EXPERIMENT NO 3 **DATE : 02/09/2021**

AIM

To write a menu driven program to draw a circle using A) Mid point circle drawing algorithm B) Polar circle generation algorithm C) Non-Polar circle generation algorithm

ALGORITHM

```
Step 1: Start
Step 2: Initialize the glut library toolkit
Step 3: Initialize window size and position
Step 4: Read x1, x2, y1, y2
Step 5: Create redrawing function midpoint_circle()
def midpoint_circle():
  x, y = 0, r
  p = 1 - r
  Plot (xc + x,yc + y) and its seven corresponding symmetric points
  while x < y:
    x += 1
    if p < 0:
      p += 2 * x + 1
    else:
      y -= 1
      p += 2 * (x - y) + 1
    Plot (xc + x,yc + y) and its seven corresponding symmetric points
Step 6: Create redrawing function polar_circle()
def polar_circle():
  theta = 0.0
  while theta <= 6.28:
    x = float(r) * math.cos(theta)
    y = float(r) * math.sin(theta)
    Set pixel(x, y)
    theta += 0.001
Step 7: Create redrawing function nonpolar_circle()
def nonpolar_circle():
  x, y = xc, r
```

```
    Plot (x - xc, y) and its seven symmetric points
    while x < (xc + r):
        x += 1
        y = math.sqrt(float((r * r) - ((x - xc) * (x - xc))))
        Plot (x - xc, y) and its seven symmetric points
Step 8: Stop
```

PROGRAM

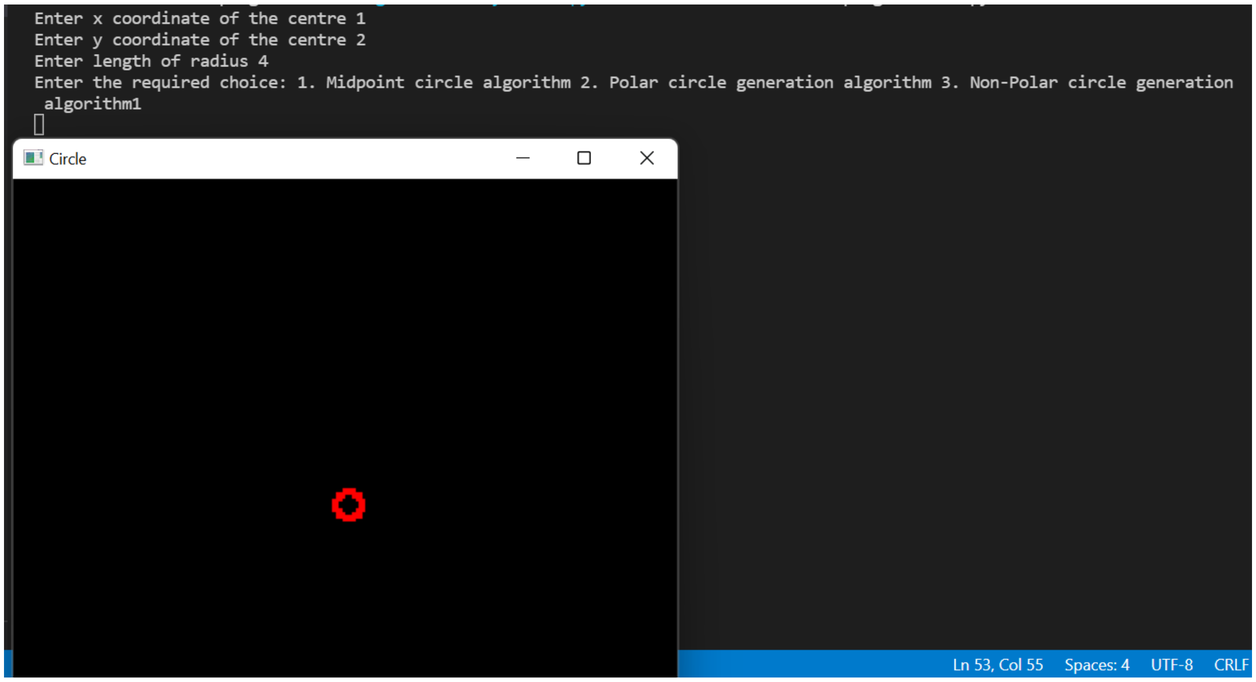
```
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
import sys
import math

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: 1. Midpoint circle algorithm 2. Polar circle generation
algorithm 3. Non-Polar circle generation algorithm"))

    glutCreateWindow("Circle")
    init_display()
    if choice == 1:
        glutDisplayFunc(midpoint_circle)
    elif choice == 2:
        glutDisplayFunc(polar_circle)
    elif choice == 3:
        glutDisplayFunc(nonpolar_circle)
    else:
        glutDisplayFunc(no_circle)
        print("Invalid option chosen!")

    glutMainLoop()

main()
```

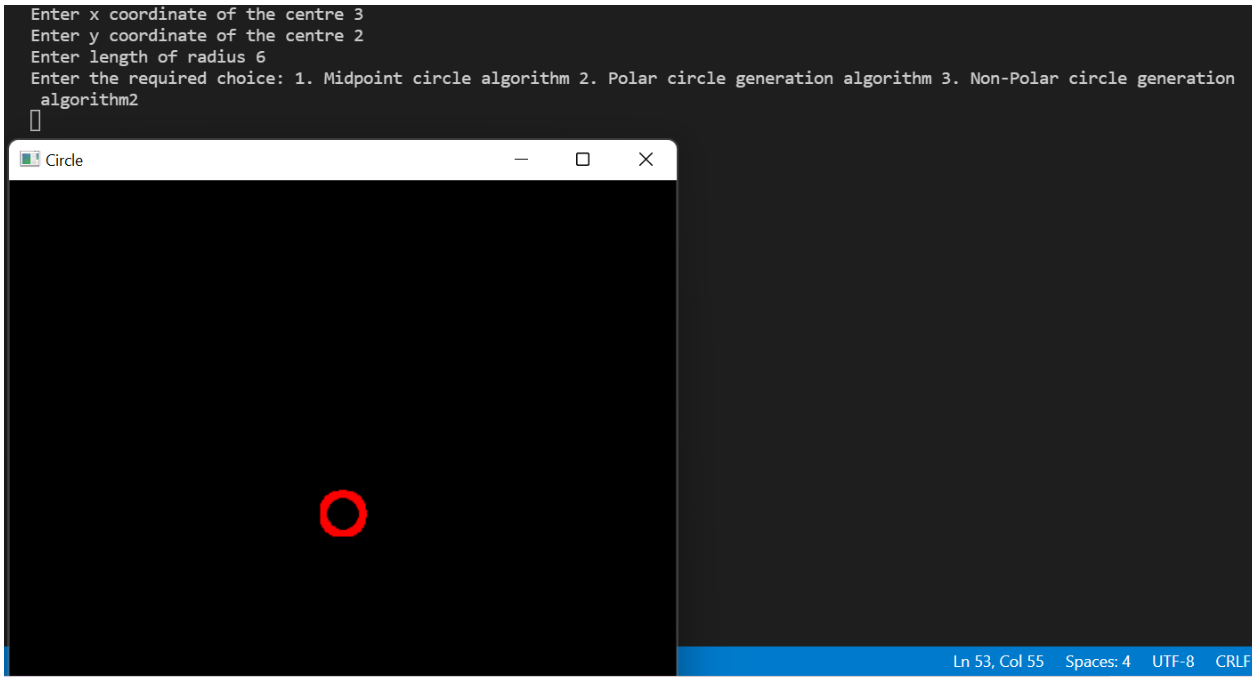
RESULT

The required program has been created.

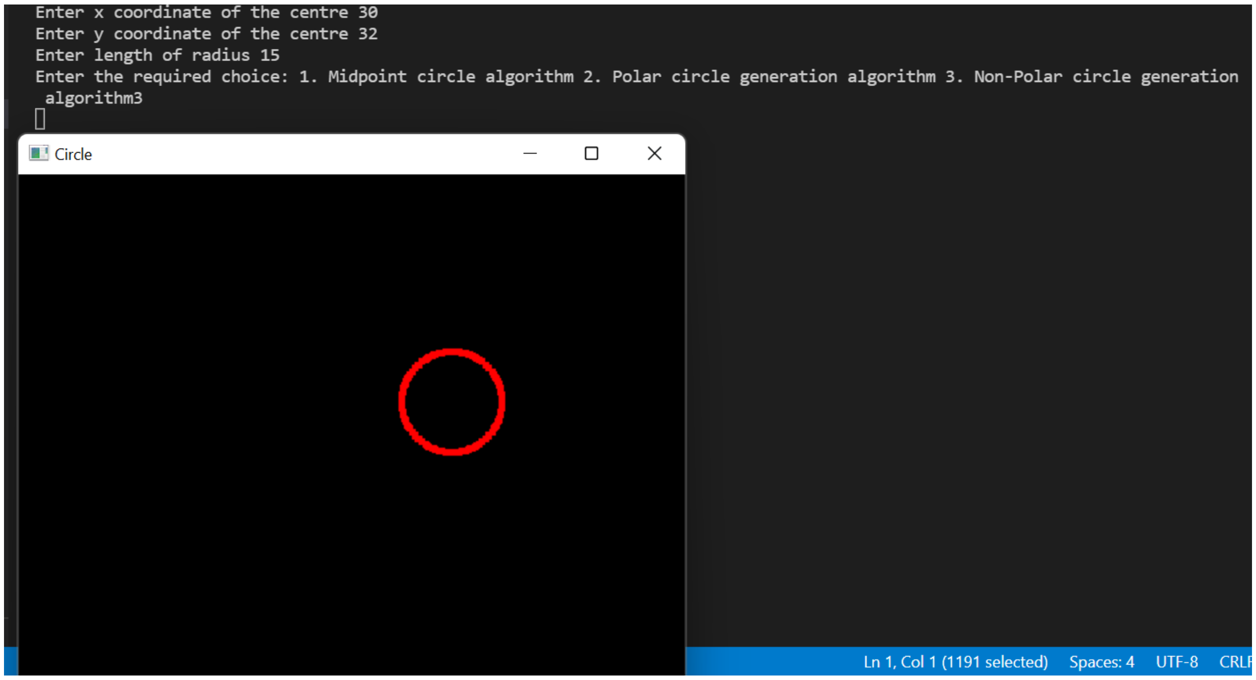
INPUT/OUTPUT



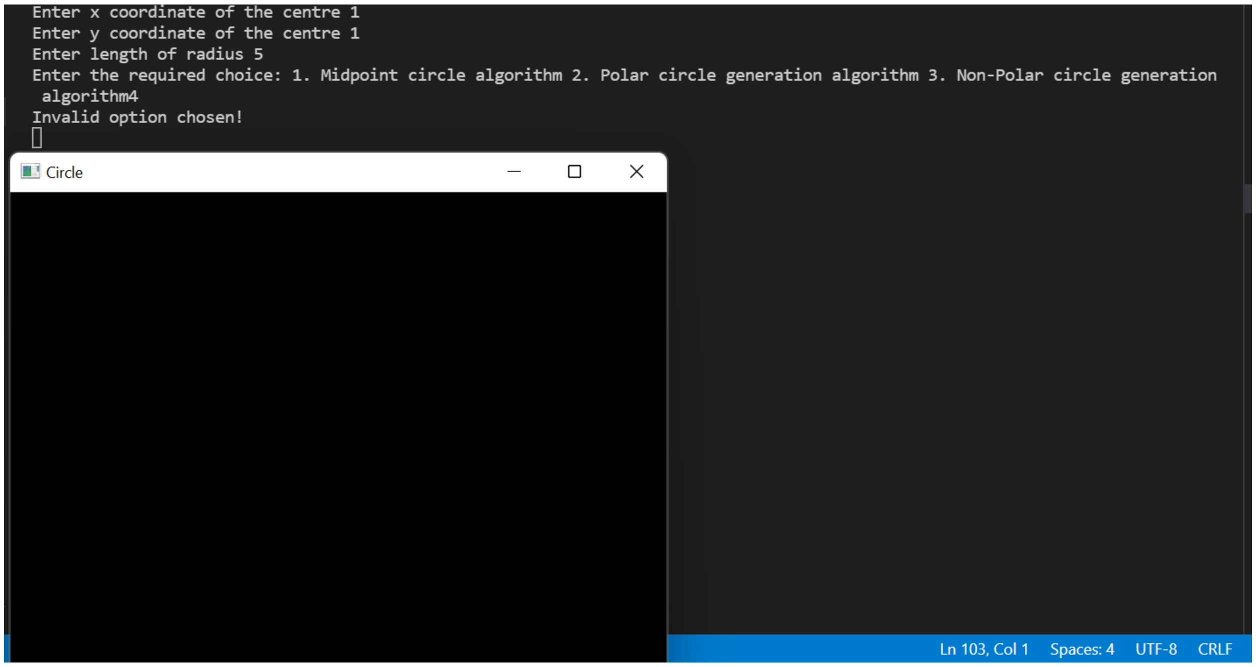
```
Enter x coordinate of the centre 1
Enter y coordinate of the centre 2
Enter length of radius 4
Enter the required choice: 1. Midpoint circle algorithm 2. Polar circle generation algorithm 3. Non-Polar circle generation
 algorithm1
```



```
Enter x coordinate of the centre 3
Enter y coordinate of the centre 2
Enter length of radius 6
Enter the required choice: 1. Midpoint circle algorithm 2. Polar circle generation algorithm 3. Non-Polar circle generation
 algorithm2
```



```
Enter x coordinate of the centre 30
Enter y coordinate of the centre 32
Enter length of radius 15
Enter the required choice: 1. Midpoint circle algorithm 2. Polar circle generation algorithm 3. Non-Polar circle generation
 algorithm3
```



```
Enter x coordinate of the centre 1
Enter y coordinate of the centre 1
Enter length of radius 5
Enter the required choice: 1. Midpoint circle algorithm 2. Polar circle generation algorithm 3. Non-Polar circle generation
 algorithm4
Invalid option chosen!
```