UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

REPORTE DE PRÁCTICA Nº 08

NOMBRE COMPLETO: Arroyo Ramìrez Carlos Alberto

Nº de Cuenta: 320185865

GRUPO DE LABORATORIO: 03

GRUPO DE TEORÍA: 04

SEMESTRE 2026-1

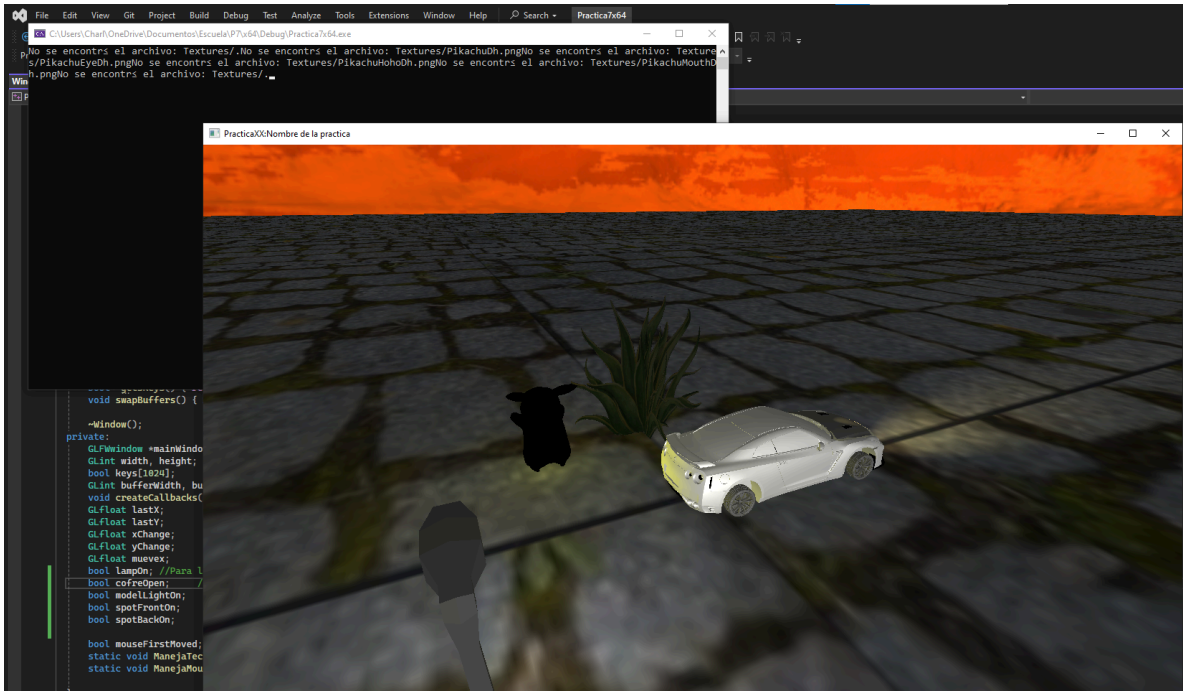
FECHA DE ENTREGA LÍMITE: 26/10/2025

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

1.- Ejecución de los ejercicios que se dejaron, comentar cada uno y capturas de pantalla de bloques de código generados y de ejecución del programa.

Para esta práctica volví a utilizar mi modelo 3d del carro, agregar un cofre y nuevamente agregar movilidad a esta, uni 3 luces en total, una para el cofre, para que al momento de abrirse o cerrarse apuntará en la dirección del cofre, por otro lado las otras 2 luces funcionan solo cuando el carro avanzaba o retrocedía. Al final inserte un modelo 3d que utilizare en mi proyecto y le agregue que con una tecla prendiera una luz, como si el personaje brillará, en este caso un Pikachu:



Para empezar en mi main tuve que agregar indices a las luces para poder controlar el encendido de estas:

```
int idxLampPL = -1; // lámpara (point light)
int idxModelPL = -1; // point light del modelo
int idxSpotFront = -1; // spotlight delantero del coche
int idxSpotBack = -1; // spotlight trasero del coche
int idxCofrePL = -1;
int idxCofreSpot = -1;
```

Cree diferentes luces para mis dos modelos:

```
//Luz Pikachu P (Prender)
idxModelPL = pointLightCount;  // guarda el índice
pointLights[pointLightCount] = PointLight(
    1.0f, 1.0f, 0.0f,   // color: amarillo
    0.1f, 1.0f,
    -3.0f, -1.0f, -3.0f,
    0.3f, 0.2f, 0.1f
);
pointLightCount++;
```

```
// Spotlight delantero
idxSpotFront = spotLightCount;
spotLights[spotLightCount] = SpotLight(
    1.0f, 1.0f, 0.0f,    // color amarillo
    0.0f, 2.0f,
    0.0f, 0.0f, 0.0f,
    0.0f, -1.0f, 0.0f,
    1.0f, 0.1f, 0.0f,
    15.0f
);
spotLightCount++;

// Spotlight trasero
idxSpotBack = spotLightCount;
spotLights[spotLightCount] = SpotLight(
    1.0f, 0.0f, 1.0f,    // color magenta
    0.0f, 2.0f,
    0.0f, 0.0f, 0.0f,
    0.0f, -1.0f, 0.0f,
    1.0f, 0.1f, 0.0f,
    15.0f
);
spotLightCount++;

// Spotlight del cofre
idxCofreSpot = spotLightCount;
spotLights[spotLightCount] = SpotLight(
    1.0f, 0.85f, 0.4f,
    0.0f, 2.0f,
    0.0f, 0.0f, 0.0f,
    0.0f, -1.0f, 0.0f,
    1.0f, 0.1f, 0.0f,
    20.0f
);
spotLightCount++;
```

Tuve que volver a generar la animación para el cofre, ajustar el modelo y anclar la luz Spotlight en el cofre del modelo:

```
//  Cofre
//
// Animación suave de apertura/cierre
float target = mainWindow.getCofreOpen() ? COFRE_MAX_ANGLE : 0.0f;
if (cofreAngleDeg < target) {
    cofreAngleDeg = glm::min(cofreAngleDeg + COFRE_SPEED_DPS * deltaTime, target);
}
else if (cofreAngleDeg > target) {
    cofreAngleDeg = glm::max(cofreAngleDeg - COFRE_SPEED_DPS * deltaTime, target);
}


model = modelaux;
model = glm::translate(model, glm::vec3(1.0f, 0.5f, 0.0f));  // posición base
model = glm::scale(model, glm::vec3(0.5f, 0.5f, 0.5f));
model = glm::rotate(model, -90.0f * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
model = glm::rotate(model, 360.0f * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::rotate(model, cofreAngleDeg * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::translate(model, glm::vec3(0.0f, 1.0f, 0.0f));  // posición final

glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Cofre.RenderModel();

// Spotlight montado en la tapa del cofre borde frontal
glm::vec4 cofreLocalPos(0.0f, 0.95f, -0.25f, 1.0f);
glm::vec3 cofreLocalDir(0.0f, -0.30f, -1.0f);


glm::vec3 cofreWorldPos = glm::vec3(model * cofreLocalPos);

glm::vec3 cofreWorldDir = glm::normalize(glm::mat3(model) * cofreLocalDir);

if (idxCofreSpot >= 0) {
    spotLights[idxCofreSpot].SetFlash(cofreWorldPos, cofreWorldDir);
}
```

Para establecer las luces delanteras y traseras de mi coche emplee este bloque de código:

```
float carBaseX = 0.0f + mainWindow.getmuevex();
float carBaseY = -0.8f;
float carBaseZ = 0.0f;
glm::vec3 carPos(carBaseX, carBaseY, carBaseZ);

float yawCar = 0.0f * toRadians; //Rotacion de mis faros
glm::vec3 forward(cosf(yawCar), 0.0f, sinf(yawCar));
glm::vec3 back = -forward;

float h = 0.4f;
float dFront = 1.8f;
float dBack = 1.6f;

glm::vec3 frontPos = carPos + glm::vec3(0.0f, h, 0.0f) + forward * dFront;
glm::vec3 backPos = carPos + glm::vec3(0.0f, h, 0.0f) + back * dBack;

glm::vec3 frontDir = glm::normalize(forward + glm::vec3(0.0f, -0.15f, 0.0f));
glm::vec3 backDir = glm::normalize(back + glm::vec3(0.0f, -0.15f, 0.0f));

if (idxSpotFront >= 0) {
    spotLights[idxSpotFront].SetFlash(frontPos, frontDir);
}
if (idxSpotBack >= 0) {
    spotLights[idxSpotBack].SetFlash(backPos, backDir);
}
```

Agregue condicionales para que funcionara que si al no presionar una tecla se apagará y si se presiona se prendiera:

```
// PointLights activas
PointLight activePL[MAX_POINT_LIGHTS];
unsigned int activePLCount = 0;

if (idxLampPL >= 0 && mainWindow.getLampOn()) {
    activePL[activePLCount++] = pointLights[idxLampPL];
}
if (idxModelPL >= 0 && mainWindow.getModelLightOn()) {
    activePL[activePLCount++] = pointLights[idxModelPL];
}

// SpotLights activas controladas por Y/U
bool* keys = mainWindow.getsKeys();

SpotLight activeSL[MAX_SPOT_LIGHTS];
unsigned int activeSLCount = 0;

if (idxSpotFront >= 0 && keys[GLFW_KEY_Y]) {
    activeSL[activeSLCount++] = spotLights[idxSpotFront];
}
if (idxSpotBack >= 0 && keys[GLFW_KEY_U]) {
    activeSL[activeSLCount++] = spotLights[idxSpotBack];
}
if (idxCofreSpot >= 0 ) {
    activeSL[activeSLCount++] = spotLights[idxCofreSpot];
}
```

También hice cambios en Window.h, agregue nuevos Getters:

```
bool getModelLightOn() const { return modelLightOn; } // luz puntual del modelo
bool getSpotFrontOn() const { return spotFrontOn; }   // spotlight delantero del coche
bool getSpotBackOn()  const { return spotBackOn; }    // spotlight trasero del coche
```

Y nuevos estados:

```
bool lampOn; //Para lampara
bool cofreOpen;     //para rotacion del cofre
bool modelLightOn;   // para la luz puntual del modelo elegido
bool spotFrontOn;    // spotlight delantero del coche
bool spotBackOn;     // spotlight trasero del coche
```

Para Window.cpp hice las siguientes modificaciones:

```
//Para abrir cofre: (H)
if (key == GLFW_KEY_H && action == GLFW_PRESS) {
    theWindow->cofreOpen = !theWindow->cofreOpen; // abre/cierra
}

//Para apagar luces: (L)
if (key == GLFW_KEY_L && action == GLFW_PRESS) {
    theWindow->lampOn = !theWindow->lampOn;
}

// Point light del Pikachu (P)
if (key == GLFW_KEY_P && action == GLFW_PRESS) {
    theWindow->modelLightOn = !theWindow->modelLightOn;
}
```

2.- Liste los problemas que tuvo a la hora de hacer estos ejercicios y si los resolvió explicar cómo fue, en caso de error adjuntar captura de pantalla

Tuve un problema con la importación del modelo 3D de Pikachu, al parecer cuando lo exportaba en obj, no cargaba la textura con él, me di cuenta que el error estuvo al exportar porque cuando lo importaba a blender aparecía violeta, es decir que tenía cargada una textura pero no sabía la ubicación de esta.

3.- Conclusión:

a. Los ejercicios del reporte: Complejidad, Explicación.

b. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias para mejorar desarrollo de la práctica
c. Conclusión

Esta práctica fue de las más largas y detalladas, ya que implicó coordinar varios elementos a la vez. Uno de los principales problemas fue que el modelo de Pikachu no cargó sus texturas al importar el archivo .obj, lo que hizo que apareciera completamente negro y tomara tiempo revisar la causa. También resultó complicado hacer que la luz del cofre siguiera correctamente el movimiento de la tapa, sobre todo por las transformaciones necesarias para que la dirección se actualizará de forma precisa al rotar. Además, el manejo de los índices y la activación de las distintas luces (puntuales y spotlights) requirió muchas pruebas hasta lograr que todo encendiera y se apagará correctamente. A pesar de eso, el resultado final fue bueno: las luces funcionan, las animaciones se ven naturales y el comportamiento general de la escena quedó completo.

1. Bibliografía en formato APA

Jo, J., & Lasseigne, T. (n.d.). Light casters. LearnOpenGL. https://learnopengl.com/Lighting/Light-casters

The GLFW Project. (n.d.). Input guide. GLFW. https://www.glfw.org/docs/3.3/input_guide.html

Lighthouse3D. (n.d.). GLSL Tutorial – Spotlights. https://www.lighthouse3d.com/tutorials/glsl-tutorial/spotlights/