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 01

NOMBRE COMPLETO: Mino Guzmán Yara Amairani

Nº de Cuenta: 422017028

GRUPO DE LABORATORIO: 03

GRUPO DE TEORÍA: 04

SEMESTRE 2025-2

FECHA DE ENTREGA LÍMITE: 15 febrero 2025

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

1.- Actividades:

- Ventana cambia el color de fondo de forma random tomando rango de golores RGB y con una periodicidad de 2 segundos.

```
// Inclusión de bibliotecas necesarias para OpenGL, entrada/salida y manejo de tiempo
#include <stdio.h>      // Para entrada/salida estándar
#include <string.h>     // Para manipulación de cadenas
#include <stdlib.h>     // Para funciones generales como srand()
#include <time.h>       // Para obtener semilla de números aleatorios
#include <glew.h>       // Biblioteca de extensiones de OpenGL
#include <glfw3.h>      // Biblioteca para crear ventanas y manejar contextos OpenGL
```

Se agregó la librería time.h para obtener los colores aleatorios a partir de cierto tiempo que se le dé.

```
// Generar un color aleatorio diferente al último
do {
    colorChoice = rand() % 3;  // 0: Rojo, 1: Verde, 2: Azul
} while (colorChoice == lastColorChoice);

// Guardar la elección actual
lastColorChoice = colorChoice;

// Establecer color de fondo según la elección
switch (colorChoice) {
case 0: glClearColor(1.0f, 0.0f, 0.0f, 1.0f); break; // Rojo puro
case 1: glClearColor(0.0f, 1.0f, 0.0f, 1.0f); break; // Verde puro
case 2: glClearColor(0.0f, 0.0f, 1.0f, 1.0f); break; // Azul puro
}
```

Con el do-while, se generó la aleatoriedad de los colores utilizando rand y sacando un módulo de 3, ya que son 3 colores los que se necesitan mostrar de fondo sin que se repita un mismo patrón.
Se utilizaron los case para definir los colores a mostrarse (rojo, verde y azul).

```
// Variables para control de tiempo de cambio de color
double initialTime = glfwGetTime();
double nextColorChangeTime = initialTime + 2.0;
```

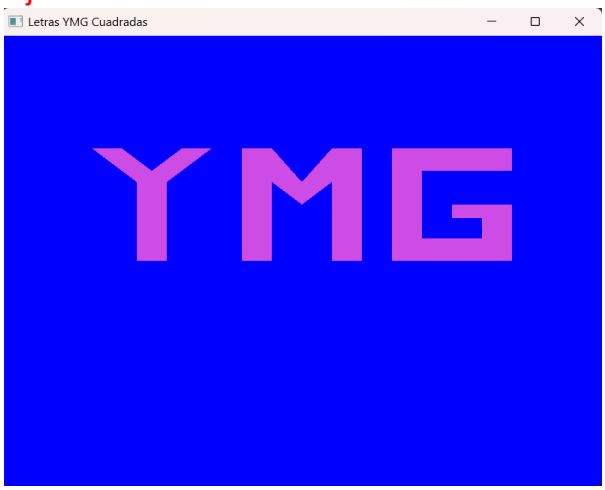
Aquí se va a controlar el tiempo para controlar el cambio de los colores.

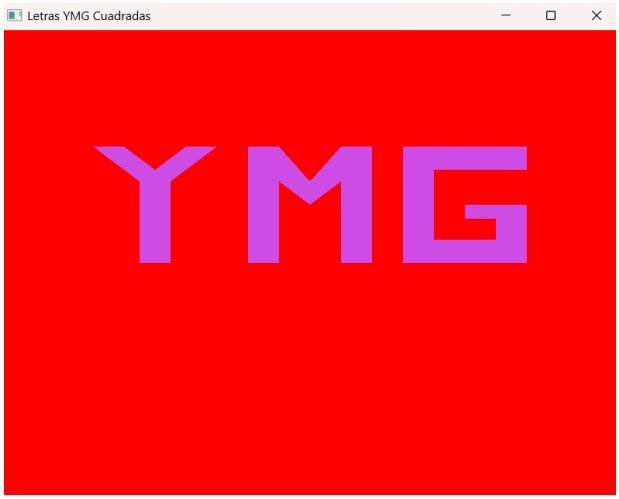
```
while (!glfwWindowShouldClose(mainWindow))
{
    // Obtener tiempo actual
    double currentTime = glfwGetTime();

    // Cambiar color cada 2 segundos
    if (currentTime >= nextColorChangeTime) {
        updateBackgroundColor();
        nextColorChangeTime = currentTime + 2.0;
    }
```

A partir del tiempo actual, se van a contar 2 segundos para que ocurra el cambio de colores en el fondo.

Ejecución:





- 3 letras iniciales de sus nombres creadas a partir de triángulos, todas las letras son del mismo color.

```
// Shader de fragmentos: define el color de las letras
const char* fShader = R"(
    #version 330                            // Versión de GLSL
    out vec4 color;                         // Color de salida
    void main()
    {
        color = vec4(0.8, 0.3, 0.9, 1.0);  // Color púrpura para las letras
    }
)";
```

Escogí que el color de las letras fuera púrpura.

```
// Función para definir la geometría de las letras YMG
void CrearLetras()
{
    // Definir un array de vértices para dibujar las letras
    // Cada trio de vértices define un triángulo
    // Las coordenadas van de -1.0 a 1.0 en el plano x,y
    GLfloat vertices[] = {
        // Y
        //Triangulo 1
        -0.7f,0.5f,0.0f,
        -0.6f,0.5f,0.0f,
        -0.6f,0.4f,0.0f,
        //Triangulo 2
        -0.6,0.5f,0.0f,
        -0.6,0.4f,0.0f,
        -0.5,0.4f,0.0f,
        //Triangulo 3
        -0.5,0.4,0.0f,
        -0.4,0.4f,0.0f,
        -0.4,0.5,0.0f,
        //Triangulo 4
        -0.4,0.4f,0.0f,
        -0.4,0.5,0.0f,
        -0.3,0.5f,0.0f,
        //Triangulo 5
        -0.6f,0.4f,0.0f,
        -0.4,0.4f,0.0f,
        -0.45,0.35f,0.0f,
        //Triangulo 6
        -0.6f,0.4f,0.0f,
        -0.45f,0.35f,0.0f,
```

En la función void CrearLetras, se crearon las letras utilizando triángulos que, a su vez formaban o trapecios, o paralelogramos o rectángulos, en total, se realizaron 25 triángulos.

```
// Crear ventana con dimensiones y título
GLFWwindow* mainWindow = glfwCreateWindow(WIDTH, HEIGHT, "Letras YMG Cuadradas", NULL, NULL);
if (!mainWindow)
{
    printf("Falló la creación de la ventana GLFW\n");
    glfwTerminate();
    return 1;
}
```

Se definió el título de la ventana, que, en este caso son las iniciales de mi nombre y se le dieron dimensiones.

```
// Dibujar las letras
glBindVertexArray(VAO);
glDrawArrays(GL_TRIANGLES, 0, 75);  // 75 vértices en total
glBindVertexArray(0);
glUseProgram(0);
```

Aquí se dibujan las letras y, de acuerdo con los triángulos que fueron realizados, es la cantidad de vértices que habrá. En mi caso, fueron 75 vértices.

Ejecución:

2.- Problemas a la hora de hacer las actividades solicitadas:

- Me costó trabajo el ir yendo paso a paso buscando las coordenadas correctas para que mis letras se vieran cuadradas, ya que en un inicio todas me quedaban triangulares debido a que no me quedaban las figuras geométricas necesarias para que las letras se mostrasen cuadradas.

3.- Conclusión:

a. Los ejercicios del reporte fueron relativamente sencillos, ya que para el uso de colores aleatorios cada cierto tiempo era algo que ya se había visto anteriormente solo que con números en vez de colores y, para crear las letras de mi nombre si fue más complicado debido a que

me fallaban las coordenadas, pero, una vez que quedaron, entendí cómo se tienen que ir buscando las coordenadas para que encajaran y lograran formarse.

b. Siento que faltó explicar un poco más a detalle la práctica, pero en general la práctica estuvo bien.
c. En conclusión, me gustó realizar esta práctica ya que exploté más mi capacidad para poder crear letras de mi nombre y que el fondo fuera cambiando sin que siguieran un patrón los colores.

Bibliografía

- generating random colored squares using opengl. (s. f.). Stack Overflow. https://stackoverflow.com/questions/7856795/generating-random-colored-squares-using-opengl
- Aspectos básicos de WebGL. (2012, 9 febrero). web.dev. https://web.dev/articles/webgl-fundamentals?hl=es-419
- CODIGO DE COLORES. (2023, 21 abril). Código de colores OpenGl Graficos 2D y 3D Ejemplo de implementacion. https://codigodecolor.com/codigo-de-colores-opengl/