programming assignment 01

zetwhite

01. main 함수의 구성

```
def main():
    #windowing
    glutInit(sys.argv)
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB|GLUT_DEPTH)
    glutInitWindowSize(WINDOW_WIDTH, WINDOW_HEIGHT)
    glutInitWindowPosition(POSITION_X, POSITION_Y)
    glutCreateWindow("201724515 윤승희 programming Assignmnet01")

    glClearColor(0, 0.0, 0.0, 0)

    #register call backs
    glutDisplayFunc(disp)
    glutKeyboardFunc(MyKeyBoard)
    glutIdleFunc(disp)

    glutMainLoop()
```

main함수에서는 #windowing부분에서 window를 띄우고 disp함수와 MyKeyBoard함수를 callback 함수로 등록한다.

02. MyKeyBoard 함수의 구성

```
def MyKeyBoard(key,x,y):
    global viewDistance, viewSatelite, halt
    if key == '+': # Zoom IN
        if viewDistance[0] > 5:
            viewDistance[0] -= 5
            viewDistance[1] -= 5
            viewDistance[2] -= 5
    elif key == '-': # Zoom OUT
        생략…
    elif key == 't':
        생략…
    elif key == 'q':
        exit(0)
    elif key == 'h':
        생략….
    glutPostRedisplay()
    return 0
```

MyKeyBoard 함수에서는 '+', '-', 'q', 'h', 't'의 입력에 따라 적절히 반응하도록 전역변수를 수정해주는 역할을 한다. 여기서 수정한 전역변수를 참조하여 display함으로 적절히 수정되어 화면이 나타나게 된다.

03. disp 함수의 구성

```
def disp() :
    global r, angle_earth_r, angle_earth, x_pluto, y_pluto, distance_earth
    global x_satelite, y_satelite, z_satelite, viewDistance, viewSatelite

    glClear(GL_COLOR_BUFFER_BIT)

    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluPerspective(30, 1.0, 2, 100)

    glMatrixMode(GL_MODELVIEW)
    glLoadIdentity()

    #3/4분면 화면 - 명왕성에서 바라보는 장면
    glViewport(POSITION_X, POSITION_Y, WINDOW_WIDTH/2, WINDOW_HEIGHT/2)
    glPushMatrix()
    angle_earth_r = angle_earth/180.0*pi
    gluLookAt(x_pluto,  y_pluto,  0,  distance_earth*math.cos(angle_earth_r),
distance_earth*math.sin(angle_earth_r), 0, 0, 0, 1)
    drawScene()
    glPopMatrix()

    #4/4분면 화면 - 지구에서 바라보는 장면
        … 생략

    #2/4분면 화면 - 전체 조명
        … 생략

    #1/4분면 화면 - 태양이 12시 방향으로 바라봄
        … 생략
    glFlush()
```

disp화면에서는 각 분할 화면을 출력해준다. 먼저 glViewport로 몇 분할 화면인지 설정해준 다음, glLookAt 함수를 통하여 카메라의 위치(시점, 초점)를 설정해준다. 이때 카메라의 위치가 행성에 위치에 따라 적절히 바껴야함으로, 행성을 위치를 전역변수로 두고, 그 전역변수를 참조하여 카메라의 위치를 설정한다.

카메라의 위치를 적절히 설정한 이후, drawScene()을 호출하여 태양계를 그린다.

04. drawScene()

```
def drawScene() :
    global angle, angle_earth, angle_pluto, slope_satelite, r, pi, halt

    #sun
    glColor3f(1, 0, 0)
    glutSolidSphere(1.0, 20, 20)

    if(not halt) :
        angle += 1
        angle %=361*16

    #명왕성
    glPushMatrix()
    angle_pluto = 1/4.0*angle + 40
    drawPluto(20, 15, angle_pluto, 1, angle_pluto*8)
    glPopMatrix()

    #earth
    glPushMatrix()
    glColor3f(0, 0.5, 1.0)
    angle_earth = angle
    drawEarth(distance_earth, angle_earth, 1, angle_earth*4, 0, 15)

    #인공위성
    drawSatelite(r/3, angle_earth, 1, 0, angle_earth/12)
    glPopMatrix()
```

drawScene()함수에서는 각 행성들을 그린다.
이때 halt라는 전역변수는 현재 행성들이 돌아가고 있는지 아닌지를 나타내주는 전역변수이다.
이 전역변수는 앞에서 본 MyKeyBoard()함수에 의하여 값이 정해진다. 이 값이 False라면 angle 값을 1씩 늘려주면서 행성들이 회전하게 된다.
각 행성들을 그리는 함수는 drawPluto, drawEarth, drawSatelite이며 세 함수 비슷하게 구성되어 있으므로 아래에서는 drawEarth()함수만 살펴본다.

05. drawEarth()

```
def drawEarth(distance, angle, size, spin = 0.0, slope = 0.0, selfR= 0.0) :
    global angle_earth
    glColor3f(0, 0.5, 1.0)
    glRotatef(slope, 1, 0, 0)
    glBegin(GL_LINE_STRIP) #공전 축 그림
    for i in range (0, 361):
        theta = 2.0* 3.141592* i /360
        x = distance * math.cos(theta)
        y = distance * math.sin(theta)
        glVertex3f(x, y, 0)
    glEnd()
    glRotatef(angle, 0, 0, 1) # 공전시킴
    glTranslatef(distance, 0, 0)
    drawCube(selfR, spin, size)
```

drawEarth()함수는 지구 궤도를 그리고 지구를 공전시켜주는 함수이다.
glBegin(GL_LINE_STRIP) ~ glEnd()까지가 지구의 공전 궤도를 그려주는 부분이며,
glRotatef(angle, 0, 0, 1), glTranslatef(distance, 0, 0)이 지구를 x축으로 이동시킨 다음 회전시켜 지구를 공전하도록 만드는 부분이다.
이후 drawCube()함수가 정육면체인 지구를 그리고 회전축을 설정, 자전시켜주는 함수이다.

06. drawCube()

```
CubeCo1 = [0, 0, 0]
CubeCo2 = [0, 1, 0]
CubeCo3 = [0, 1, 1]
CubeCo4 = [0, 0, 1]
CubeCo5 = [-1, 0, 0]
CubeCo6 = [-1, 1, 0]
CubeCo7 = [-1, 1, 1]
CubeCo8 = [-1, 0, 1]

def drawSquare(color, point1, point2, point3, point4):
    glBegin(GL_QUADS)
    glColor(color[0], color[1], color[2])
    glVertex3fv(point1)
    glVertex3fv(point2)
    glVertex3fv(point3)
    glVertex3fv(point4)
    glEnd()

def drawCube(selfR, spin, size) :
    glPushMatrix()
    glRotatef(selfR, 1, 0, 0)
    glRotatef(spin, 0, 0, 1)
    glScalef(size, size, size)
    glTranslatef(0.5, -0.5, -0.5)
    color = [0, 0.5, 1.0]
    drawSquare(color, CubeCo1, CubeCo2, CubeCo3, CubeCo4)
    drawSquare(color, CubeCo1, CubeCo5, CubeCo6, CubeCo2)
    drawSquare(color, CubeCo2, CubeCo6, CubeCo7, CubeCo3)
    drawSquare(color, CubeCo4, CubeCo3, CubeCo7, CubeCo8)
    drawSquare(color, CubeCo4, CubeCo8, CubeCo5, CubeCo1)
    drawSquare(color, CubeCo8, CubeCo7, CubeCo6, CubeCo5)
    glPopMatrix()
```

drawCube는 (0, 0, 0)좌표에 자전하는 육면체를 그려주는 함수이다.
selfR의 각도만큼 육면체를 기울려주며 spin에 따라서 자전하도록 한다. size에 따라서 육면체의 크기를 키워준다.
내부적으로 각 면을 그려주는 함수가 drawSquare이며, drawSquare에서 각 면의 좌표를 넘겨받아 면을 그려준다.