UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

EJERCICIOS DE CLASE Nº 08

NOMBRE COMPLETO: Barragán Pilar Diana

Nº de Cuenta: 318147981

GRUPO DE LABORATORIO: 03

GRUPO DE TEORÍA: 04

SEMESTRE 2025-1

FECHA DE ENTREGA LÍMITE: 09 de octubre de 2024

CALIFICACIÓN: _____

Ejercicio de clase práctica 8: Iluminación 2.

1. **Agregar su dado de 10 caras y editar sus normales para que las caras del dado sean iluminadas correctamente.**

Como describí anteriormente para implementar el dado de 10 caras simplemente volví a crear la pirámide pentagonal añadiendo sus respectivas texturas y en la parte de las normales las modifique para que la luz se viera correctamente, en todas las caras coloque la normal del eje y en negativo, es decir en -1 ya que siempre se va a buscar que la pirámide se ilumine al verla desde arriba y al rotar la pirámide también se ilumina si la vez desde abajo, para las primeras tres caras coloque la normal en z como negativa ya que estas las veremos más comúnmente desde el frente así que la luz debe ir en dirección negativa a z para que las caras se visualicen correctamente, por el contrario ya que las últimas dos caras se verán más desde atrás coloque el vector de z en 1. Para la normal de x, la primera y ultima cara tienen una asignación positiva ya que estas se verán mayormente desde el lado derecho, por otro lado, todas las demás caras se verán más del lado izquierdo y por ello la asignación es negativa.

```
GLfloat piramidepentagonal_vertices[] = {
    //X     Y      Z           S    T           NX     NY     NZ
    0.5f, -0.5f,  0.0f,       0.29f, 0.57f,    -1.0f,  -1.0f,   -1.0f,  // Vértice 0  //0        CARA 1
    0.15f, -0.5f,  0.47f,     0.04f, 0.57f,    -1.0f,  -1.0f,  -1.0f,  // Vértice 1  //1
    0.0f,  0.5f,  0.0f,       0.17f, 0.98f,    -1.0f,  -1.0f,  -1.0f,   // Vértice 5 (punta de la pirámide) //2

    0.15f, -0.5f,  0.47f,     0.63f, 0.57f,    1.0f, -1.0f, -1.0f,  // Vértice 1  //3           CARA 2
   -0.4f, -0.5f,  0.29f,      0.37f, 0.57f,    1.0f, -1.0f, -1.0f,  // Vértice 2  //4
    0.0f,  0.5f,  0.0f,       0.5f, 0.98f,     1.0f, -1.0f, -1.0f,   // Vértice 5 (punta de la pirámide) //5

   -0.4f, -0.5f,  0.29f,      0.95f, 0.57f,    1.0f,  -1.0f,  -1.0f,  // Vértice 2  //6          CARA 3
   -0.4f, -0.5f, -0.29f,      0.72f, 0.57f,    1.0f,  -1.0f,  -1.0f,  // Vértice 3   //7
   0.0f,  0.5f,  0.0f,        0.83f, 0.98f,    1.0f,  -1.0f,  -1.0f,   // Vértice 5 (punta de la pirámide) //8

   -0.4f, -0.5f, -0.29f,      0.29f, 0.07f,     1.0f,  -1.0f,  1.0f,  // Vértice 3   //9         CARA 4
    0.15f, -0.5f, -0.47f,     0.05f, 0.07f,     1.0f,  -1.0f,  1.0f,  // Vértice 4   //10
    0.0f,  0.5f,  0.0f,       0.165f, 0.43f,    1.0f,  -1.0f,  1.0f,   // Vértice 5 (punta de la pirámide) //11

    0.15f, -0.5f, -0.47f,     0.62f, 0.07f,     -1.0f,   -1.0f,   1.0f,  // Vértice 4   //12     CARA 5
    0.5f, -0.5f,  0.0f,       0.38f, 0.07f,     -1.0f,   -1.0f,   1.0f,  // Vértice 0  //13
    0.0f,  0.5f,  0.0f,       0.5f, 0.43f,      -1.0f,   -1.0f,   1.0f   // Vértice 5 (punta de la pirámide) //14
};

Mesh* piramide = new Mesh();
piramide->CreateMesh(piramidepentagonal_vertices, piramidepentagonal_indices, 120, 24);
meshList.push_back(piramide);
```

Figura 1. Asignación de normales en el dado de 10 caras.

2. **Apagar con teclado la luz (pointlight) de su lámpara creada para el reporte de la práctica 7.**

Para apagar la luz dado que en el escenario solo tengo una luz pointlight en el arreglo esta es tanto la primera como la ultima por ello en la parte de los shaders dentro del ciclo while puse una condicional en donde si se presiona una cierta tecla se manda al shader la luz puntual o no se manda, esto dejando de contar la luz en

el respectivo contador que tenemos de luz puntuales. Para la asignación de las teclas con las que se encenderá o apagará la lámpara utilice dos if en Windows.cpp asignando que al presionar la tecla K la luz se apaga esto asignándole a la variable pres un true, pero si se presiona la tecla L la luz se enciende ya que la variable pres pasará a ser false.

```
//información al shader de fuentes de iluminación
shaderList[0].SetDirectionalLight(&mainLight);
//Encendido y apagado de lampara
if (mainWindow.getpres()==true) {
    shaderList[0].SetPointLights(pointLights, pointLightCount-1);
}
else {
    shaderList[0].SetPointLights(pointLights, pointLightCount);
}

shaderList[0].SetSpotLights(spotLights, spotLightCount);
```

Figura 2. If en el ciclo while para encender y apagar la luz.

```
if (key == GLFW_KEY_K && action == GLFW_PRESS)
{
    theWindow->pres = true;
}
if (key == GLFW_KEY_L && action == GLFW_PRESS)
{
    theWindow->pres = false;
}
```

Figura 3. Asignación de teclas para que se encienda o apague la luz.

En este ejercicio no tuve inconveniente alguno a la hora de compilar y ejecutar el código.

CONCLUSIONES:

Con la implementación de ambos ejercicios asignados me fue posible entender de mejor forma como es que funciona la iluminación con respecto a objetos, pues al asignar las respectivas normales al dado de diez caras, me familiarice más con el uso de estas y como es que afecta la luz a la figura, por otro lado el comprender como es que se utilizan varias luces en una misma escena y que estas pueden estar tanto activadas como apagadas me ayudo ver mejor y con mayor profundidad la aplicación de los arreglos de luces.