Daily Assignment 10

- Write your own **myLookAt()** function (of the following form) that behaves exactly same as gluLookAt()

```
def myLookAt(eye, at, up): # eye, at, up are 1D numpy array of length 3
```

- Start from today's gluLookAt() practice code, add your myLookAt() and call it instead of gluLookAt()

- Hint:
- Everything you need to write code is on page 29-32, 38
- l2 norm of **v** : ||**v**|| = np.sqrt(np.dot(**v**, **v**))
- **a** x **b** (cross product) : np.cross(**a**, **b**)
- **a** · **b** (inner product) : np.dot(**a**, **b**)
- Use glMultMatrixf() to multiply your viewing matrix to the current transformation matrix

```python
def myLookAt(eye, at, up):
    w = (eye-at) / np.sqrt(np.dot(eye-at, eye-at))
    u = np.cross(up, w) / np.sqrt(np.dot(np.cross(up, w), np.cross(up, w)))
    v = np.cross(w, u)
    Mv = np.array([[u[0], u[1], u[2], -np.dot(u, eye)],
                   [v[0], v[1], v[2], -np.dot(v, eye)],
                   [w[0], w[1], w[2], -np.dot(w, eye)],
                   [0,0,0,1]])
    glMultMatrixf(Mv.T)

def render(camAng):
    # ...

    # gluLookAt(1*np.sin(camAng),1,1*np.cos(camAng), 0,0,0, 0,1,0)
    myLookAt(np.array([1*np.sin(camAng),1,1*np.cos(camAng)]), np.array([0,0,0]),
np.array([0,1,0]))
```