Daily Assignment 7

- Write down a Python program to..
- Draw a triangle using the render() function in 21 page of today's lecture slides (DO NOT modify it!)
  - Use 4x4 matrices for transformation!
- If you **press or repeat** a key, the triangle should be transformed as shown in the Table:
  - And for camera rotation, increase/decrease camAng parameter that passed to render()
- Transformations should be **accumulated**
  - You'll need two global variables to store current accumulated transformation and current camera angle
- **Set the window title to your student number.**
- Set the window size to (480,480).

Key	Transformation
Q	Translate by -0.1 in x direction **w.r.t global coordinate**
E	Translate by 0.1 in x direction **w.r.t global coordinate**
A	Rotate about y axis by 10° clockwise **w.r.t local coordinate**
D	Rotate about y axis by 10° counterclockwise **w.r.t local coordinate**
W	Rotate about x axis by 10° clockwise **w.r.t local coordinate**
S	Rotate about x axis by 10° counterclockwise **w.r.t local coordinate**
1	Rotate camera 10° clockwise
3	Rotate camera 10° counterclockwise

```
#...
gCamAng = 0
gComposedM = np.identity(4)

def render(T, camAng):
    #...

def key_callback(window, key, scancode, action,
mods):
    global gCamAng, gComposedM
    if action==glfw.PRESS or action==glfw.REPEAT:

        if key==glfw.KEY_1:
            gCamAng += np.radians(-10)
        elif key==glfw.KEY_3:
            gCamAng += np.radians(10)

        elif key==glfw.KEY_Q:
            M = np.identity(4)
            M[:3,3] = [-.1,0,0]
            gComposedM = M @ gComposedM

        elif key==glfw.KEY_E:
            M = np.identity(4)
            M[:3,3] = [.1,0,0]
            gComposedM = M @ gComposedM
```

```
    elif key==glfw.KEY_A:
        M = np.identity(4)
        th = np.radians(-10)
        M[:3,:3] = [[np.cos(th), 0, np.sin(th)],
                    [0,1,0],
                    [-np.sin(th), 0, np.cos(th)]]
        gComposedM = gComposedM @ M

    elif key==glfw.KEY_D:
        M = np.identity(4)
        th = np.radians(10)
        M[:3,:3] = [[np.cos(th), 0, np.sin(th)],
                    [0,1,0],
                    [-np.sin(th), 0, np.cos(th)]]
        gComposedM = gComposedM @ M

    elif key==glfw.KEY_W:
        M = np.identity(4)
        th = np.radians(-10)
        M[:3,:3] = [[1,0,0],
                    [0, np.cos(th), -np.sin(th)],
                    [0, np.sin(th), np.cos(th)]]
        gComposedM = gComposedM @ M

    elif key==glfw.KEY_S:
        M = np.identity(4)
        th = np.radians(10)
        M[:3,:3] = [[1,0,0],
                    [0, np.cos(th), -np.sin(th)],
                    [0, np.sin(th), np.cos(th)]]
        gComposedM = gComposedM @ M
```

```
def main():
    global gCamAng, gComposedM
    if not glfw.init():
        return
    window = glfw.create_window(480,480,"2016xxxxxx", None,None)
    if not window:
        glfw.terminate()
        return
    glfw.make_context_current(window)
    glfw.set_key_callback(window, key_callback)

    while not glfw.window_should_close(window):
        glfw.poll_events()

        render(gComposedM, gCamAng)

        glfw.swap_buffers(window)

    glfw.terminate()

if   name    == "__main__":
    main()<
```