PRACTICA DE EVALUACIÓN CONTINUA 02

Crear un programa que presente la escena de la figura 1. La escena muestra varios objetos situados en los vértices de un cubo de dimensión 3 centrado en el origen de coordenadas. Para su construcción tener presente que cada objeto tiene su propio origen en el origen del sistema de coordenadas, todos están representados en modelo alámbrico.

Para indicar el sentido creciente de los ejes se han dibujado unos conos, rojo para el eje X, azul para el eje Y, amarillo para el eje Z. Los ejes para valores positivos tienen color negro y grosor 3, los ejes para valores negativos tienen grosor 1 y el color del cono correspondiente.

La escena debe girar en sentido positivo sobre el **eje-Y azul** , si se teclea la letra 'y' , sobre el **eje X rojo** si se teclea la letra 'x' y sobre el **eje Z amarillo** si se teclea la letra 'z', si las teclas de selección se corresponden con letras mayúsculas los giros serán negativos.

Cuando se teclee 'e' la escena debe aparecer mas pequeña y si se teclea 'E' debe aumentar, utilizar para ello la función de escalado. `glScaled(porc_x, porc_y, porc_z);`

Cuando se teclee 't' la tetera debe rotar sobre su eje vertical.

Para la realización de esta PEC debéis tener en cuenta los conceptos explicados el documento *z-Buffer y pila de matrices OpenGL* del TEMA 2 TRANSFORMACIONES 3D de la asignatura en UBUVirtual.

Cada alumno deberá entregar un fichero con el código fuente. El nombre del fichero debe ser: *PEC2_apellido1_apellido2_nombre.cpp*
Cada alumno deberá entregar un vídeo (máximo 10 minutos) comentando el código del programa y mostrando su ejecución. Para grabar el video se recomienda utilizar la herramienta https://screencast-o-matic.com/

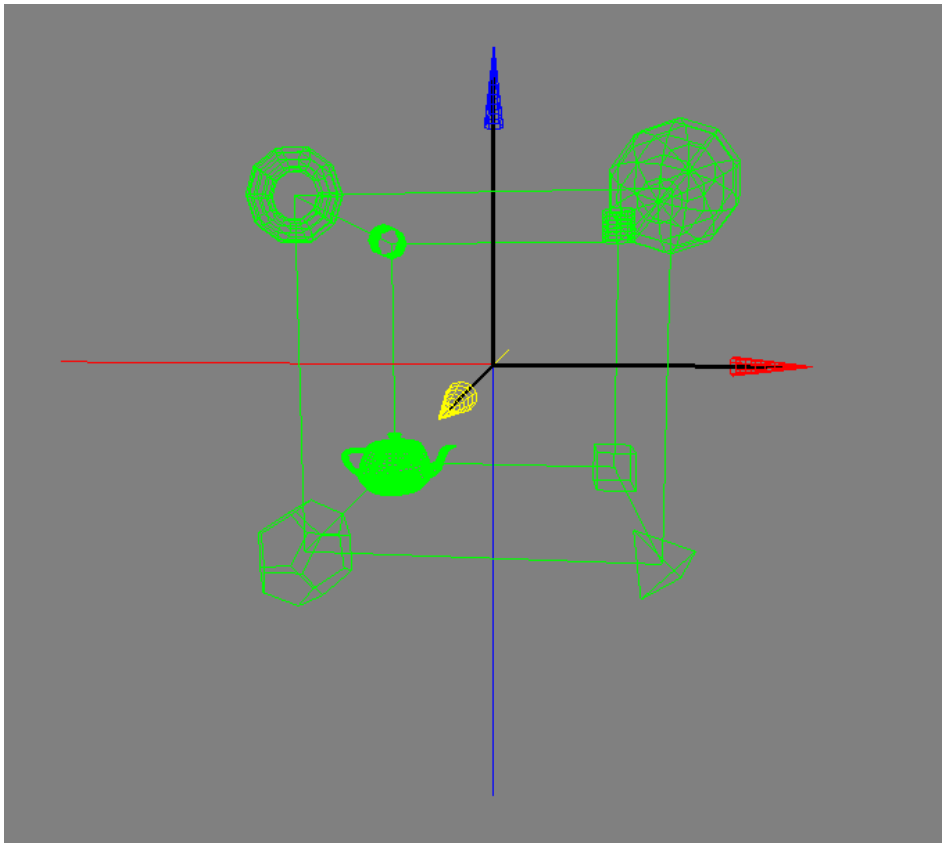

Figura 1

En la figura mostrada se ha utilizado proyección perspectiva, pero podéis probar con proyección paralela.

La relación figuras es la siguiente: dodecaedro, tetraedro, cubo, tetera, toro, esfera, cilindro_1 y cilindro_2. Los cilindros tienen distintas orientaciones.

Utiliza el tamaño a las figuras que consideres oportuno para una correcta visualización.

GLUT tiene varios objetos definidos por defecto listos para visualizar, tales objetos incluyen esferas, toros, conos y sólidos platónicos (poliedros regulares), y la famosa *taza de té*. Estas se pueden representar en forma alámbrica o modelo sólido con las caras sombreadas. La siguiente lista muestra las funciones usadas para dibujar estos objetos:

- **cube**: `glutWireCube(GLdouble size)`; Cada lado tiene de longitud `size`.
- **sphere**: `glutWireSphere(GLdouble radius, GLint nSlices, GLint nStacks)`
- **torus**: `glutWireTorus(GLdouble inRad, GLdouble outRad, Glint nSlices, GLint nStacks)`
- **teapot:** `glutWireTeapot(Gldouble size)`

También existe glutSolidCube(); glutSolidSphere(), etc. La forma del toro se define por el radio interno y el externo. La esfera y el toro se aproximan a caras poligonales, y es posible definir los parámetros nSlices y nStacks para especificar cuantas caras se deben usar en la aproximación. nSlices es el número de subdivisiones alrededor del eje z, y nStacks es el número de "bandas" a lo largo del eje z, como si fueran disco.

Las funciones usadas para mostrar los cuatro sólidos platónicos (el quinto es el cubo ya mencionado) son las siguientes:

- **tetrahedron:** `glutWireTetrahedrom()`
- **octahedron:** `glutWireOctahedron ()`
- **dodecahedron:** `glutWireDodacahedron()`
- **icosahedron:** `glutWireIcosahedron()`

Todas los cuerpos geométricos se definen en el origen. También están disponibles los siguientes sólidos:

- **cono:** `glutWireCone(GLdouble baseRad, GLdouble height, GLint nSlice, GLint nStacks)`
- **tapered cylinder:** `gluCylinder(GLUquadricObj * qobj, GLdouble baseRad, GLdouble topRad, Gldouble height, GLint nSlice, GLint nStacks)`

El eje del cono y del cilindro cónico coincide con el eje *z*. Las bases están situadas en *z* = 0 y *z* = height a lo largo del eje *z*. El radio del cono y del cilindro para z = 0 se define por baseRad. El radio del cilindro cónico para *z* = height es topRad.

El **cilindro** es una familia de formas, distinguidas por el valor de topRad. Cuando topRad es 1, no es cónico, y estamos en el caso del típico cilindro de base circular. Cuando topRad es 0, el cilindro es el cono.

Nota que el dibujo del cilindro en OpenGL requiere algún trabajo extra, porque es un caso especial de una superficie cuádrica. Para dibujar primero hay que definir un objeto nuevo cuadrático, segundo hay que asignarle un estilo de línea para dibujar (GLU_LINE, el modelo alámbrico, GLU_FILL, para el sólido), y tercero dibujar el objeto. El código se muestra a continuación.

```
GLUquadricObj * qobj = gluNewQuadric(); // crea un objeto cuadrático
gluQuadricDrawStyle(qobj, GLU_LINE); //estilo alámbrico
gluCylinder(qobj, baseRad, topRad, height, nSlices, nStacks); Dibuja el
cilindro
```

Algunas páginas de referencia para Open GL:

http://www.glprogramming.com/blue/index.html
http://www.opengl.org/sdk/docs/man/
http://www.opengl.org/resources/libraries/glut/glut-3.spec.pdf
http://www.opengl.org/documentation/specs/glut/spec3/spec3.html

LIBRO ROJO Y AZUL DE OPENGL
http://glprogramming.com/red/
http://glprogramming.com/blue/
http://www.opengl.org/documentation/specs/glut/spec3/node113.html