xc, -x + yc);
      }
      glEnd();
}
void myDisplay() {
      glClear(GL_COLOR_BUFFER_BIT);
      //blue ring
      midPointCircleAlgo(100, 400, 50);
      //yellow ring
      glColor3f(1.0, 1.0, 0);
      midPointCircleAlgo(160, 350, 50);
      //black ring
      glColor3f(0, 0, 0);
      midPointCircleAlgo(220, 400, 50);
      //green ring
      glColor3f(0, 0.78, 0.27);
      midPointCircleAlgo(280, 350, 50);
      //red ring
      glColor3f(1, 0, 0);
      midPointCircleAlgo(340, 400, 50);

      glFlush();
}
int main(int argc, char* argv[]) {
      glutInit(&argc, argv);
```

```
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowSize(500, 500);
	glutCreateWindow("Circle object");
	glutDisplayFunc(myDisplay);
	myInit();
	glutMainLoop();
	return 1;
}
```

OUTPUT :



RESULT :

Thus drew the Olympic rings in openGL using the midpoint circle drawing algorithm successfully.