CURSO: CC322 - 2017

Practica 03. Sistema de ventanas GLUT, Primitivas Básicas Transformaciones.
Las modificaciones que haga deberá guardarlas en archivos diferentes.
Al final entregará sus archivos en formato comprimido con el nombre: <nombre-apellido-Lab03>.zip

Copie el siguiente fragmento en un archivo con extension ".py", p.ej. `6.py`
Trate de correrlo y verifique que no haya errores de tipeo ni de otra indole.
Tambien Puede bajarlo desde el repositorio en la nube, en la carpeta LABORATORIO

```
#!
# This is statement is required by the build system to query build info
if __name__ == '__build__':
   raise Exception
import string
__version__ = string.split('$Revision: 1.1.1.1 $')[1]
__date__ = string.join(string.split('$Date: 2007/02/15 19:25:21 $')[1:3], ' ')
__author__ = '<Modificado por: ---Ponga su nombre aqui------------>'
#
from OpenGL.GL import *
from OpenGL.GLUT import *
from OpenGL.GLU import *
import sys
# Some api in the chain is translating the keystrokes to this octal string
# so instead of saying: ESCAPE = 27, we use the following.
ESCAPE = '\033'
# Number of the glut window.
window = 0
trimove = -0.5
cuadmove= 0.5
# A general OpenGL initialization function.  Sets all of the initial parameters.
def InitGL(Width, Height):          # We call this right after our OpenGL window
is created.
    glClearColor(0.0, 0.0, 0.0, 0.0)   # This Will Clear The Background Color To
Black
    glClearDepth(1.0)              # Enables Clearing Of The Depth Buffer
    glDepthFunc(GL_LESS)            # The Type Of Depth Test To Do
    glEnable(GL_DEPTH_TEST)             # Enables Depth Testing
    glShadeModel(GL_SMOOTH)             # Enables Smooth Color Shading

    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()               # Reset The Projection Matrix
                              # Calculate The Aspect Ratio Of The Window
    gluPerspective(45.0, float(Width)/float(Height), 0.1, 100.0)
    glMatrixMode(GL_MODELVIEW)
# The function called when our window is resized (which shouldn't happen if you
enable fullscreen, below)
def ReSizeGLScene(Width, Height):
    if Height == 0:                  # Prevent A Divide By Zero If The Window Is
Too Small
       Height = 1
    glViewport(0, 0, Width, Height)       # Reset The Current Viewport And
Perspective Transformation
```

```
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluPerspective(45.0, float(Width)/float(Height), 0.1, 100.0)
    glMatrixMode(GL_MODELVIEW)
# The main drawing function.
def DrawGLScene():

   # Clear The Screen And The Depth Buffer
   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
   glLoadIdentity()              # Reset The View
   # Move Left 1.5 units and into the screen 6.0 units.
   glTranslatef(-1.5, 0.0, -6.0)
   # Since we have smooth color mode on, this will be great for the Phish
Heads :-).
   # Draw a triangle
   glBegin(GL_POLYGON)                  # Start drawing a polygon
   glColor3f(1.0, 0.0, 0.0)             # Red
   glVertex3f(0.0, 1.0, 0.0)            # Top
   glColor3f(0.0, 1.0, 0.0)             # Green
   glVertex3f(1.0, -1.0, 0.0)           # Bottom Right
   glColor3f(0.0, 0.0, 1.0)             # Blue
   glVertex3f(-1.0, -1.0, 0.0)          # Bottom Left
   glEnd()                              # We are done with the polygon
   # Move Right 3.0 units.
   glTranslatef(3.0, 0.0, 0.0)
   # Draw a square (quadrilateral)
   glColor3f(0.3, 0.5, 1.0)             # Bluish shade
   glBegin(GL_QUADS)                    # Start drawing a 4 sided polygon
   glVertex3f(-1.0, 1.0, 0.0)           # Top Left
   glVertex3f(1.0, 1.0, 0.0)            # Top Right
   glVertex3f(1.0, -1.0, 0.0)           # Bottom Right
   glVertex3f(-1.0, -1.0, 0.0)          # Bottom Left
   glEnd()                              # We are done with the polygon
   #  since this is double buffered, swap the buffers to display what just got
drawn.
   glutSwapBuffers()



# The function called whenever a key is pressed. Note the use of Python tuples to
pass in: (key, x, y)
def keyPressed(*args):
   # If escape is pressed, kill everything.
    if args[0] == ESCAPE:
       sys.exit()
def main():
   global window
   # For now we just pass glutInit one empty argument. I wasn't sure what should
or could be passed in (tuple, list, ...)
   # Once I find out the right stuff based on reading the PyOpenGL source, I'll
address this.
   glutInit(sys.argv)
   # Select type of Display mode:
   #  Double buffer
   #  RGBA color
   # Alpha components supported
   # Depth buffer
   glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE | GLUT_DEPTH)

   # get a 640 x 480 window
   glutInitWindowSize(640, 480)
   # the window starts at the upper left corner of the screen
   glutInitWindowPosition(0, 0)

   # Okay, like the C version we retain the window id to use when closing, but
for those of you new
   # to Python (like myself), remember this assignment would make the variable
local and not global
   # if it weren't for the global declaration at the start of main.
   window = glutCreateWindow("Leccion semana 4 - transformaciones")
       # Register the drawing function with glut, BUT in Python land, at least
using PyOpenGL, we need to
   # set the function pointer and invoke a function to actually register the
callback, otherwise it
   # would be very much like the C version of the code.
   glutDisplayFunc(DrawGLScene)

   # Uncomment this line to get full screen.
   #glutFullScreen()
   # When we are doing nothing, redraw the scene.
   glutIdleFunc(DrawGLScene)

   # Register the function called when our window is resized.
   glutReshapeFunc(ReSizeGLScene)

   # Register the function called when the keyboard is pressed.
   glutKeyboardFunc(keyPressed)
   # Initialize our window.
   InitGL(640, 480)
   # Start Event Processing Engine
   glutMainLoop()
# Print message to console, and kick off the main to get it rolling.
print "Hit ESC key to quit."
main()
```

A continuacion Usted Modificará el programa.

1. Modificando la funcion glTranslatef haga que los objetos presentados se muevan en direcciones Opuestas sobre el eje X. Cuando los objetos lleguen al limite de la ventana deberan cambiar su direccion de movimiento. Genere el archivo 7.py

2.Utilizando la funcion glRotatef modifique 6.py y haga que los objetos presentados empiecen a rotar. El triangulo al rededor del eje Y, el cuadrado al rededor del eje X. Genere el archivo 8.py

3. A partir de 8.py. Cambie el orden de las funciones glTranslatef y glRotatef. Que sucede?.
Genere el archivo 81.py

4. a partir de 8.py. Cambie la funcion keyPressed para que acepte las teclas **u** (up) y **d** (down) para trasladar los objetos en direcciones opuestas sobre el eje Z. y las teclas **h** (high) y **l** (low) para aumentar y disminuir la velocidad de traslacion cada ves que se presionen. Genere el archivo 82.py

5. A partir de los archivos anteriores dibuje un “taladro” con punta triangular rotatoria y cuerpo de pentagono, que pueda controlar su movimiento a lo largo de los 3 ejes x, y, z Genere el archivo 9.py