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 03

NOMBRE COMPLETO: Jose Carlos Carranza Paula

Nº de Cuenta: 421073850

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 04

SEMESTRE 2026-1

FECHA DE ENTREGA LÍMITE: 07 de septiembre del 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.

1.- Generar una pirámide rubik (pyraminx) de 9 pirámides por cara.

Cada cara de la pyraminx que se vea de un color diferente y que se vean las separaciones entre instancias (las líneas oscuras son las que permiten diferenciar cada pirámide pequeña)

Agregar en su documento escrito las capturas de pantalla necesarias para que se vean las 4 caras de toda la

pyraminx o un video en el cual muestra las 4 caras

Cree mi pirámide por que la que venia estaba como deforme.

Utilizando rgb y el mesh Color en lugar del mesh.

Cree un vector para guardar los mesh color.

```
vector<MeshColor*> meshListColor;
//Vertex Shader
```

```
void CrearPiramideTriangular()
{
    GLfloat v[] = {
        // Base (gris)  y = -0.5
        -0.5f, -0.5f,  0.2887f,   0.6f, 0.6f, 0.6f,
         0.5f, -0.5f,  0.2887f,   0.6f, 0.6f, 0.6f,
         0.0f, -0.5f, -0.5774f,   0.6f, 0.6f, 0.6f,

        // Cara 1 (rojo): V0, V1, V3
        -0.5f, -0.5f,  0.2887f,   1.0f, 0.0f, 0.0f,
         0.5f, -0.5f,  0.2887f,   1.0f, 0.0f, 0.0f,
         0.0f,  0.5f,  0.0f,      1.0f, 0.0f, 0.0f,

        // Cara 2 (verde): V1, V2, V3
         0.5f, -0.5f,  0.2887f,   0.0f, 1.0f, 0.0f,
         0.0f, -0.5f, -0.5774f,   0.0f, 1.0f, 0.0f,
         0.0f,  0.5f,  0.0f,      0.0f, 1.0f, 0.0f,

        // Cara 3 (azul): V2, V0, V3
         0.0f, -0.5f, -0.5774f,   0.0f, 0.0f, 1.0f,
        -0.5f, -0.5f,  0.2887f,   0.0f, 0.0f, 1.0f,
         0.0f,  0.5f,  0.0f,      0.0f, 0.0f, 1.0f
    };
```

```
MeshColor* obj = new MeshColor();
obj->CreateMeshColor(v, 12 * 6);
meshListColor.push_back(obj);

GLfloat v2[] = {
    // Base (blanco)
    -0.5f, -0.5f,  0.2887f,   1.0f, 1.0f, 1.0f,
     0.5f, -0.5f,  0.2887f,   1.0f, 1.0f, 1.0f,
     0.0f, -0.5f, -0.5774f,   1.0f, 1.0f, 1.0f,

    // Cara 1 (blanco)
    -0.5f, -0.5f,  0.2887f,   1.0f, 1.0f, 1.0f,
     0.5f, -0.5f,  0.2887f,   1.0f, 1.0f, 1.0f,
     0.0f,  0.5f,  0.0f,      1.0f, 1.0f, 1.0f,

    // Cara 2 (blanco)
     0.5f, -0.5f,  0.2887f,   1.0f, 1.0f, 1.0f,
     0.0f, -0.5f, -0.5774f,   1.0f, 1.0f, 1.0f,
     0.0f,  0.5f,  0.0f,      1.0f, 1.0f, 1.0f,

    // Cara 3 (blanco)
     0.0f, -0.5f, -0.5774f,   1.0f, 1.0f, 1.0f,
    -0.5f, -0.5f,  0.2887f,   1.0f, 1.0f, 1.0f,
     0.0f,  0.5f,  0.0f,      1.0f, 1.0f, 1.0f
};
```

```
MeshColor* obj2 = new MeshColor();
obj2->CreateMeshColor(v2, 12 * 6);
meshListColor.push_back(obj2);
```

Use el shadercolor para poder usar el mesh Color.

```
void CreateShaders()
{

    Shader* shader2 = new Shader();
    shader2->CreateFromFiles(vShaderColor, fShader);
    shaderList.push_back(*shader2);
}
```

Y despues empeze a crear mis pirámides y modificandolas con scale translate y rotate. Y para verla use la cámara.

```
glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
//-----------------PIRÁMIDE 1-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, -1.0f, -6.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//-----------------PIRÁMIDE 2-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(1.0f, -1.0f, -6.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//-----------------PIRÁMIDE 3-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-1.0f, -1.0f, -6.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();
```

```
//-----------------PIRÁMIDE 4-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.5f, 0.0f, -6.2887f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//-----------------PIRÁMIDE 5-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-0.5f, 0.0f, -6.2887f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//-----------------PIRÁMIDE 6-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, 1.0f, -6.5774f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//-----------------PIRÁMIDE 7-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-0.5f, -1.0f, -6.8661f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();
```

```
//----------------PIRÁMIDE 8----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, -1.0f, -7.7322f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//----------------PIRÁMIDE 9----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.5f, -1.0f, -6.8661f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//----------------PIRÁMIDE 10----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-0.0f, 0.0f, -7.1548f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();
```

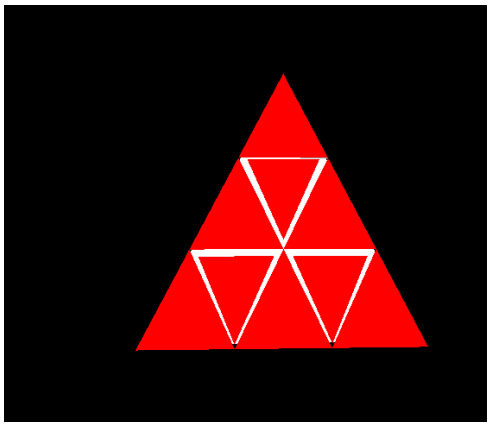
```
//----------------PIRÁMIDE 11----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.5f, -1.05f, -5.9f));
model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0, 0, 1));
model = glm::rotate(model, glm::radians(27.0f), glm::vec3(1, 0, 0));
model = glm::scale(model, glm::vec3(0.8f, 0.9f, 0.6f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//----------------PIRÁMIDE 12----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-0.5f, -1.05f, -5.9f));
model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0, 0, 1));
model = glm::rotate(model, glm::radians(27.0f), glm::vec3(1, 0, 0));
model = glm::scale(model, glm::vec3(0.8f, 0.9f, 0.6f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();
```

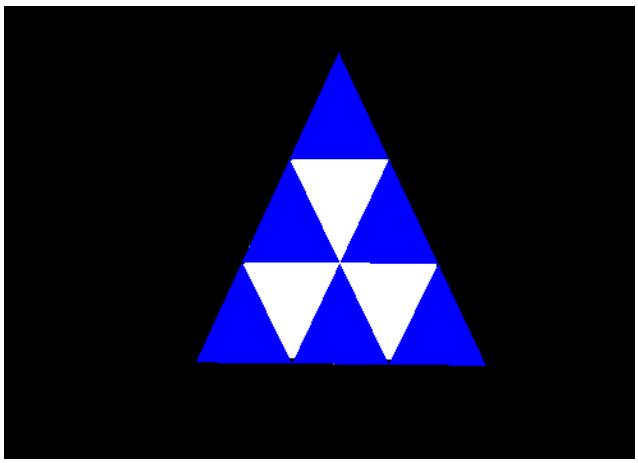
Y cree una piramide blanca tambien para crear las separaciones.

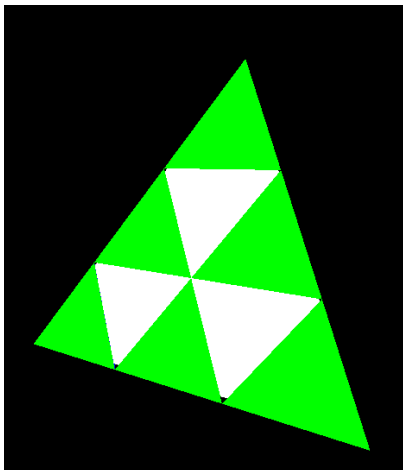
```
//-----------------PIRÁMIDE 13-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, 0.0f, -6.1887f));
model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0, 0, 1));
model = glm::rotate(model, glm::radians(27.0f), glm::vec3(1, 0, 0));
model = glm::scale(model, glm::vec3(0.8f, 0.9f, 0.6f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[0]->RenderMeshColor();

//-----------------PIRÁMIDE Blanca-----------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-0.0f, 0.0f, -6.5774f));
model = glm::scale(model, glm::vec3(2.9f, 2.9f, 2.9f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
meshListColor[1]->RenderMeshColor();
```

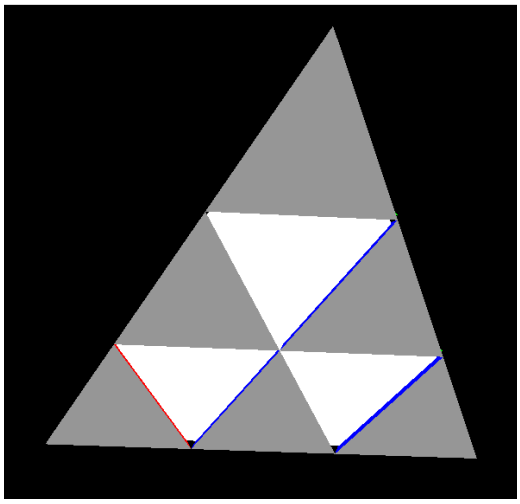
Aunque al final no logre incrustar todas las pirámides ni pintar las divisiones de la manera requerida.

Lo que pude hacer fue lo siguiente.









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

- Al cerrar la pestaña que abria el .exe no me cargo y ya no me dejaba ejecutar el codigo tuve que reiniciar la laptop

3.- Conclusión:

a. Los ejercicios del reporte: Complejidad, Explicación.
   Creo que fue un ejercicio difícil ya que tenía que crear mi propia pirámide ya que la que nos dio en clase estaba media deforme eso o me equivoque de figura.
b. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias para mejorar desarrollo de la práctica
   - Creo que falto explicar un poco como hacer los bordes estuvo dificil esa parte fue la que mas busque como hacer y al final

no pude hacerlo.También me costó darle ángulos a mi pirámide.

c. Conclusión
    - Me gusto el ejercicio aunque no pudiera resolverlo ya que me ayudó a comprender mejor qué cosas usar y que otras no pero sí me gustaría que nos dijera como se resuelve lo de la pirámide y supe usar mejor todo creo que fue un ejercicio que me ayudó a reforzar todos los temas visto hasta ahora en el laboratorio.

1. Bibliografía en formato APA

   No se utilizaron fuentes bibliográficas.