

Computer Graphics, Lab Assignment 4

Handed out: March 23, 2021

Due: 23:59, March 23, 2021 (NO SCORE for late submissions!)

- Only accept answers submitted via git push to this course project for you at <https://hconnect.hanyang.ac.kr> (<Year>_<Course no.>_<Class code>/<Year>_<Course no.>_<Student ID>.git).
- Place your files under the directory structure <Assignment name>/<Problem no.>/<your file> just like the following example.

```
+ 2021_ITE0000_2019000001
+ LabAssignment4/
+ 1/
+   - 1.py
+ 2/
+   - 2.py
+ 3/
+   - 3.py
```

- The submission time is determined not when the commit is made but when the git push is made.

1. Write down a Python program to draw a transformed triangle in a 2D space.
 - A. Set the window title to **your student ID** and the window size to (480,480).
 - B. Complete the render() function below to draw a triangle in the manner described in C.
 - i. **You have to use OpenGL transformation functions. Do not use numpy matrix multiplication for composing transformations.**

```

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

    # draw coordinates
    glBegin(GL_LINES)
    glColor3ub(255, 0, 0)
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([1.,0.]))
    glColor3ub(0, 255, 0)
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([0.,1.]))
    glEnd()

    glColor3ub(255, 255, 255)

    #####
    # implement here
    #####

    drawTriangle()

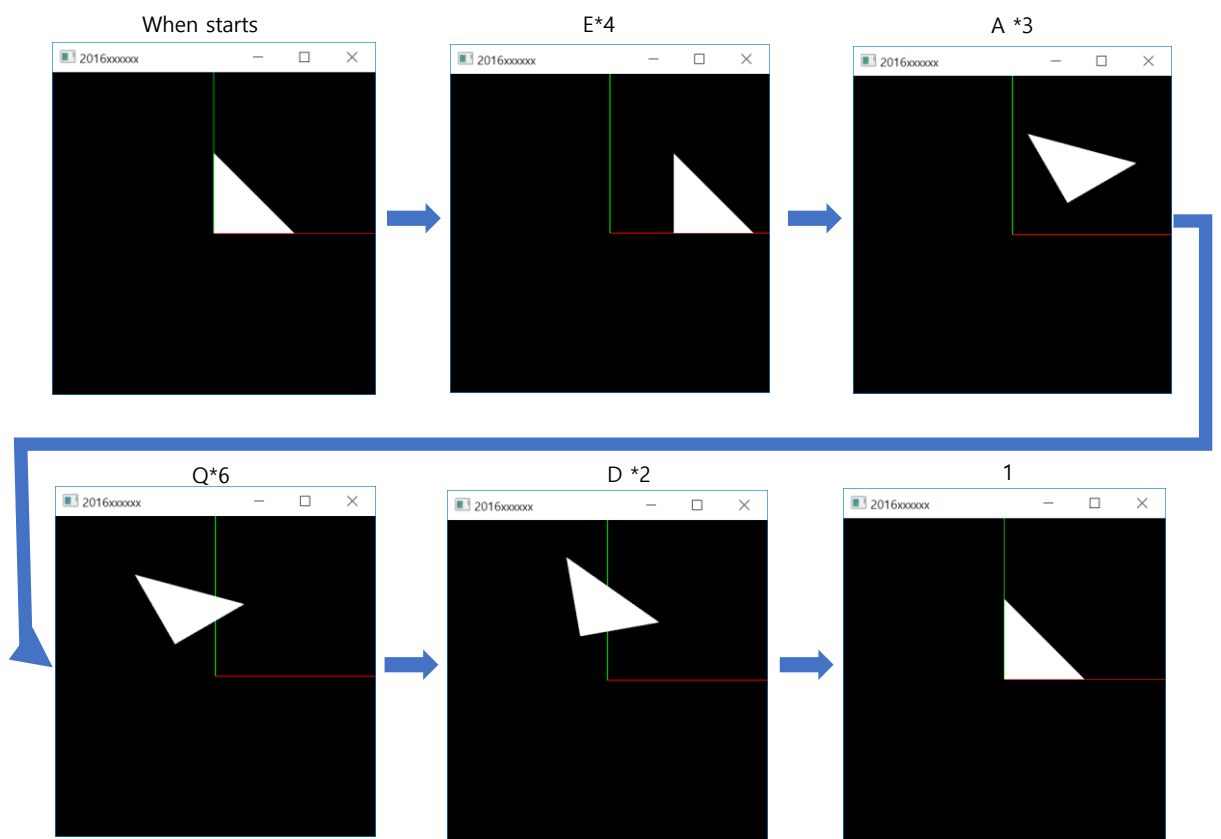
def drawTriangle():
    glBegin(GL_TRIANGLES)
    glVertex2fv(np.array([0.,.5]))
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([.5,0.]))
    glEnd()

```

- C. If you press or repeat a key, the triangle should be transformed as shown in the Table:

Key	Transformation
Q	Translate by -0.1 in x direction
E	Translate by 0.1 in x direction
A	Rotate by 10 degrees counterclockwise
D	Rotate by 10 degrees clockwise
1	Reset the triangle with identity matrix

- D. Transformations should be accumulated (composed with previous one) unless you press '1'.
- i. You may need a global variable (like a python list object) to store key inputs.
- E. Files to submit: A Python source file (Name the file whatever you want (in English). Extension should be .py)
- F. Expected result:



2. Write down a Python program to draw rotating point $p1=(0.5, 0)$, $p2=(0, 0.5)$ and vector $v1=(0.5, 0)$, $v2=(0, 0.5)$ in a 2D space.
 - A. Set the window title to **your student ID** and the window size to (480,480).
 - B. Use the following `render()` and fill "# your implementation" parts to render $p1, p2$ and $v1, v2$.
 - i. Hint: Render the vector $v1, v2$ as a line segment starting from the origin (0,0).
 - ii. Hint2: You need different translation matrix for $p1$ and $p2$ to render them correctly.

```

def render(th):
    glClear(GL_COLOR_BUFFER_BIT)
    glLoadIdentity()

    # draw coordinate
    glBegin(GL_LINES)
    glColor3ub(255, 0, 0)
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([1.,0.]))
    glColor3ub(0, 255, 0)
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([0.,1.]))
    glEnd()

    glColor3ub(255, 255, 255)

    # calculate matrix M1, M2 using th
    # your implementation

    # draw point p
    glBegin(GL_POINTS)
    # your implementation
    glEnd()

    # draw vector v
    glBegin(GL_LINES)
    # your implementation
    glEnd()

```

- C. Expected result: Uploaded LabAssignment4-2.mp4
- i. Do not mind the initial angle.
- D. p_1, p_2 and v_1, v_2 should be $-t$ rad rotated when t seconds have elapsed since the program was executed.
- E. You need to somehow combine a rotation matrix and a translation matrix to produce the expected result.
- F. Files to submit: A Python source file (Name the file whatever you want (in English). Extension should be .py)