

Proyecto II

Este proyecto consiste en crear un pequeño museo que consta de dos pisos como se ve en la figura:

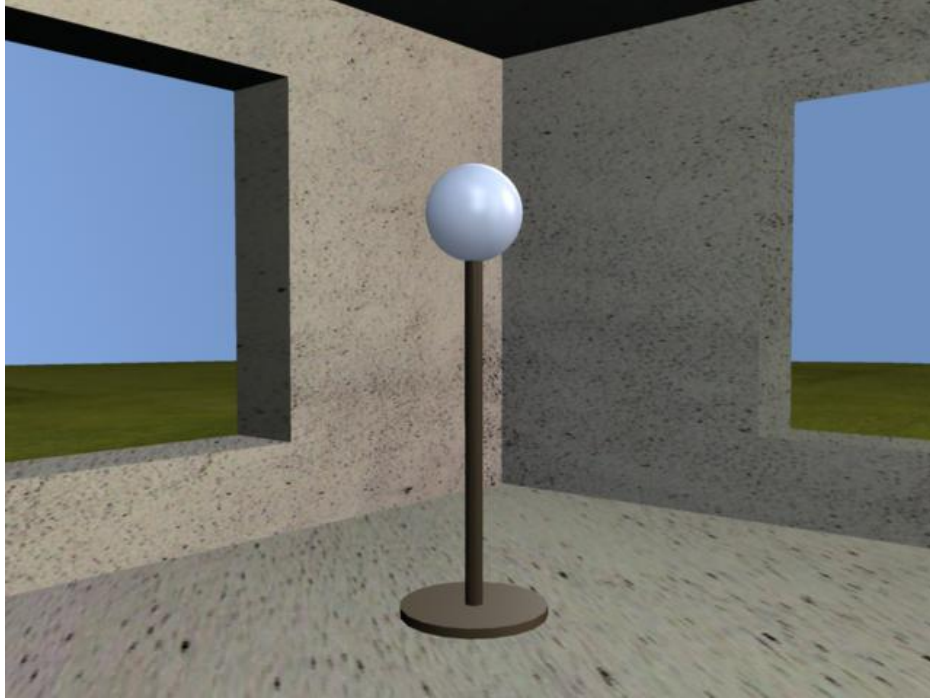


El programa deberá consistir en lo siguiente:

- El usuario debe desplazarse en cualquier dirección.



- Se deberán de modelar al menos 4 lámparas.



Nota: Pueden diseñar la lámpara como quieran no necesariamente deben modelarla como se muestra en la figura. Dicha lámpara debe ser modelada por ustedes usando las primitivas de opengl. Y recuerden que en opengl tienen máximo de 8 luces.

- Deben de colocar al menos 6 objetos en el museo, 3 por cada piso.



- Las paredes y piso del museo deben ser texturizadas.
- Algunos objetos deben estar colocados encima de una base como se muestra en la figura de arriba.

Bono Extra:

- Añadir más decoración al museo.

Link de Interés.

- Repositorio de objeto:
 - <http://shapes.aimatshape.net/>
 -
- Librería para cargar los objetos:
 - <http://www.cs.princeton.edu/gfx/proj/trimesh2/>
- IDE de C++
 - <http://www.codeblocks.org/>
- Texturas:
 - <http://www.cgtextures.com/>

NOTA 1: A la librería del Trimesh no se le da soporte para Visual C++, en pruebas que hice no he podido compilarla en Visual Studio, por consiguiente recomiendo que usen el GCC que viene en el programa de Code Blocks, el cual se puede instalar tanto en Linux como en Windows. Aquí les presento un programa de ejemplo que recomiendo correr antes de empezar hacer el museo.

1. Abren el programa Code Blocks, y como nuevo proyecto seleccionan Console Application.
2. Luego en el main.cpp que les crea colocan este código:

```
#include "TriMesh.h"
#include <iostream>
using namespace std;

int main(int argc, char *argv[])
{
    const char *filename = "foo.ply";
    TriMesh *m = TriMesh::read(filename);
    if (!m)
        exit(1);
}
```

```

cout << "There are " << m->vertices.size() << " vertices" << endl;
cout << "Vertex 0 is at " << m->vertices[0] << endl;

// Convert triangle strips to faces, if necessary
m->need_faces();
cout << "Face 0 has vertices " << m->faces[0][0] << ", "
    << m->faces[0][1] << ", and " << m->faces[0][2] << endl;

m->need_normals();
cout << "Vertex 0 has normal " << m->normals[0] << endl;
cout << "Final " << m->normals[0] << endl;
}

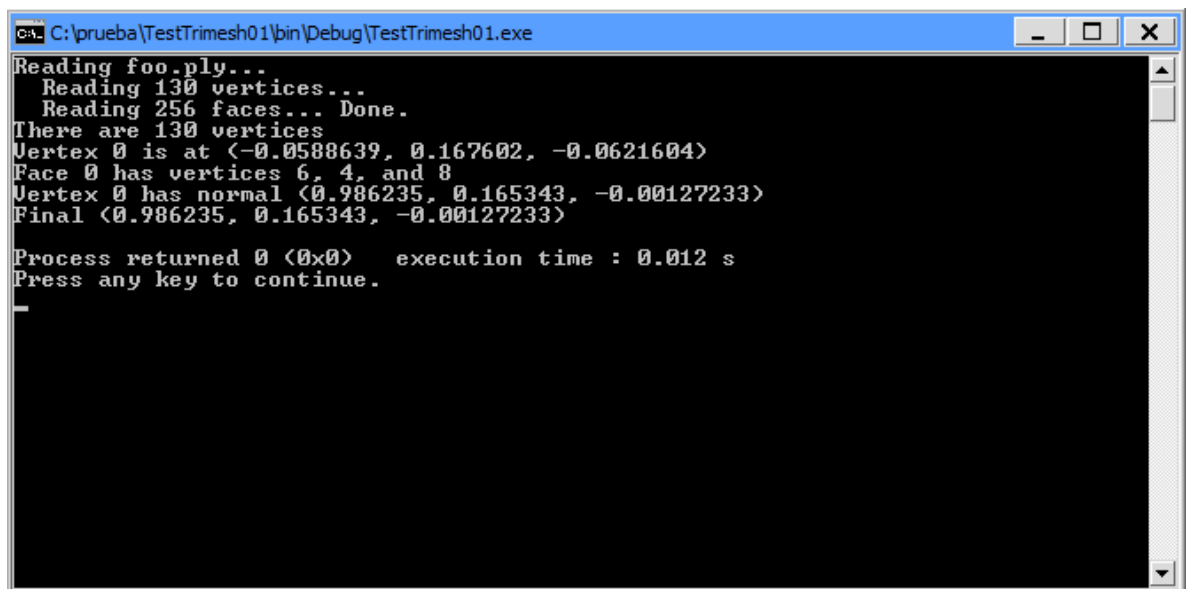
```

3. Luego en el menú Project – Build Options:

- En Linker Settings van añadir el archivo libtrimesh.a que se encuentra en la carpeta de lib.Win32.
- En Search Directory van añadir la carpeta Include del trimesh.

4. Luego copien el archivo floo.ply en la carpeta donde está el proyecto.

5. Luego si todo compila bien les debe de aparecer una ventana igual a esta



```

C:\prueba\TestTrimesh01\bin\Debug\TestTrimesh01.exe
Reading foo.ply...
Reading 130 vertices...
Reading 256 faces... Done.
There are 130 vertices
Vertex 0 is at (-0.0588639, 0.167602, -0.0621604)
Face 0 has vertices 6, 4, and 8
Vertex 0 has normal (0.986235, 0.165343, -0.00127233)
Final (0.986235, 0.165343, -0.00127233)

Process returned 0 (0x0)   execution time : 0.012 s
Press any key to continue.

```

Fíjense que este ejemplo muestra cómo obtener las informaciones de vértices, caras y normales, que es lo que básicamente necesitan para cargar los modelos en opengl.

Nota 2: Recuerden que el uso de las normales es importante al momento de crear objeto con las primitivas de OpenGL, como GL_TRIANGLES, GL_QUADS, etc, si al modelar el museo no tienen en cuenta el uso de las normales, la iluminación no funcionara en el museo, así mismo, si al momento de cargar los modelos no establecen las normales, no se verán iluminados los modelos. Las funciones glut como Sphere, Cube , etc, se ven iluminados por que por defecto dichas funciones carga normales predeterminadas, lo que no ocurre cuando usan las primitivas de OpenGL.

Nota 3: Algunos objetos poseen una cantidad de triángulos muy elevadas, en algunos modelos del repositorio se permite seleccionar el mismo modelo con diferentes cantidades de triángulos. Traten de no trabajar con objetos de alta resolución, ya que al cargar un objeto tal vez no haya problema pero al cargar 3 o 4 objetos de alta resolución, el rendimiento de la computadora puede bajar, claro está, esto dependerá también del equipo en donde estén corriendo el programa.