학번: **이름:**

문제 사이의 빈 공간에 답변을 작성할 것.

문제가 있는 각 페이지의 앞면에 답안을 작성할 것 (뒷면에 작성하지 말 것).

각 페이지마다 이름과 학번을 기입할 것.

1. (True/False 문제) Mesh 표현 방식 중 separate triangle 방식은 indexed triangle set 방식에 비해 메모리 공간 측면에서 더 효율적이다.

2. (True/False 문제) Phong illumination model에서는 cos() 함수의 n승을 이용해서 물체 표면의 highlight를 표현한다.

3. (x_i, y_i)와 같은 식으로 표기되는 2차원 공간 상의 점들의 집합으로 표현된 어떤 도형을 x축 방향으로 2배, y축 방향으로 1.5배 scaling하는 2 x 2 transformation 행렬은 무엇인가? (답안은 편집기의 수식(Equation)기능을 사용하지 말고 일반 텍스트로 입력할 것. 줄바꿈(new line)을 통해 행렬의 형태로 보이게만 입력하면 됨)

4. R은 4 x 4 rotation matrix, T는 4 x 4 translation matrix라고 할 때, 임의의 점 p를 global frame 기준으로 R만큼 회전한 다음 T만큼 평행이동 한 점 p'은 어떻게 표현될 수 있는가?

5. 아래 코드의 (a)부분에서의 OpenGL current transformation matrix는 무엇인가?

```
def render():
    global M1, M2; // 4x4 transformation matrices (numpy n-dim array)
    glClear(GL_COLOR_BUFFER_BIT)
    glLoadIdentity()
    glPushMatrix()
    glMultMatrixf(M1.T)
    glPushMatrix()
    glMultMatrixf(M2.T)
    drawSomething()
    glPopMatrix()
    // (a)
```

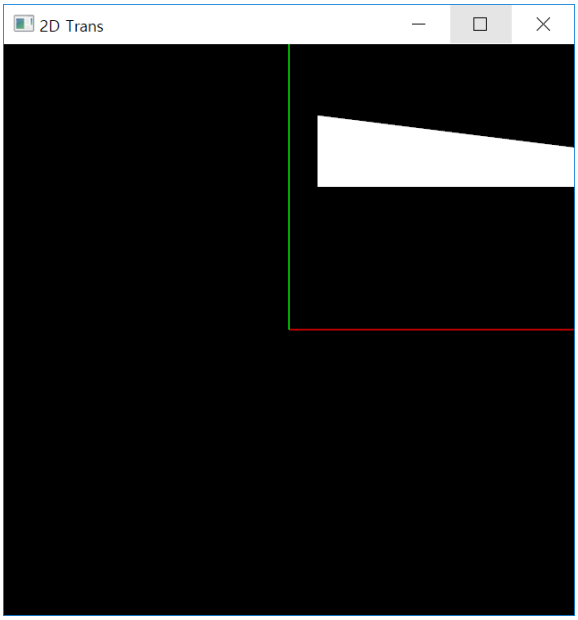
학번: **이름:**

6. Polygon 내부의 특정 위치의 색상값을 결정하기 위해 Phong shading은 ___(a)___을 interpolate하여 계산에 사용하고, Gourad shading은 ___(b)___를 interpolate하여 사용한다. 빈칸을 채우세요.

7. yaw-pitch-roll convention은 ZYX Euler angles에 해당하며, yaw가 z축, pitch가 y축, roll이 x축에 대한 회전각을 각각 의미한다. $R_x(\theta)$, $R_y(\theta)$, $R_z(\theta)$가 각각 x, y, z축에 대해 θ rad만큼 회전하는 rotation matrix를 의미한다고 할 때, 임의의 yaw, pitch, roll 각도가 각각 0.2, 0.5, 1.0 rad으로 주어진 경우 이 회전을 나타내는 rotation matrix를 이 문제에서 사용하고 있는 기호들을 이용하여 표현하세요.

8. modeling, viewing, projection, viewport transformation matrix 들을 각각 $\mathbf{M}_m, \mathbf{M}_v, \mathbf{M}_{pj}, \mathbf{M}_{vp}$, 이라고 하자. object space 상의 한 정점의 위치 $\mathbf{p}_o$ 를 screen space 상의 해당하는 정점의 위치 $\mathbf{p}_s$ 로 변환하는 수식을 쓰세요.

9. 어떤 (a)와 (b) 쌍이 아래 코드에 들어갔을 때 아래와 같은 이미지를 만들어 낼까?



학번: 이름:

```
def render(M, u):
    glClear(GL_COLOR_BUFFER_BIT)
    glLoadIdentity()
    glBegin(GL_TRIANGLES)
    glColor3ub(255, 255, 255)
    glVertex2fv(M @ np.array([0.0,0.5]) + u)
    glVertex2fv(M @ np.array([0.0,0.0]) + u)
    glVertex2fv(M @ np.array([0.5,0.0]) + u)
    glEnd()
def main():
    # ...
    while not glfw.window_should_close(window):
        # ...
        M = np.array(__(a)__)
        u = np.array(__(b)__)
        render(M, u)
```

1)

1)
```
(a)
[[.5,0]
 [0,4]]
(b)
[.5,.1]
```

2)
```
(a)
[[2,0]
 [0,2]]
(b)
[.5,.1]
```

3)
```
(a)
[[4,0]
 [0,.5]]
(b)
[.1,.5]
```

4)
```
(a)
[[4,0]
 [0,.5]]
(b)
[.5,.5]
```

5)
```
(a)
[[4,1]
 [0,.5]]
(b)
[.1,.5]
```

10. 어떤 오브젝트에 global frame에 대해 아래와 같은 순서로 변환을 적용하고 싶을 때,

학번: **이름:**

1) 첫번째로 translated by (1,0,2)

2) 두번째로 scaled by (-2, 2, 4)

3) 세번째로 rotated π/2 about y axis

glLoadIdentity() 호출과 오브젝트를 그리는 코드 사이에 들어갈 수 있는 알맞은 코드 조각들을 **모두** 고르세요 (코드에 이미 import numpy as np가 되어 있다고 가정).

1)
```
glTranslatef(1, 0, 2)
glScalef(-2, 2, 4)
glRotatef(np.pi/2, 0, 1, 0)
```

2)
```
glRotatef(90, 0, 1, 0)
glScalef(-2, 2, 4)
glTranslatef(1, 0, 2)
```

3)
```
glTranslatef(1, 0, 2)
glScalef(-2, 2, 4)
glRotatef(90, 1, 0, 0)
```

4)
```
glRotatef(np.pi/2, 0, 1, 0)
glScalef(-2, 2, 4)
glTranslatef(1, 0, 2)
```

5)
```
glRotatef(90, 0, 1, 0)
glScalef(-2, 2, 4)
T = np.identity(4)
T[:3,3] = [1, 0, 2]
glMultiMatrixf(T.T)
```

11. 정점 A는 회색 box object의 정점들 중 하나이다. Box object의 local frame에 대해 표현된 A의 위치($\mathbf{A}^{\{l\}}$) 가 $(1,1,1)^T$ 이며 modeling transformation matrix가 다음과 같다고 할

학번: **이름:**

때:

```
[[1,0,0,1],
 [0,1,0,2],
 [0,0,1,3],
 [0,0,0,1]]
```

정점 A의 global frame에 대한 위치 ($\mathbf{A}^{\{g\}}$)를 계산해서 적으세요.

Object space World space

12. 어떤 object 상의 vertex v의 위치를 p라고 하자. object의 local frame이 global frame이 일치하는 초기 상태에서부터 시작하여, object에 아래의 순서로 object local frame에 대해 transformation을 적용한다면, 아래 transformation들이 모두 수행된 후 vertex A의 위치는 global frame에 대해 어떻게 표현되는가?

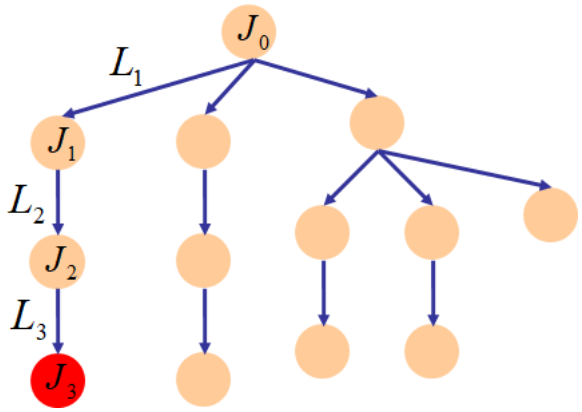
    첫번째로, transformed by 4x4 matrix A,

    두번째로, transformed by 4x4 matrix B,

    세번째로, transformed by 4x4 matrix C

13. 아래 그림은 어떤 articulated body model의 hierarchical structure를 나타낸다. 여기서 Jn은 joint transformation (pure rotation)을 나타내며 Ln은 link transformation (pure translation)을 나타낸다. 이 때 joint 3의 위치를 global frame에 대해 표현하는 수식을 적으세요. 그림에 표시된 transformation들과 원점(origin point) **O**을 이용해서 답안을 작성할 것. (joint 3의 joint transformation이 J3 이며, joint 3는 J3 frame의 원점에 위치한다고 가정한다)

학번: 이름:



14. 아래 코드에서 box4를 그릴 때 적용되는 transformation matrix는 무엇일까요? (참고로 아래의 glTranslate(T)와 glRotate(R) 같은 식의 함수 호출 방식은 실제 pyopengl의 용법과 다르지만, 각각 T에 해당하는 translation matrix를 만들어 current transformation matrix의 오른쪽에 곱하는 것, R에 해당하는 rotation matrix를 만들어 current transformation matrix의 오른쪽에 곱하는 것을 의미한다고 가정)

```
glLoadIdentity()
glPushMatrix()
glTranslate(T1)
Draw box1
glPushMatrix()
glRotate(R1)
Draw box2
glPushMatrix()
glRotate(R2)
Draw box3
glPopMatrix()
glPopMatrix()
glPushMatrix()
glRotate(R3)
Draw box4
glPopMatrix()
glPopMatrix()
```

학번: **이름:**

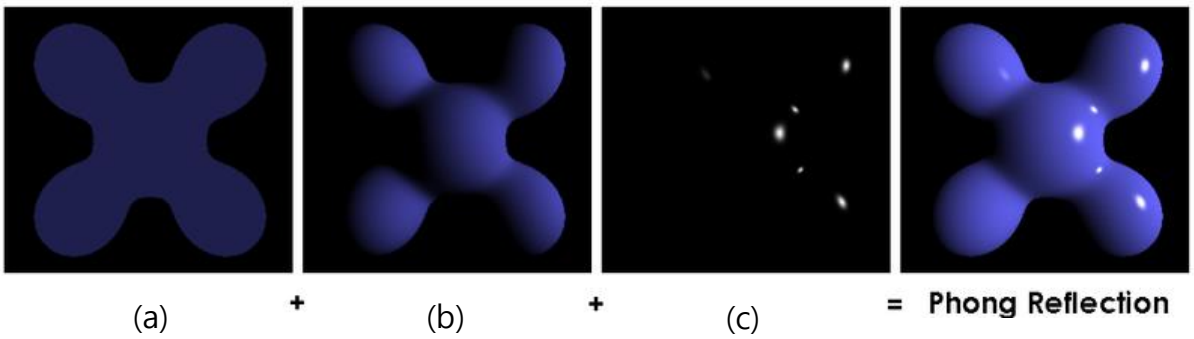
15. 다음 중 3차원 회전을 모두 표현할 수 없어 실제로 사용될 수 없는 Euler angles의 조합을 모두 선택하세요.

    1) XYZ
    2) ZXZ
    3) ZXX
    4) YZX
    5) YYX
    6) YXY

16. 자유도 (degrees of freedom)에 대한 다음 빈 칸을 채우세요.

    1) 하나의 평면 위에서의 translation은 __(a)__ 자유도 운동이다.
    2) 3차원 공간에서의 rotation은 __(b)__ 자유도 운동이다.
    3) 3차원 공간에서의 rigid motion은 __(c)__ 자유도 운동이다.

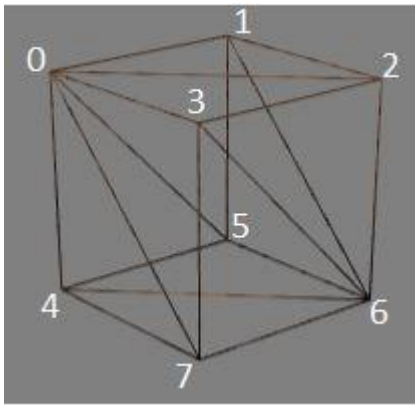
17. 아래 그림은 Phong illumination model의 각 component와 이들을 모두 합한 결과를 이미지로 나타낸 것이다. (a), (b), (c)에 들어갈 Phong illumination model의 각 component의 이름을 적으세요.



18. 3개의 data point x(0)=1, x(1)=-2, and x(2)=3을 지나는 가장 낮은 차수의 다항식을 적으세요.

학번: 이름:

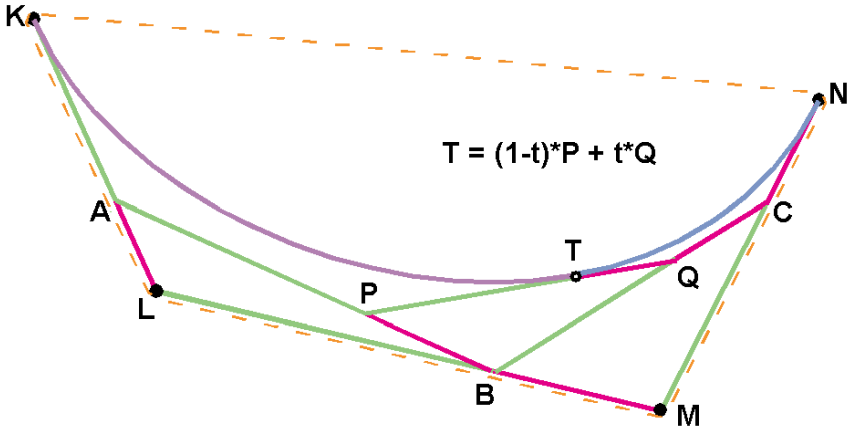
19. 아래는 triangle mesh로 표현된 cube object의 각 vertex index를 표시한 그림과, 이 object를 glDrawElements()함수로 그리기 위해 index array를 생성하는 코드이다. index array를 완성하기 위해 아래 빈칸 (a), (b)를 채우세요 ("앞면"을 나타내는 vertex winding order으로는 "시계반대방향"을 사용할 것).




```
iarr = np.array([
            (0,2,1),
            (0,3,2),
            (4,5,6),
            (4,6,7),
            (0,1,5),
             ___(a)___,
            (3,6,2),
            (3,7,6),
            (1,2,6),
            (1,6,5),
            (0,7,3),
            ___(b)___,
           ])
```

20. 0≤t≤1인 parameter t에 대해 두 개의 rotation matrix R1과 R2사이를 선형으로 보간하는 slerp(R1, R2, t)함수를 R1, R2, t를 이용해 표현하세요 (exp와 log 함수를 함께 사용해도 됨).

21. 아래 그림은 de Casteljau's Algorithm이 어떻게 Bezier curve를 만드는지 설명하는 그림이며, t 시점의 curve위의 한 점 T와 P, Q 사이의 관계가 기술되어 있다. 해당 점 T와 A, B, C 사이의 관계를 기술하는 식을 쓰세요.



학번: **이름:**

22. Affine transformation의 합성에 있어 변환이 적용되는 순서가 중요하다. 예를 들면, 어떤 물체를 rotate한 후 translate한 것과, translate 한 후 rotate한 것은 결과가 다르다. 왜 이런 현상이 발생하는지 그 이유를 간략히 설명하세요 (힌트: affine transformation이 표현되는 방식에 대해 생각해볼 것).

23. 아래의 render() 함수는 부정확하게 lighting이 되는 렌더링 결과를 만든다. Lighting이 제대로 되려면 아래 코드에서 어떤 부분이 추가 혹은 변경되어야 하는지 적으세요. drawFrame(), drawCube_glDrawArray() 함수는 모두 정확하게 구현되었고, np로 표현되는 numpy module 모듈의 사용이나 global variable의 사용, gluLookAt() 의 사용에는 모두 문제가 없으며, render() 함수 외에 다른 곳에서 gl*() 함수를 사용하는 곳은 없다고 가정합니다.

학번: **이름:**

```
def render():
    global gCamAng, gCamHeight
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT)

    glEnable(GL_DEPTH_TEST)

    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluPerspective(45, 1, 1,10)

    glMatrixMode(GL_MODELVIEW)
    glLoadIdentity()
    gluLookAt(5*np.sin(gCamAng),gCamHeight,5*np.cos(gCamAng), 0,0,0,
0,1,0)

    drawFrame()

    glEnable(GL_LIGHTING)
    glEnable(GL_LIGHT0)

    glPushMatrix()
    lightPos = (3.,4.,5.,1.)
    glLightfv(GL_LIGHT0, GL_POSITION, lightPos)
    glPopMatrix()

    lightColor = (1.,1.,1.,1.)
    ambientLightColor = (.1,.1,.1,1.)
    glLightfv(GL_LIGHT0, GL_DIFFUSE, lightColor)
    glLightfv(GL_LIGHT0, GL_SPECULAR, lightColor)
    glLightfv(GL_LIGHT0, GL_AMBIENT, ambientLightColor)

    objectColor = (1.,0.,0.,1.)
    specularObjectColor = (1.,1.,1.,1.)
    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, objectColor)
    glMaterialfv(GL_FRONT, GL_SHININESS, 10)
    glMaterialfv(GL_FRONT, GL_SPECULAR, specularObjectColor)

    glPushMatrix()
    glScalef(.2,.2,.2)
    drawCube_glDrawArray()
    glPopMatrix()

    glDisable(GL_LIGHTING)
```

24. 아래 bvh 파일이 표현하는 모션의 첫번째 프레임에서 “link1” 관절의 global frame에 대해 표현된 위치를 T(v) for translation (v is a vector), Rx(θ), Ry(θ), Rz(θ) for rotation (θ in degrees) 함수들과 원점(origin point) **O**을 이용해 표현하세요.

학번: **이름:**

HIERARCHY

ROOT link0

{

OFFSET 0 0 0

CHANNELS 6 Xposition Yposition Zposition Xrotation Yrotation Zrotation

JOINT link1

{

OFFSET 1 0 0

CHANNELS 3 Zrotation Xrotation Yrotation

JOINT link2

{

OFFSET 0 1 0

CHANNELS 3 Zrotation Xrotation Yrotation

End Site

{

OFFSET 0 0 1

}}}}

MOTION

Frames: 1

Frame Time: 0.033333

3 2 1 30 20 10 40 50 60 0 0 0

학번: **이름:**

25. p_0, p_1 두 점을 지나는 직선을 1차 다항식으로 표현하고자 한다.

    1) coefficients & variable로 정리된, 추가 파라미터 t에 대한 1차 다항식으로 표현하세요.

    2) basis functions & points로 정리된, 추가 파라미터 t에 대한 1차 다항식으로 표현하세요.

    3) 2)번의 답안에서 p_0, p_1 각 점에 대한 basis function $b_0(t)$, $b_1(t)$가 각각 무엇인지 적으세요.

26. 한 개의 quad polygon을 렌더링 하기 위한 아래의 코드를 빈 칸 (a), (b), (c)를 채워 완성하세요. 답안은 숫자(scalar)로 작성해야 하며, python 변수명 등 python expression 형식의 답안은 오답처리 됩니다.

```
varr = np.array([
              (  0 ,  0 ), # v0
              ( .5 ,  0 ), # v1
              ( .5 , .5 ), # v2
              (  0 , .5 ), # v3
              ], 'float32')
iarr = np.array([
          (0,1,2,3),
          ])
glVertexPointer((a), GL_FLOAT, (b), varr)
glDrawElements(GL_QUADS, (c), GL_UNSIGNED_INT, iarr)
```

27. 오른쪽 그림으로 표현된 각각의 2차원 변환에 해당하는 transformation matrix를 왼쪽에서 찾으세요. 답안은 빈칸 (1), (2), (3), (4), (5)에 해당하는 답안의 순서로 적을 것.

학번: 이름:

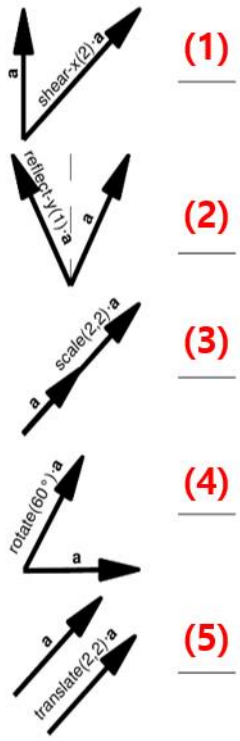
a) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

b) $\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

c) $\begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}$

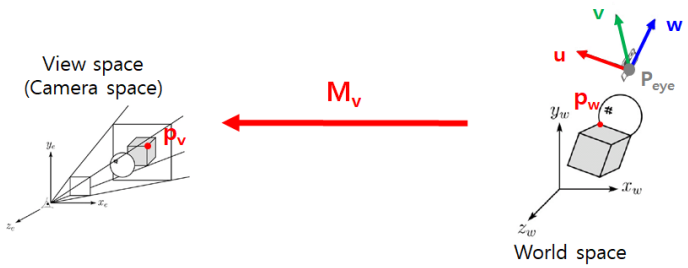
d) $\begin{bmatrix} 0.5 & -0.866 & 0 \\ 0.866 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

e) $\begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$



28. Camera frame이 세 개의 orthonormal basis vector **u, v, w** 와 한 개의 point $\mathbf{P}_{eye}$에 의해 정의되어 있다 (아래 그림 참조). 특정 vertex의 world space 상에서의 위치 $\mathbf{p}_w$ 를 camera space 상에서의 위치 $\mathbf{p}_v$ 로 변환하는 viewing transformation matrix $\mathbf{M}_v$ 를 적으세요. (각 unit vector는 **u** = (ux, uy, uz), **v** = (vx, vy, vz), **w** = (wx, wy, wx)로 표현됨).

(답안은 편집기의 수식(Equation)기능을 사용하지 말고 일반 텍스트로 입력할 것. 줄바꿈(new line)을 통해 행렬의 형태로 보이게만 입력하면 됨)

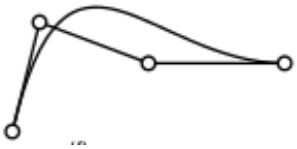


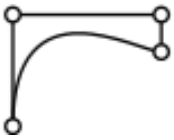
학번: **이름:**

29. 아래 그림은 4가지 종류의 curve와 각 curve를 만드는 control points/polygon을 함께 보여준다. 이 중에서 Bezier curve가 아닌 것들을 모두 고르고, 각각이 Bezier curve가 아닌 이유를 적으세요. 아래에서 control point가 겹쳐 있는 경우는 없습니다 (4가지 모두 4개의 control point를 사용).

a. 

b. 

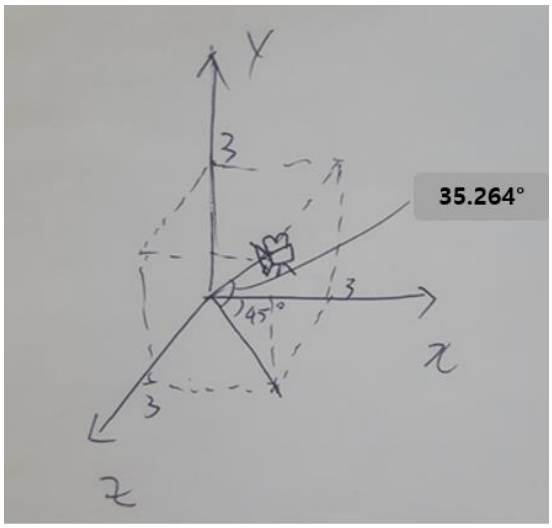
c. 

d. 

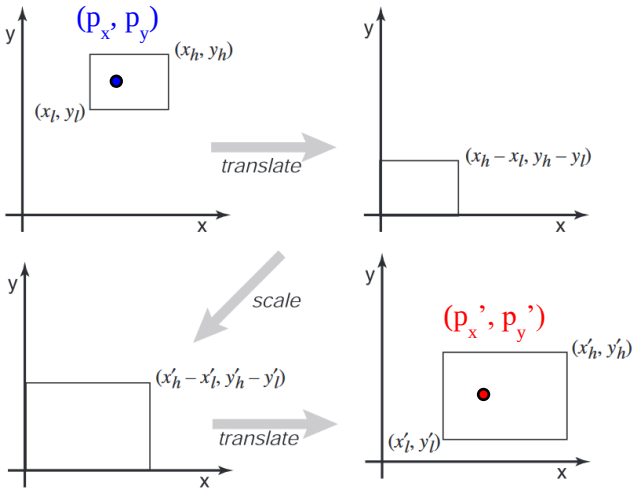
30. "moving camera"와 "moving world"는 본질적으로 동일한 operation이다. 아래 그림을 참고해서, 아래 render 함수와 동일한 결과를 보이도록 아래 render 함수의 gluLookAt 함수 호출을 두 번의 glRotatef() 함수 호출과 한 번의 glTranslatef() 함수 호출로 대체하는 코드를 작성하세요 (square root의 계산이 필요하면 np.sqrt()함수를 사용하면 된다).

```
def render():
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glEnable(GL_DEPTH_TEST)
    glPolygonMode( GL_FRONT_AND_BACK, GL_LINE )
    glLoadIdentity()

    gluPerspective(45, 1, 1,10)

    gluLookAt(3,3,3, 0,0,0, 0,1,0)

    drawFrame()

    glColor3ub(255, 255, 255)
    drawCubeArray()
```

학번: 이름:



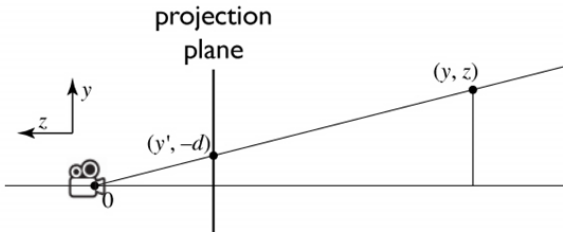
31. 아래 그림과 수식은 사각형 영역 $(x_l, y_l) \sim (x_h, y_h)$ 내부의 한 점 (p_x, p_y)을 다른 사각형 영역 $(x_l', y_l') \sim (x_h', y_h')$ 내부의 한 점 (p_x', p_y')으로 변환하는 과정을 나타내고 있습니다. 빈칸 (a)~(f)를 채워 수식을 완성하세요.



$$\begin{pmatrix} p_x' \\ p_y' \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & 0 & (a) \\ 0 & 1 & (b) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} (c) & 0 & 0 \\ 0 & (d) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & (e) \\ 0 & 1 & (f) \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} p_x \\ p_y \\ 1 \end{pmatrix}$$

학번: **이름:**

32. 아래 그림은 3D point (x, y, z)를 projection plane 상의 2D point (x', y')으로 변환하는 perspective projection 과정을 그린 그림과 수식이다. 아래 수식에서 빈칸 (a) ~ (g)를 채우세요.



$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \text{\_\_(a)\_\_} \\ \text{\_\_(b)\_\_} \\ 1 \end{bmatrix} \sim \begin{bmatrix} \text{\_(c)\_} \\ \text{\_(d)\_} \\ -z \end{bmatrix} = \begin{bmatrix} \text{\_(e)\_} & 0 & 0 & 0 \\ 0 & \text{\_(f)\_} & 0 & 0 \\ 0 & 0 & \text{\_(g)\_} & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

정답:

33. 4 x 4의 shape을 가지는 2차원 numpy.ndarray 객체로 표현된 affine transformation matrix **M**이 있다. 정점 (0,1,0), (0,0,0), (1,0,0)으로 구성된 삼각형을 이 affine transformation matrix **M**으로 변환하여 그리는 아래의 두 가지 버전의 render() 함수를 빈칸 (a) ~ (g)를 채워서 완성하세요. 두 가지 render() 모두 **M**을 파라미터로 받습니다. (Numpy 모듈은 'np'라는 이름으로 접근하여 사용할 수 있음).

학번: **이름:**

```
def render(M):     # implementation 1
    glClear(GL_COLOR_BUFFER_BIT)
    glLoadIdentity()

    glMultMatrixf( (a) )
    glBegin(GL_TRIANGLES)
    glVertex3fv( (b) )
    glVertex3fv( (c) )
    glVertex3fv( (d) )
    glEnd()

def render(M):     # implementation 2
    glClear(GL_COLOR_BUFFER_BIT)
    glLoadIdentity()

    glBegin(GL_TRIANGLES)
    glVertex3fv( (e) )
    glVertex3fv( (f) )
    glVertex3fv( (g) )
    glEnd()
```

34. 아래 텍스트는 6개의 joint를 가지는 어떤 bvh 파일의 hierarchy 섹션의 내용이다. 만일 T-pose 상태에서 각 joint의 global frame에 대해 표현된 위치가 아래 테이블과 같다면, 빈칸 (a), (b), (c), (d), (e)에 들어갈 내용을 채우세요.

joint	global position
J0	(0.0, 0.0, 0.0)
J1	(0.5, 0.5, 0.0)
J2	(0.75, 1.0, 0.0)
J3	(0.25, 1.0, 0.0)
J4	(-0.5, 0.5, 0.0)
J5	(-0.5, 1.0, 0.0)

학번: **이름:**

```
HIERARCHY
ROOT J0
{
	OFFSET 0.00   0.00   0.00
	CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation
Yrotation
	JOINT J1
	{
		OFFSET  ____(a)____
		CHANNELS 3 Zrotation Xrotation Yrotation
		JOINT J2
		{
			OFFSET  ____(b)____
			CHANNELS 3 Zrotation Xrotation Yrotation
			End Site
			{
				OFFSET  0.00   0.50   0.00
			}
		}
		JOINT J3
		{
			OFFSET  ____(c)____
			CHANNELS 3 Zrotation Xrotation Yrotation
			End Site
			{
				OFFSET  0.00   0.50   0.00
			}
		}
	}
	JOINT J4
	{
		OFFSET  ____(d)____
		CHANNELS 3 Zrotation Xrotation Yrotation
		JOINT J5
		{
			OFFSET  ____(e)____
			CHANNELS 3 Zrotation Xrotation Yrotation
			End Site
			{
				OFFSET  0.00   0.50   0.00
			}
		}
	}
}
```