**UNIVERSIDAD**

**REY JUAN CARLOS**

GRÁFICOS 3D

MÁSTER DE INFORMÁTICA GRÁFICA

JUEGOS Y REALIDAD VIRTUAL

PRÁCTICA Nº2

|  |  |
| --- | --- |
| Integrantes: | Carlos Rizzo  Joan Torres López |
|  | Cristian Adrian Penizzotto |
|  |  |
|  | Periodo 2019-2020 |

Contenido.

[1. Objetivo. 3](#_Toc30322388)

[2. Desarrollo. 3](#_Toc30322389)

[3. Desarrollo de Parte Obligatoria. 3](#_Toc30322390)

[3.1. Iluminando al objeto con 2 fuentes de luz. 3](#_Toc30322391)

[3.2. Funcionalidad de atenuación de intensidad lumínica. 4](#_Toc30322392)

[3.3. Luz direccional y luz focal. 5](#_Toc30322393)

[4. Desarrollo de Parte Opcional. 7](#_Toc30322394)

[4.1. Implementación de Bump Mapping. 7](#_Toc30322395)

[4.2. Implementación de niebla. 8](#_Toc30322396)

[4.3. Nuevo modelo. 9](#_Toc30322397)

[5. Conclusiones. 10](#_Toc30322398)

# Objetivo.

El objetivo de la práctica es entender y poner en práctica las técnicas de iluminación vistas en clase en un cauce gráfico real. Se repasarán conceptos de iluminación, modelos de sombreados, tipos de luces y como configurar los shaders para obtener los resultados deseados.

# Desarrollo.

A continuación, se describe de forma abreviada los resultados obtenidos para las distintas instancias del práctico.

# Desarrollo de Parte Obligatoria.

# Iluminando al objeto con 2 fuentes de luz.

El objetivo de este apartado es iluminar al cubo con distintas fuentes de luces. En nuestro caso, vemos en la Figura 3.1‑1 el cubo iluminado con dos fuentes de luz. En la Figura 3.1‑2 la definición de las luces y la Figura 3.1‑3 muestra el código para iluminar un fragmento.

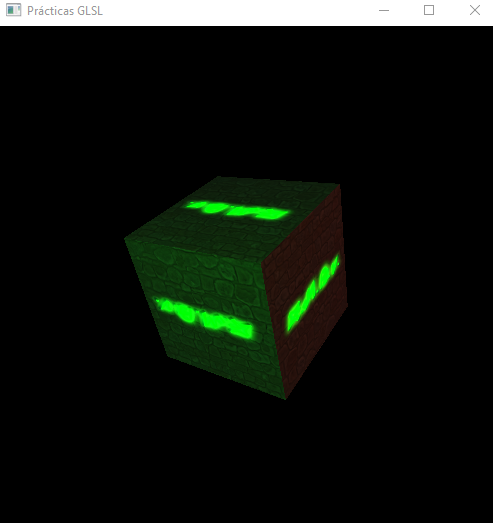
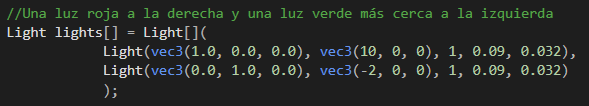
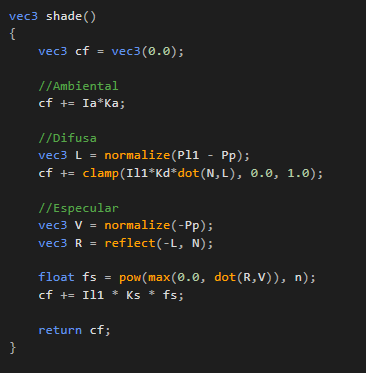


Figura 3.1‑1 Iluminación de 2 fuentes de luz sobre el cubo



**Figura 3.1‑2** Definición de las luces



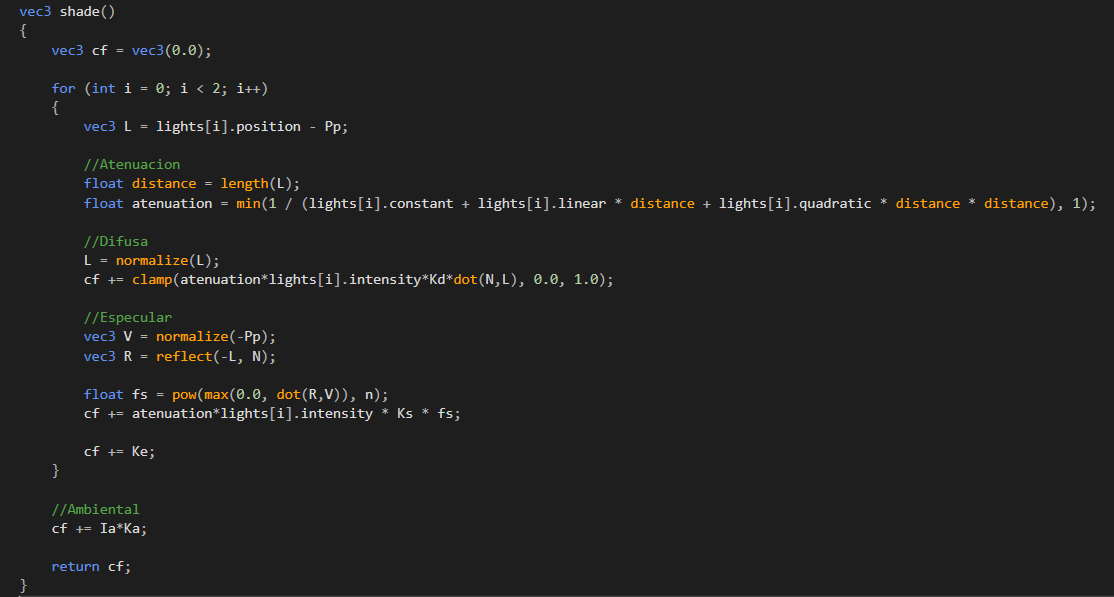
**Figura 3.1‑3** Código de iluminación Phong

# Funcionalidad de atenuación de intensidad lumínica.

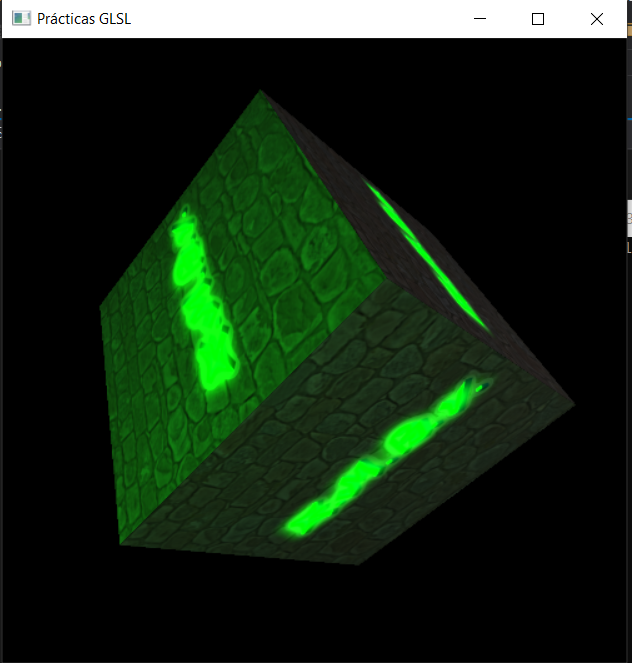
A continuación, se implementó una atenuación de la intensidad lumínica en función de la distancia desde la fuente de luz.

En Figura 3.2‑1, se muestra el código para realizar la atenuación, en el shader.v3.frag.

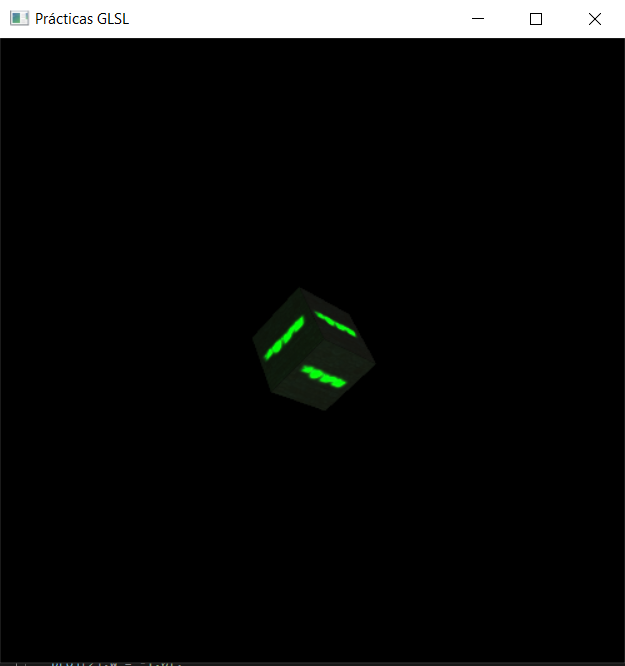
En Figura 3.2‑2 y Figura 3.2‑3 se observa al cubo en dos posiciones diferentes. En donde se pueden apreciar los efectos de la intensidad lumínica atenuados por la distancia desde las fuentes de luz que se mantienen en la misma posición que la cámara.



**Figura 3.2‑1** Atenuación de intensidad lumínica con la distancia



**Figura 3.2‑2** Intensidad lumínica sobre el cubo cercano al plano near

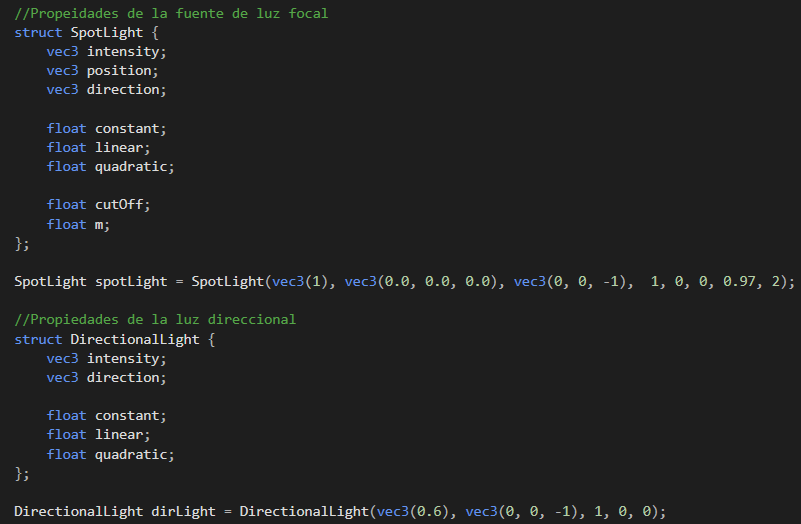


**Figura 3.2‑3** Intensidad lumínica sobre el cubo cercano al plano far

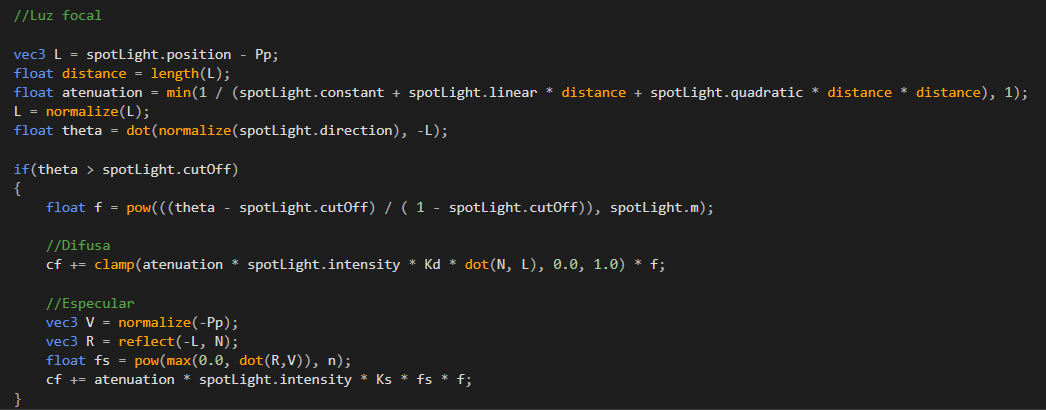
# Luz direccional y luz focal.

Se detalla en esta consigna la implementación de una luz direccional y una focal. Entre las figuras “Figura 3.3‑1, Figura 3.3‑2, Figura 3.3‑3” se muestra el código para la definición de las luces.

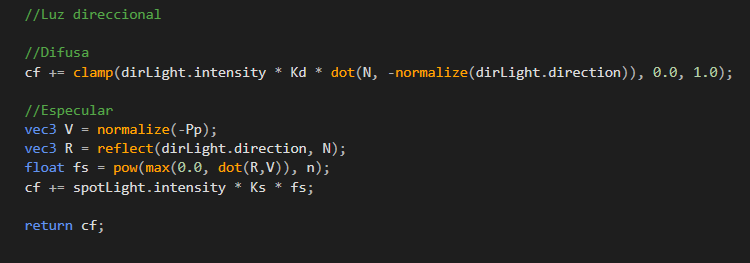
En Figura 3.3‑4, vemos estas luces sobre el cubo, en donde se observa en una zona la intensidad lumínica concentrada debido a la luz focal y hacia los extremos la influencia de la luz direccional.



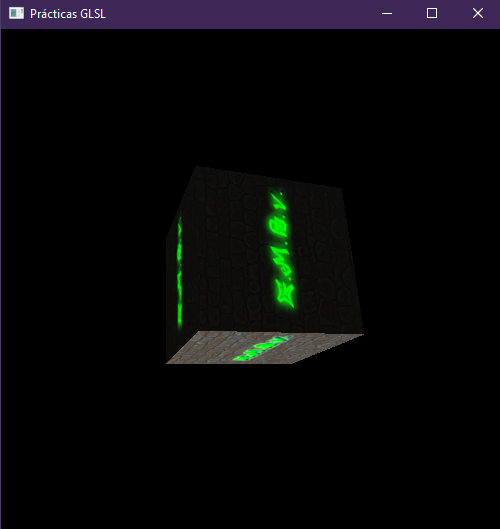
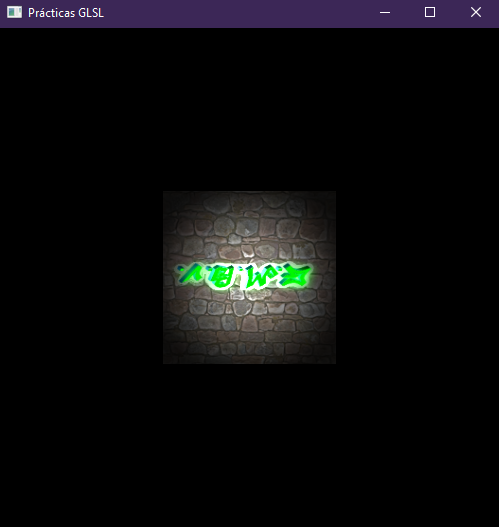
**Figura 3.3‑1** Definiciones de la luz focal y direccional



**Figura 3.3‑2** Implementación de la luz focal



**Figura 3.3‑3** Implementación de la luz direccional



**Figura 3.3‑4** Luz focal (izq) y direccional (der)

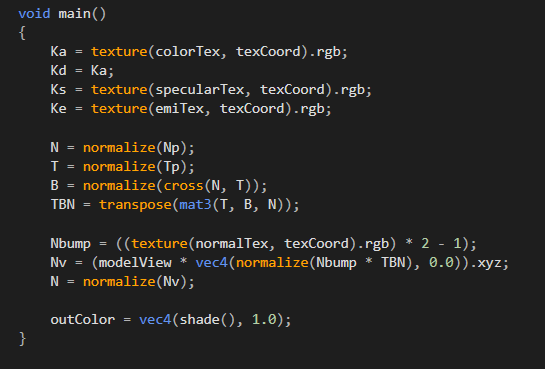
# Desarrollo de Parte Opcional.

# Implementación de Bump Mapping.

A continuación, se aplicará la técnica de bump mapping, con la cual lograremos un efecto de profundidad o rugosidad en las superficies, solamente cambiando las normales.

En Figura 4.1‑1 se muestra el sharder.v6.frag escrito para modificar las normales.

En Figura 4.1‑2 se observa el efecto rugoso y de profundidad logrado sobre el cubo en las rocas y la unión de estas.



**Figura 4.1‑1** Código en shader.v6.frag



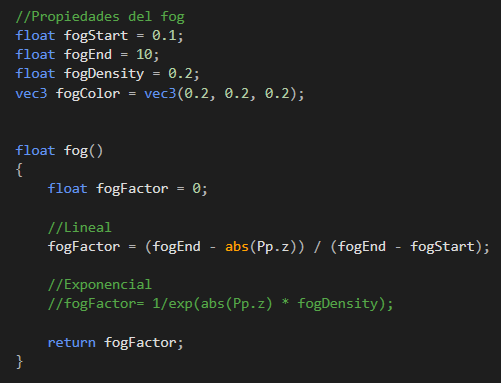
**Figura 4.1‑2** Imagen de cubo con efecto de rugosidad

dada por el bump mapping

# Implementación de niebla.

Se realiza a continuación la implementación de niebla “fog”, en función de la distancia desde el plano near, como se observa en la Figura 4.2‑2 y Figura 4.2‑3. Como se aprecia, el objeto a medida que se aleja del plano near, empieza a notarse la niebla sobre el cubo. El efecto de la niebla será diferente dependiendo de cual se haya programado, ya sea lineal o exponencial. Ver en Figura 4.2‑4 el modelo de fog lineal y en Figura 4.2‑5 el modelo de fog exponencial.

En Figura 4.1‑1, se detalla el código implementado para el fog en el shader. Se programó tanto una fog lineal, como una exponencial.

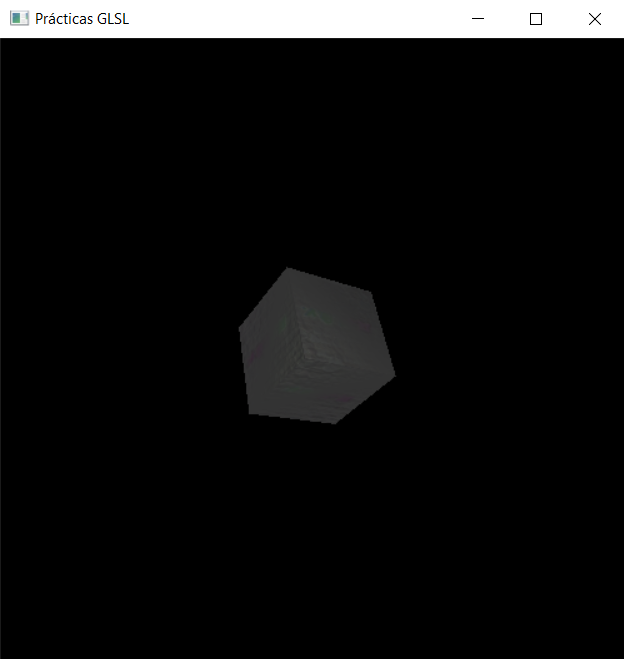


**Figura 4.2‑1 Código en** Shader.7.frag para la

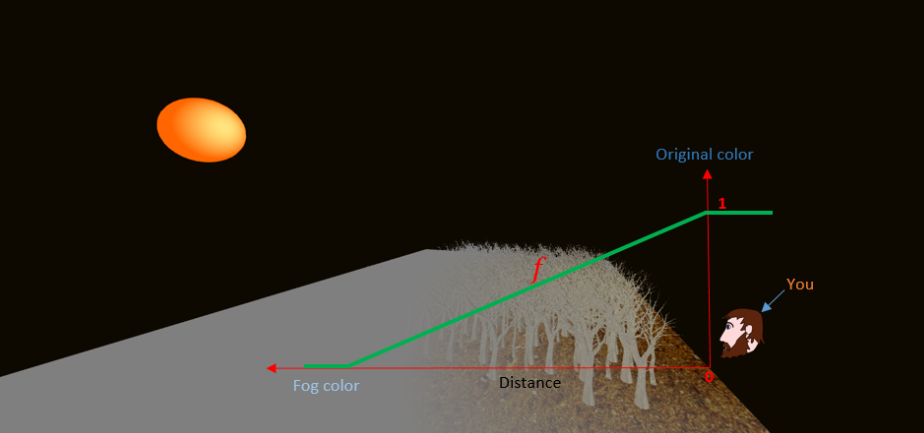
implementación de Fog



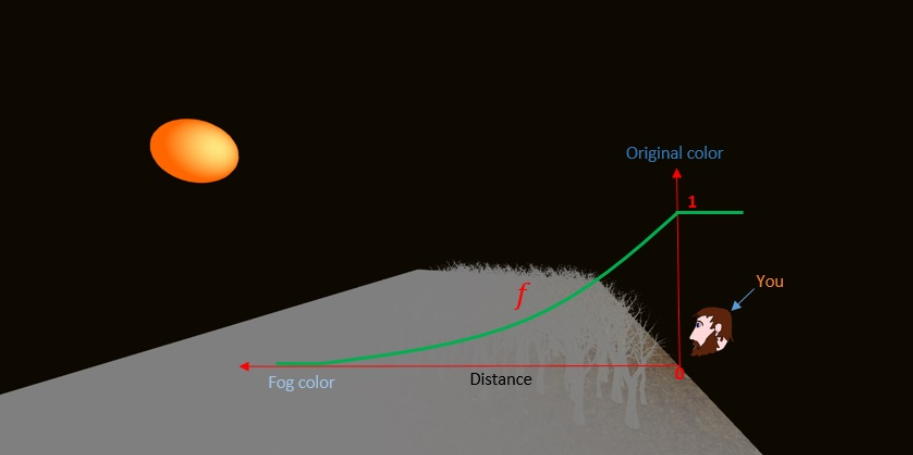
**Figura 4.2‑2** Implementación de Fog



**Figura 4.2‑3** Implementación de Fog



**Figura 4.2‑4** Implementación de fog lineal



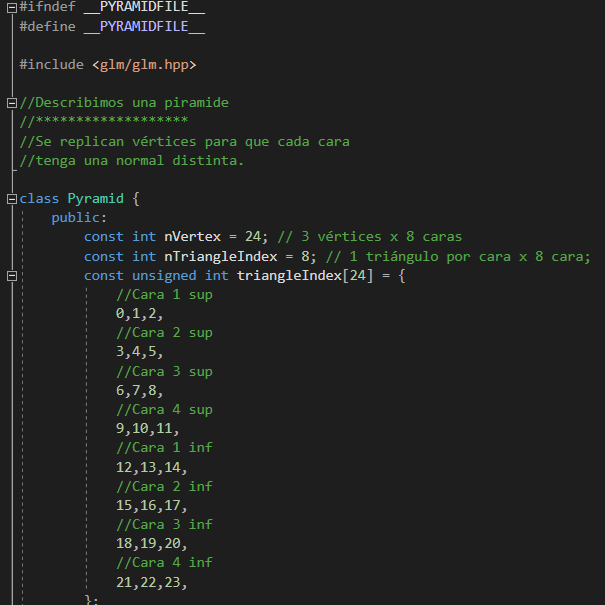
**Figura 4.2‑5** Implementación de fog exponencial

# Nuevo modelo.

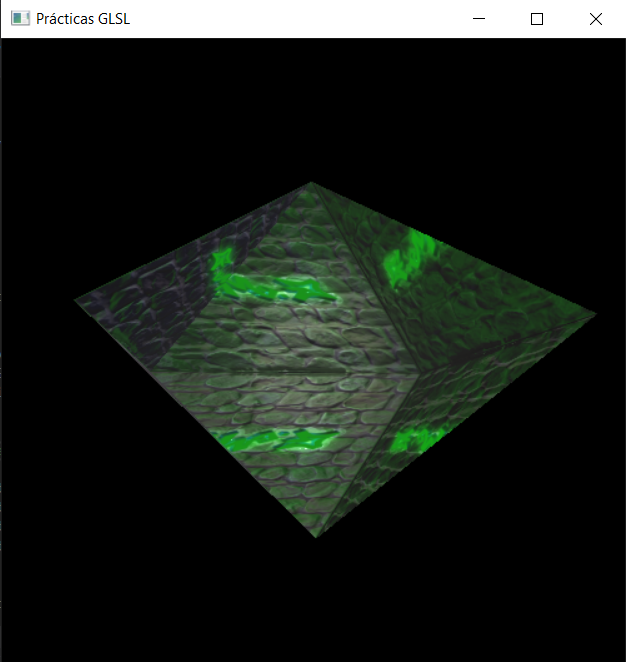
Finalmente, en este último punto de la actividad optativa, se pide generar un objeto nuevo. Para ello se definió una cabecera llamada “PYRAMID.hpp” con la cual se pasa toda la información necesaria para generación de la geometría de un octaedro. Se muestra a los fines de ejemplo solo una parte del archivo cabecera “PYRAMID.hpp”.

En Figura 4.3‑1, se muestra parte del código de cabecera “PYRAMID.hpp”, en el cual como ya se comentó se ha pasado la información necesaria de triángulos, vértices, normales, etc, para poder generar el objeto nuevo.

En Figura 4.3‑2, se observa el objeto creado.



**Figura 4.3‑1** Cabecera “PYRAMID.hpp”



**Figura 4.3‑2** Objeto nuevo creado

# Conclusiones.

Mediante la práctica se obtuvo manejo de los distintos tipos de iluminación y técnicas de sombreado aplicadas a un cauce gráfico real.