Daily Assignment 17

- In Gouraud shading, one vertex has only one normal. This makes using glDrawElements() easier.

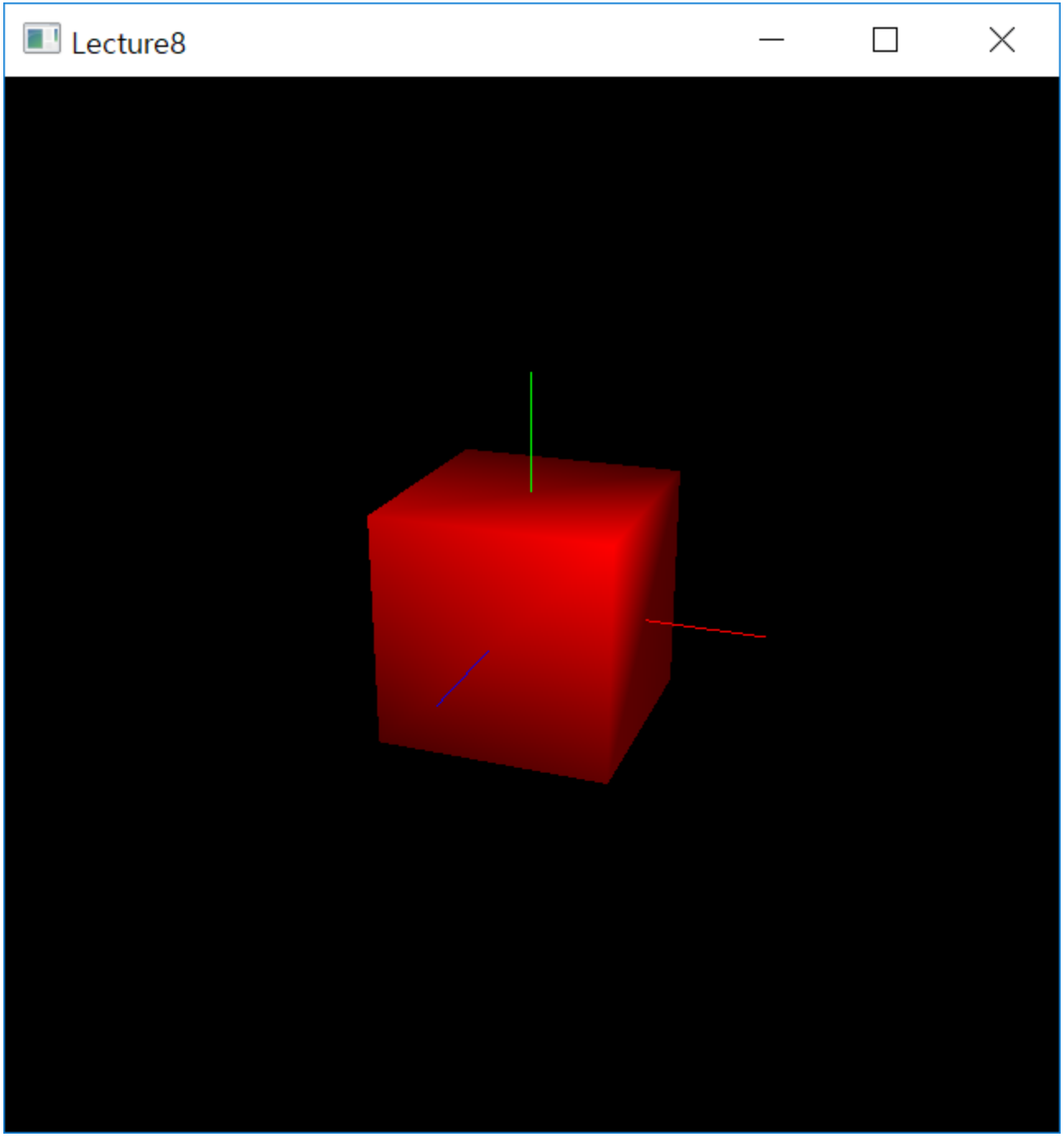
- Start from code in today's lecture slides, draw a smooth-shaded cube using code segments in next pages.
  - **What you have to do is to fill the blanks in createVertexAndIndexArrayIndexed()**
  - Fill proper "smooth" normal vectors. (You do not actually need to compute the average of face normals)
  - You may need normalized() function (in the next page) to make an arbitrary length normal vector to a unit normal vector
  - Your render() should call **drawUnitCube_glDrawElements()** (in the next page) to draw a cube

- **Change light position & color and material color as you want.**

```
def createVertexAndIndexArrayIndexed():
    varr = np.array([
            _____________________________,
            [ 0.5, 0.5,-0.5],
            _____________________________,
            [-0.5, 0.5,-0.5],
            _____________________________,
            [-0.5, 0.5, 0.5],
            _____________________________,
            [ 0.5, 0.5, 0.5],
            _____________________________,
            [ 0.5,-0.5, 0.5],
            _____________________________,
            [-0.5,-0.5, 0.5],
            _____________________________,
            [-0.5,-0.5,-0.5],
            _____________________________,
            [ 0.5,-0.5,-0.5],
            ], 'float32')
    iarr = np.array([
            [0,1,2],
            [0,2,3],
            [4,5,6],
            [4,6,7],
            [3,2,5],
            [3,5,4],
            [7,6,1],
            [7,1,0],
            [2,1,6],
            [2,6,5],
            [0,3,4],
            [0,4,7],
            ])
    return varr, iarr
```

```
def l2norm(v):
    return np.sqrt(np.dot(v, v))

def normalized(v):
    l = l2norm(v)
    return 1/l * np.array(v)

def drawUnitCube_glDrawElements():
    global gVertexArrayIndexed, gIndexArray
    varr = gVertexArrayIndexed
    iarr = gIndexArray
    glEnableClientState(GL_VERTEX_ARRAY)
    glEnableClientState(GL_NORMAL_ARRAY)
    glNormalPointer(GL_FLOAT, 6*varr.itemsize,
varr)
    glVertexPointer(3, GL_FLOAT,
6*varr.itemsize,
ctypes.c_void_p(varr.ctypes.data +
3*varr.itemsize))
    glDrawElements(GL_TRIANGLES, iarr.size,
GL_UNSIGNED_INT, iarr)
```

```
gVertexArrayIndexed = None
gIndexArray = None

def main():
    global gVertexArraySeparate
    global gVertexArrayIndexed, gIndexArray

    if not glfw.init():
        return
    window = glfw.create_window(640,640,'Lecture8', None,None)
    if not window:
        glfw.terminate()
        return
    glfw.make_context_current(window)
    glfw.set_key_callback(window, key_callback)
    glfw.swap_interval(1)

    gVertexArraySeparate = createVertexArraySeparate()
    gVertexArrayIndexed, gIndexArray = createVertexAndIndexArrayIndexed()

    count = 0
    while not glfw.window_should_close(window):
        glfw.poll_events()
        ang = count % 360
        render(ang)
        count += 1
        glfw.swap_buffers(window)

    glfw.terminate()
```


Lecture8

How to Submit

- What you have to submit:
  - Only **one** .py file: ***main.py***
- Write down all your code to ***main.py***
- `> py -3 main.py` or `$ python3 main.py` should show your glfw window.

How to Submit

- Submit your assignment **only through the Assignment (과제) menu of the lecture home** at portal.hanyang.ac.kr.

- **Recommended due date: Today's lecture end time**

- (Hard due date: 23:59 Today)