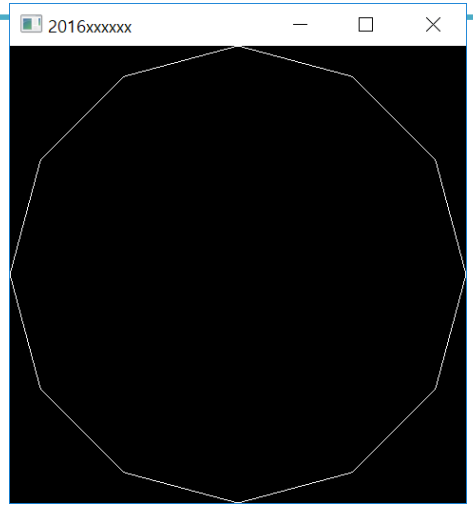
Daily Assignment 4

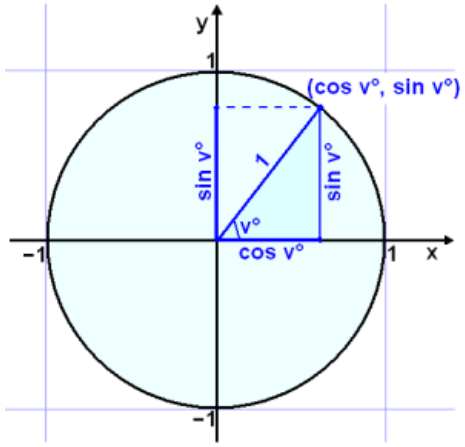


- Write down a Python program to draw a regular 12-sided polygon (dodecagon, 정12각형).
  - Use np.linspace() (or np.arrange()), np.cos(), np.sin() to compute the positions of vertices
  - Do not hardcode the position of each vertex

- **Set the window title to your student number.**
- Set the window size to (480,480).

- The 12 vertices should be specified counterclockwise starting from the vertex on the x-axis.

(Hint)



Daily Assignment 4

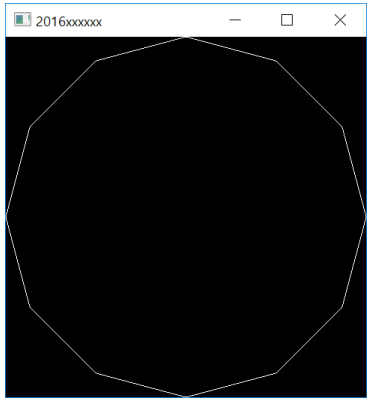
Key	Primitive Type
1	GL_POINTS
2	GL_LINES
3	GL_LINE_STRIP
4	GL_LINE_LOOP
5	GL_TRIANGLES
6	GL_TRIANGLE_STRIP
7	GL_TRIANGLE_FAN
8	GL_QUADS
9	GL_QUAD_STRIP
10	GL_POLYGON

- If the keys 1, 2, 3, … 9, 0 are entered, the primitive type should be changed.
  - Hint: Use a global variable to store the primitive type

- Global variables in Python
  - https://www.geeksforgeeks.org/global-local-variables-python/ for more information

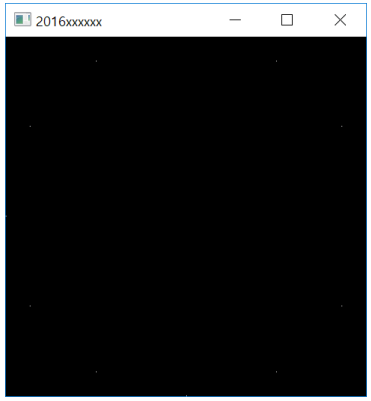
```
# This function modifies global variable 's'
def f():
    global s
    print s
    s = "Look for Geeksforgeeks Python Section"
    print s

# Global Scope
s = "Python is great!"
f()
print s
```



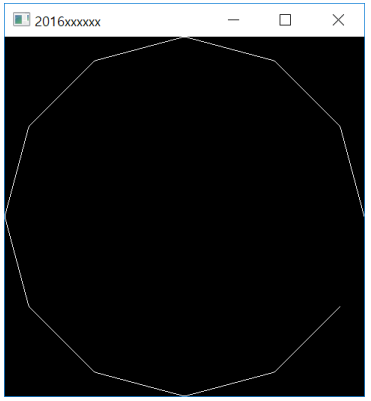
When the program starts

Press '1'

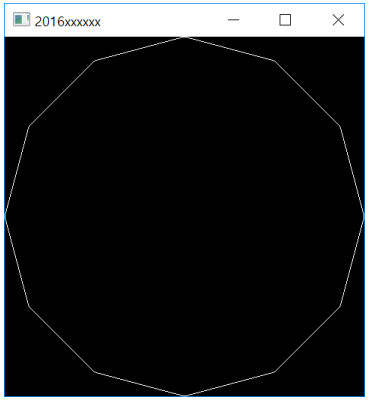


Press '2'

Press '3'



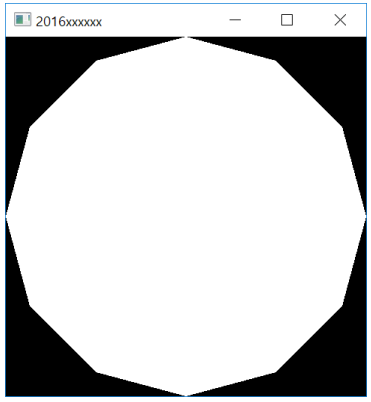
Press '4'



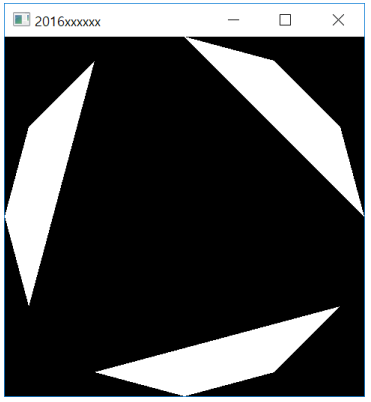
Press '5'

Press '6'

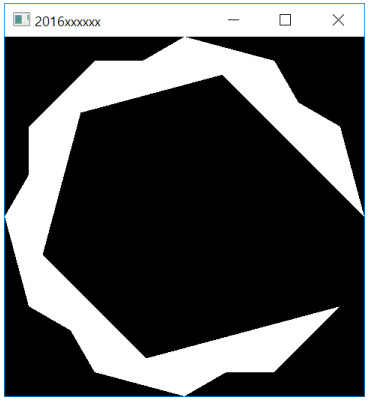
Press '7'



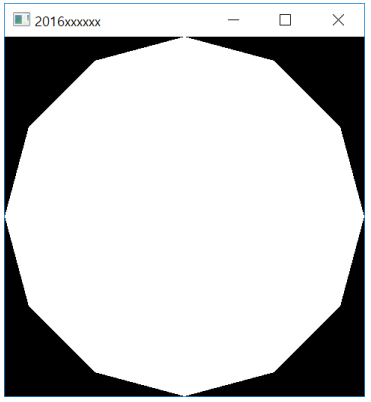
Press '8'



Press '9'



Press '0'



```python
import glfw
from OpenGL.GL import *
import numpy as np

gPrimitiveType = GL_LINE_LOOP

def render():
    global gPrimitiveType
    glClear(GL_COLOR_BUFFER_BIT)
    glLoadIdentity()

    glBegin(gPrimitiveType)
    glColor3ub(255, 255, 255)
    # [0*(2pi/12), 1*(2pi/12), ... , 11*(2pi/12)]
    for th in np.linspace(0, 2*np.pi, 12+1)[:-1]:
        x = np.cos(th)
        y = np.sin(th)
        glVertex2fv((x, y))
    glEnd()
```

```python
def key_callback(window, key, scancode,
action, mods):
    global gPrimitiveType
    if action==glfw.PRESS:
        if key==glfw.KEY_1:
            gPrimitiveType = GL_POINTS
        elif key==glfw.KEY_2:
            gPrimitiveType = GL_LINES
        elif key==glfw.KEY_3:
            gPrimitiveType = GL_LINE_STRIP
        elif key==glfw.KEY_4:
            gPrimitiveType = GL_LINE_LOOP
        elif key==glfw.KEY_5:
            gPrimitiveType = GL_TRIANGLES
        elif key==glfw.KEY_6:
            gPrimitiveType = GL_TRIANGLE_STRIP
        elif key==glfw.KEY_7:
            gPrimitiveType = GL_TRIANGLE_FAN
        elif key==glfw.KEY_8:
            gPrimitiveType = GL_QUADS
        elif key==glfw.KEY_9:
            gPrimitiveType = GL_QUAD_STRIP
        elif key==glfw.KEY_0:
            gPrimitiveType = GL_POLYGON
```

```
def main():
    # Initialize the library
    if not glfw.init():
        return
    # Create a windowed mode window and its OpenGL context
    window = glfw.create_window(480, 480, "2016xxxxxx", None, None)
    if not window:
        glfw.terminate()
        return

    glfw.set_key_callback(window, key_callback)

    # Make the window's context current
    glfw.make_context_current(window)

    # Loop until the user closes the window
    while not glfw.window_should_close(window):
        # Poll for and process events
        glfw.poll_events()

        # Render here, e.g. using pyOpenGL
        render()

        # Swap front and back buffers
        glfw.swap_buffers(window)

    glfw.terminate()

if __name__ == "__main__":
    main()
```