UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

PREVIO Nº 5

NOMBRE COMPLETO: Gómez Enríquez Agustín

Nº de Cuenta: 317031405

GRUPO DE LABORATORIO: 3

GRUPO DE TEORÍA: 5

SEMESTRE 2026 - 1

FECHA DE ENTREGA LÍMITE: 19 de octubre del 2025

CALIFICACIÓN: _____

Respuestas previo 5:

1.- Explicar los valores de con, lin, exp cómo influyen en las luces puntuales y spotlight y demostrar con capturas de pantalla los diferentes fenómenos de la luz al variar dichos valores.

En OpenGL, los tres valores con, lin, y exp se utilizan para controlar la atenuación (o decaimiento) de la luz conforme se aleja del objeto. Estos parámetros permiten simular el comportamiento real de la luz, que pierde intensidad con la distancia. La ecuación general que OpenGL usa para la atenuación de una luz puntual o spotlight es:

$$I = \frac{1}{(con + lin \cdot d + exp \cdot d^2)}$$

donde:

I = intensidad final de la luz.

d = distancia entre la luz y el fragmento (píxel).

con = componente constante.

lin = componente lineal.

exp = componente cuadrática (exponencial).

- ***Efecto del parámetro con (constante)***

  Este valor mantiene una intensidad base que no depende de la distancia.

  Un valor alto (por ejemplo, con = 1.0) hace que la luz se perciba uniforme, sin importar la distancia.

  Un valor más bajo (con = 0.1) hace que la luz se desvanezca incluso en zonas cercanas.

- ***Efecto del parámetro lin (lineal)***

  Controla la disminución lineal con la distancia. Al aumentar su valor, la luz se atenúa de manera progresiva, haciendo que los objetos más alejados se oscurezcan.

- ***Efecto del parámetro exp (cuadrático o exponencial)***

  Este término genera un decaimiento más realista (inversamente proporcional al cuadrado de la distancia).

  Ideal para simular bombillas o faros, donde la luz cae drásticamente fuera del área principal.

EJEMPLOS:

SPOTLIGHT

```
275     spotLights[3] = SpotLight(
276         1.0f, 0.0f, 0.0f,
277         1.0f, 2.0f,          // intensidades: ambient, diffuse
278         -10.0f, 0.0f, 0.0f, // posición (un poco arriba y al frente)
279         0.0f, -5.0f, 0.0f, // dirección (hacia el piso/escena)
280         1.0f, 0.0f, 0.0f,    // atenuación: con=1, lin=0, exp=0 (prácticamente sin caída)
281         15.0f);               // edge (apertura del cono, grados)
282     spotLightCount++;
```

```
275     spotLights[3] = SpotLight(
276         1.0f, 0.0f, 0.0f,
277         1.0f, 2.0f,          // intensidades: ambient, diffuse
278         -10.0f, 0.0f, 0.0f, // posición (un poco arriba y al frente)
279         0.0f, -5.0f, 0.0f, // dirección (hacia el piso/escena)
280         0.0f, 1.0f, 0.0f,    // atenuación: con=0, lin=0, exp=0 (prácticamente se difumina)
281         15.0f);               // edge (apertura del cono, grados)
282     spotLightCount++;
283
```

```
275     spotLights[3] = SpotLight(
276         1.0f, 0.0f, 0.0f,
277         1.0f, 2.0f,          // intensidades: ambient, diffuse
278         -10.0f, 0.0f, 0.0f, // posición (un poco arriba y al frente)
279         0.0f, -5.0f, 0.0f, // dirección (hacia el piso/escena)
280         0.0f, 0.0f, 1.0f,    // atenuación: con=0, lin=0, exp=0 (prácticamente se elimina)
281         15.0f);               // edge (apertura del cono, grados)
282     spotLightCount++;
```

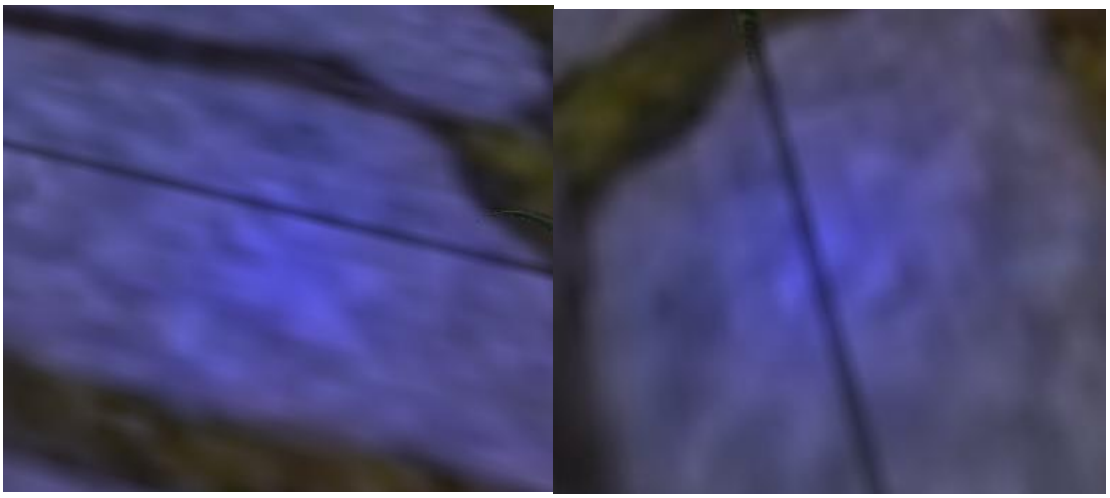
LUCES PUNTUALES

```
258     pointLights[1] = PointLight(0.0f, 0.0f, 1.0f,
259         0.2f, 0.3f,
260         0.0f, 0.0f, -7.0f,
261         1.0f, 0.0f, 0.0f);
262     pointLightCount++;
```

```
258     pointLights[1] = PointLight(0.0f, 0.0f, 1.0f,
259         0.2f, 0.3f,
260         0.0f, 0.0f, -7.0f,
261         0.0f, 1.0f, 0.0f);
262     pointLightCount++;
```

```
258     pointLights[1] = PointLight(0.0f, 0.0f, 1.0f,
259         0.2f, 0.3f,
260         0.0f, 0.0f, -7.0f,
261         0.0f, 0.0f, 1.0f);
262     pointLightCount++;
```

Prácticamente con los 3 cambios se percibe igual:



2.- ¿Cómo haces para que en tiempo de ejecución puedas elegir entre 2 arreglos de 4 luces puntuales en orden diferente, es decir: en el arreglo 1 tienes las luces verde, azul, roja, blanca y en el arreglo dos tienes a las luces blanca, verde, azul, roja?

Para cambiar entre dos conjuntos de luces en tiempo de ejecución, se puede crear dos arreglos de estructuras de luz y alternar entre ellos con una variable de control.

```
// Arreglo 1: Verde, Azul, Roja, Blanca
PointLight lightsSet1[4] = {
    PointLight(glm::vec3(0.0f, 2.0f, 0.0f), glm::vec3(0.0f, 1.0f, 0.0f)), // Verde
    PointLight(glm::vec3(2.0f, 2.0f, 0.0f), glm::vec3(0.0f, 0.0f, 1.0f)), // Azul
    PointLight(glm::vec3(-2.0f, 2.0f, 0.0f), glm::vec3(1.0f, 0.0f, 0.0f)), // Roja
    PointLight(glm::vec3(0.0f, 2.0f, 2.0f), glm::vec3(1.0f, 1.0f, 1.0f))  // Blanca
};
// Arreglo 2: Blanca, Verde, Azul, Roja
PointLight lightsSet2[4] = {
    PointLight(glm::vec3(0.0f, 2.0f, 0.0f), glm::vec3(1.0f, 1.0f, 1.0f)), // Blanca
    PointLight(glm::vec3(2.0f, 2.0f, 0.0f), glm::vec3(0.0f, 1.0f, 0.0f)), // Verde
    PointLight(glm::vec3(-2.0f, 2.0f, 0.0f), glm::vec3(0.0f, 0.0f, 1.0f)), // Azul
    PointLight(glm::vec3(0.0f, 2.0f, 2.0f), glm::vec3(1.0f, 0.0f, 0.0f))  // Roja
};
```

Y para seleccionar el arreglo:

```
int currentSet = 1; // 1 o 2
```

Para la ejecución

```
if (key == GLFW_KEY_L && action == GLFW_PRESS) {
    currentSet = (currentSet == 1) ? 2 : 1;}
```

Y dibujo:

```
if (currentSet == 1)
    shader.UsePointLights(lightsSet1, 4);
else
 shader.UsePointLights(lightsSet2, 4);
```

Referencias

- *Tutorial 8 : Shading básico*. (n.d.). Opengl-tutorial.org. Retrieved October 18, 2025, from https://www.opengl-tutorial.org/es/beginners-tutorials/tutorial-8-basic-shading/