**Computer Graphics, Lab Assignment 4**

Handed out: April 17, 2023

**Recommended due: 13:00,** April 17, 2023

**Hard due: 23:59,** April 23, 2023 **(NO SCORE for late submissions!)**

*Submit your assignment through the LMS system and gitlab.*

1. Write down a Python program to draw a hierarchical model of boxes.
   1. Set the window title to **[studentID]-[assignment#]-[prob#]** and the window size to (480,480).
   2. Start from the following code skeleton.

|  |
| --- |
| **import** glfw  **from** OpenGL**.**GL **import** **\***  **import** numpy **as** np  **from** OpenGL**.**GLU **import** **\***  **def** render**():**  glClear**(**GL\_COLOR\_BUFFER\_BIT **|** GL\_DEPTH\_BUFFER\_BIT**)**  glEnable**(**GL\_DEPTH\_TEST**)**  glMatrixMode**(**GL\_PROJECTION**)**  glLoadIdentity**()**  glOrtho**(-**2**,**2**,** **-**2**,**2**,** **-**1**,**1**)**  glMatrixMode**(**GL\_MODELVIEW**)**  glLoadIdentity**()**    drawFrame**()**  t **=** glfw**.**get\_time**()**  # blue base transformation  glPushMatrix**()**  glTranslatef**(**np**.**sin**(**t**),** 0**,** 0**)**  # blue base drawing  glPushMatrix**()**  glScalef**(**.2**,** .2**,** .2**)**  glColor3ub**(**0**,** 0**,** 255**)**  drawBox**()**  glPopMatrix**()**  # red arm transformation  glPushMatrix**()**  glRotatef**(**t**\*(**180**/**np**.**pi**),** 0**,** 0**,** 1**)**  glTranslatef**(**.5**,** 0**,** .01**)**  # red arm drawing  glPushMatrix**()**  glScalef**(**.5**,** .1**,** .1**)**  glColor3ub**(**255**,** 0**,** 0**)**  drawBox**()**  glPopMatrix**()**  glPopMatrix**()**  glPopMatrix**()**  **def** drawBox**():**  glBegin**(**GL\_QUADS**)**  glVertex3fv**(**np**.**array**([**1**,**1**,**0.**]))**  glVertex3fv**(**np**.**array**([-**1**,**1**,**0.**]))**  glVertex3fv**(**np**.**array**([-**1**,-**1**,**0.**]))**  glVertex3fv**(**np**.**array**([**1**,-**1**,**0.**]))**  glEnd**()**  **def** drawFrame**():**  # draw coordinate: x in red, y in green, z in blue  glBegin**(**GL\_LINES**)**  glColor3ub**(**255**,** 0**,** 0**)**  glVertex3fv**(**np**.**array**([**0.**,**0.**,**0.**]))**  glVertex3fv**(**np**.**array**([**1.**,**0.**,**0.**]))**  glColor3ub**(**0**,** 255**,** 0**)**  glVertex3fv**(**np**.**array**([**0.**,**0.**,**0.**]))**  glVertex3fv**(**np**.**array**([**0.**,**1.**,**0.**]))**  glColor3ub**(**0**,** 0**,** 255**)**  glVertex3fv**(**np**.**array**([**0.**,**0.**,**0**]))**  glVertex3fv**(**np**.**array**([**0.**,**0.**,**1.**]))**  glEnd**()**  **def** main**():**  **if** **not** glfw**.**init**():**  **return**  window **=** glfw**.**create\_window**(**480**,**480**,**'2017123456-lab6-1'**,** **None,None)**  **if** **not** window**:**  glfw**.**terminate**()**  **return**  glfw**.**make\_context\_current**(**window**)**  glfw**.**swap\_interval**(**1**)**  **while** **not** glfw**.**window\_should\_close**(**window**):**  glfw**.**poll\_events**()**  render**()**  glfw**.**swap\_buffers**(**window**)**  glfw**.**terminate**()**  **if** \_\_name\_\_ **==** "\_\_main\_\_"**:**  main**()** |

* 1. Add a green arm at the end of the red arm, and rotate the green arm about its local z axis.
  2. Also render local frames of the blue base, red arm, green arm using drawFrame().
  3. The blue base, red arm, green arm should be rendered using drawBox()**.**
  4. Expected result: Uploaded LabAssignment4-1.mp4
  5. Submit a single .py file - **[studentID]-[assignment#]-[prob#].py**

1. As mentioned in the previous lecture, “moving camera” and “moving world” are two equivalent operations. Based on the following figure, replace the **gluLookAt** call() in the following code with **two glRotatef() calls and one glTranslatef() call** and complete the program.

**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**)**

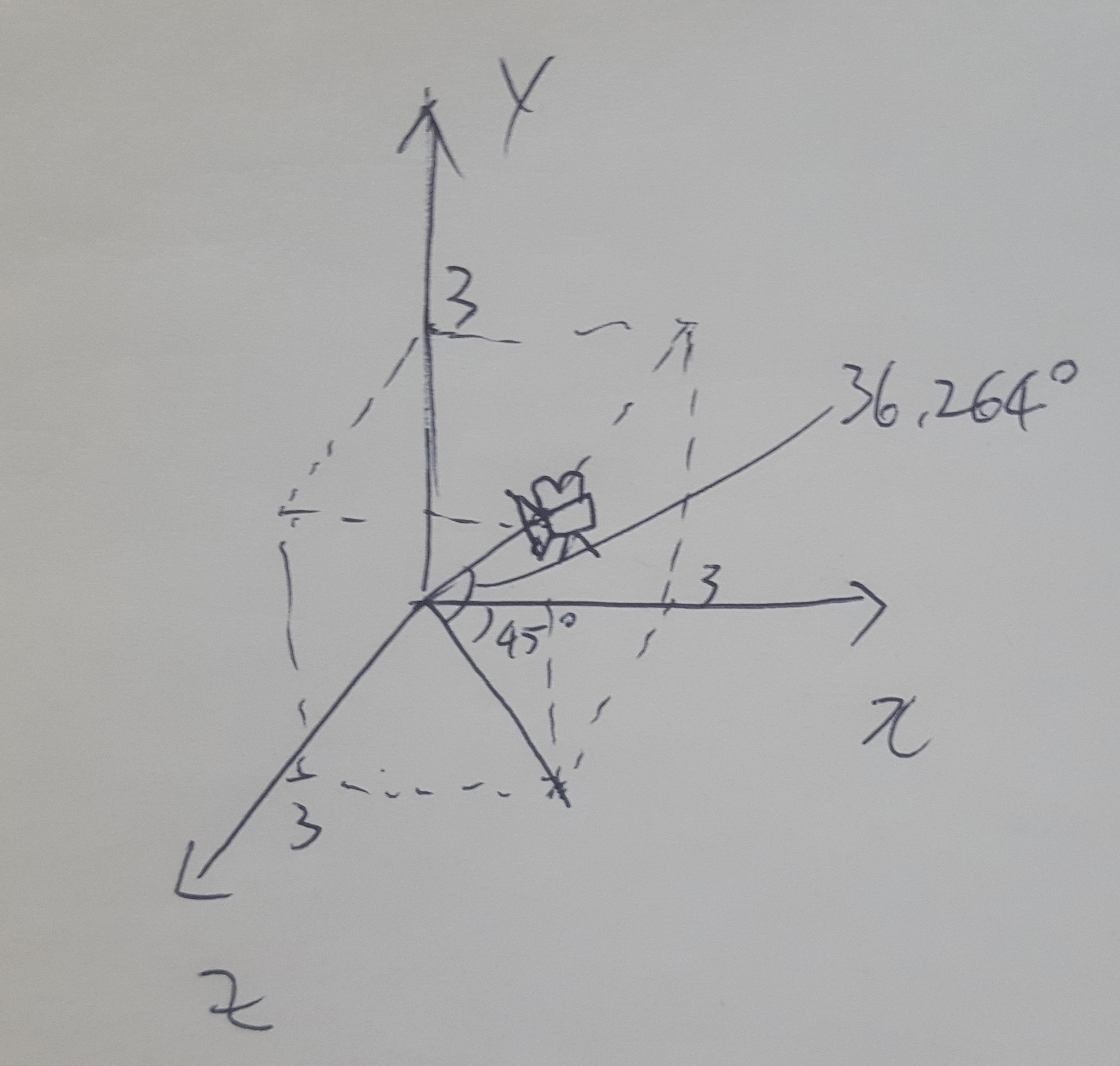
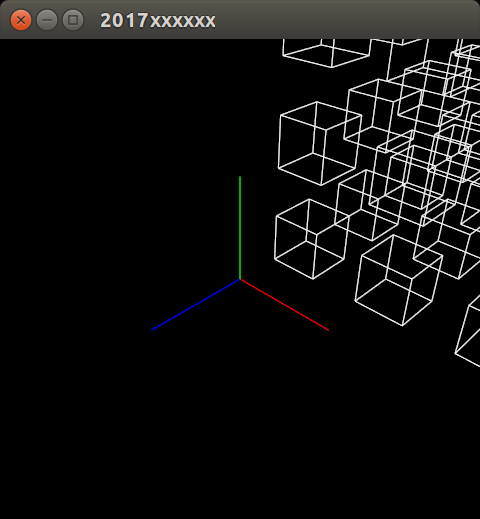
**# Replace this call with two glRotatef() calls and one glTranslatef() call**

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

drawFrame**()**

glColor3ub**(**255**,** 255**,** 255**)**

drawCubeArray**()**

* 1. 
  2. Set the window title to **[studentID]-[assignment#]-[prob#]** and the window size to (480,480).
  3. Find code for drawFrame(), drawCubeArray() from LabAssignment4-2.py.
  4. Your program should render the following scene:
     1. 
  5. Submit a single .py file - **[studentID]-[assignment#]-[prob#].py**