# Circle Drawing: Polar, Non Polar, Midpoint

**Code:**

```
ffrom OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
import sys
import math

WINDOW_SIZE = 500
SCALE = 100
xc = yc = 0
r = 1

def init_display():
      glClear(GL_COLOR_BUFFER_BIT)
      glColor3f(1, 0, 0)
      glPointSize(5)
def midpoint_circle():
      glBegin(GL_POINTS)
      global xc, yc, r
      x, y = 0, r
      p = 1 - r
      plot_symmetric_points(x, y)
      while x < y:
            x += 1
            if p < 0:
                  p += 2 * x + 1
            else:
                  y -= 1
                  p += 2 * (x - y) + 1
            plot_symmetric_points(x, y)
      glEnd()
      glFlush()
```

```
def polar_circle():
      glBegin(GL_POINTS)
      theta = 0.0
      while theta <= 6.28:
            x = float(r) * math.cos(theta)
            y = float(r) * math.sin(theta)
            glVertex2f(x / SCALE, y / SCALE)
            theta += 0.001
      glEnd()
      glFlush()


def nonpolar_circle():
      global xc, yc, r
      glBegin(GL_POINTS)
      x, y = xc, r
      plot_symmetric_points(x - xc, y)
      while x < (xc + r):
            x += 1
            y = math.sqrt(float((r * r) - ((x - xc) * (x - xc))))
            plot_symmetric_points(x - xc, y)
      glEnd()
      glFlush()


def plot_symmetric_points(x, y):
      global xc, yc
      glVertex2f((xc + x) / SCALE, (yc + y) / SCALE)
      glVertex2f((xc + x) / SCALE, (yc - y) / SCALE)
      glVertex2f((xc - x) / SCALE, (yc + y) / SCALE)
      glVertex2f((xc - x) / SCALE, (yc - y) / SCALE)
      glVertex2f((xc + y) / SCALE, (yc + x) / SCALE)
      glVertex2f((xc + y) / SCALE, (yc - x) / SCALE)
      glVertex2f((xc - y) / SCALE, (yc + x) / SCALE)
      glVertex2f((xc - y) / SCALE, (yc - x) / SCALE)
def no_circle():
      pass


def main():
```

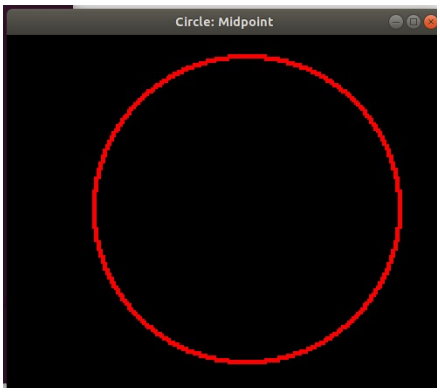
```
	glutInit(sys.argv)
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB)
	glutInitWindowSize(WINDOW_SIZE, WINDOW_SIZE)
	glutInitWindowPosition(50, 50)
	global xc, yc, r
	xc = int(input("Enter x coordinate of the centre "))
	yc = int(input("Enter y coordinate of the centre "))
	r = int(input("Enter length of radius "))
	choice = int(input("Enter the required choice:\n 1. Midpoint circle algorithm\n 2. Polar circle generation algorithm\n 3. Non-Polar circle generation algorithm\nEnter Choice:"))

	if choice == 1:
		glutCreateWindow("Circle: Midpoint")
		init_display()
		glutDisplayFunc(midpoint_circle)
	elif choice == 2:
		glutCreateWindow("Circle: Polar")
		init_display()
		glutDisplayFunc(polar_circle)

	elif choice == 3:
		glutCreateWindow("Circle: Non Polar")
		init_display()
		glutDisplayFunc(nonpolar_circle)
	else:
		glutCreateWindow("Error")
		init_display()
		glutDisplayFunc(no_circle)
		print("Invalid option chosen!")
	glutMainLoop()
main()
```

**Output:**

**1.**

```
Enter x coordinate of the centre 10
Enter y coordinate of the centre 20
```

```
Enter length of radius 70
 1. Midpoint circle algorithm
 2. Polar circle generation algorithm
 3. Non-Polar circle generation algorithm
Enter Choice:1
```




**2.**

```
Enter x coordinate of the centre 10
Enter y coordinate of the centre 20
Enter length of radius 80
Enter the required choice:
 1. Midpoint circle algorithm
 2. Polar circle generation algorithm
 3. Non-Polar circle generation algorithm
Enter Choice:2
```


**3.**

```
Enter x coordinate of the centre 10
Enter y coordinate of the centre 20
```

```
Enter length of radius 60

Enter the required choice:

 1. Midpoint circle algorithm

 2. Polar circle generation algorithm

 3. Non-Polar circle generation algorithm

Enter Choice:3
```