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: Tapia Ledesma Angel Hazel

Nº de Cuenta: 320070358

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 02

SEMESTRE 2026-1

FECHA DE ENTREGA LÍMITE: 24-08-25

CALIFICACIÓN: _____

Actividades realizadas

1.- Ventana cambia el color de fondo de forma random tomando rango de colores RGB y con una periodicidad de 2 segundos. (Verificar que al ejecutar el programa varias veces el orden de los colores si lo vean aleatorio y no siempre los mismos)
2.- 3 letras iniciales de sus nombres creadas a partir de triángulos, todas las letras son del mismo color.
Los dos ejercicios se muestran de forma simultánea y están en el mismo main

Descripción de las actividades

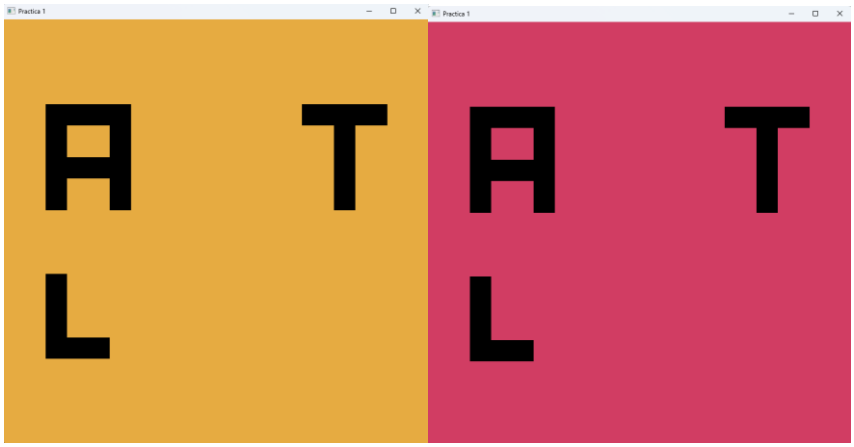
Para el caso de la primer actividad, lo que se hizo fue asignar 3 variables destinadas a almacenar los colores RGB, en este caso "red, green y blue", lo que se hizo después fue importar la biblioteca de "chronos" y "thread" para manejar el tiempo, y "iostream" y "random" para generar números aleatorios para cada variable, con un rango de 0-1 para así obtener un color aleatorio en cada ciclo.
Se crearon las variables para inicializar tanto los colores como para manejar los relojes, y poner el delay que se pide en la practica (2 segundos).
Finalmente, se trabajó la parte de la lógica, la cual consiste en un if, el cual valida si ha pasado el delay o no, en caso de que si, lo que se va a hacer es generar un número aleatorio entre 0 y 1 para cada color del RGB, obteniendo así algún color aleatorio entre la gama de colores, para terminar, se actualizan las variables de tiempo para que en el siguiente ciclo se realice la verificación del delay de nuevo, esto lo hacemos actualizando la variable "lastTime" con "currentTime" para que el siguiente ciclo se compare con ese tiempo con el actual y se siga con la lógica sin errores.

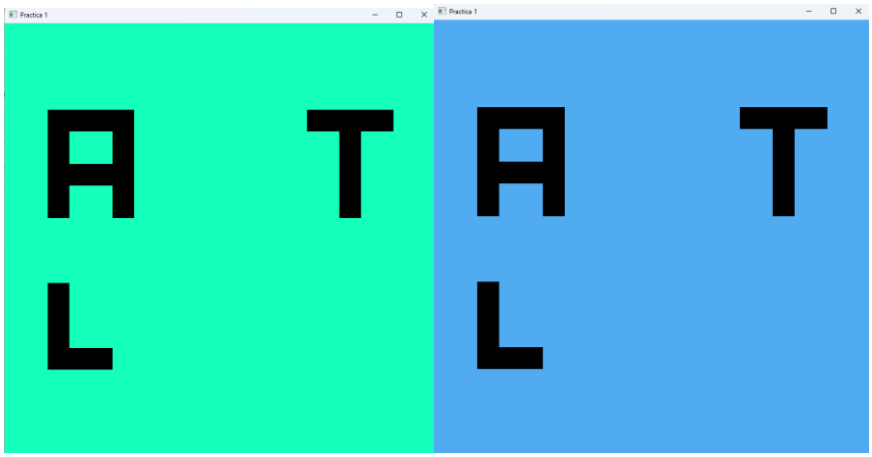
Código generado:

```
#include <stdio.h>
#include <string.h>
#include <glew.h>
#include <glfw3.h>
#include <chrono>
#include <thread>
#include <iostream>
#include <random>
//Dimensiones de la ventana
const int WIDTH = 800, HEIGHT = 800;
GLuint VAO, VBO, shader;

// Variable de semilla aleatoria basade en hardware
std::random_device rd;
std::mt19937 gen(rd());
std::uniform_real_distribution<> distrib(0.0, 1.0);

//variables de color
float red = distrib(gen), green= distrib(gen), blue= distrib(gen);

// Variables para el tiempo
const float delay = 2.0;
auto lastTime = std::chrono::high_resolution_clock::now();
```

```
//Loop mientras no se cierra la ventana
while (!glfwWindowShouldClose(mainWindow))
{
    //Recibir eventos del usuario
    glfwPollEvents();
    // Lógica de color aleatorio
    auto currentTime = std::chrono::high_resolution_clock::now();
    std::chrono::duration<double> elapsedTime = currentTime - lastTime;

    if (elapsedTime.count() >= delay) {
        red = distrib(gen);
        green = distrib(gen);
        blue = distrib(gen);
        lastTime = currentTime;
    }
    //Limpiar la ventana
    glClearColor(red,green,blue,1.0f);
    glClear(GL_COLOR_BUFFER_BIT);

    glUseProgram(shader);

    glBindVertexArray(VAO);
    glDrawArrays(GL_TRIANGLES,0,48);
    glBindVertexArray(0);

    glUseProgram(0);

    glfwSwapBuffers(mainWindow);

    //NO ESCRIBIR NINGUNA LÍNEA DESPUÉS DE glfwSwapBuffers(mainWindow);
}
```

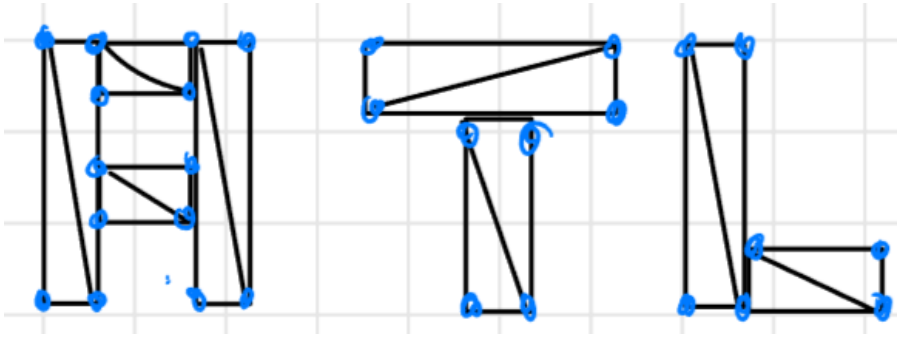
Ejecución





Para el 2do punto, lo que se hizo fue establecer el color de las letras, en este caso de negro para que no se pierdan con el RGB, posteriormente lo que se realizó fue un análisis de cómo generar las letras a partir de triángulos, para esto, primero se hizo la letra y despues se fue partiendo con triángulos, este paso fue importante, ya que me permitió además de saber como trazar los triangulos, conocer cuantos eran y de esta manera poner los vertices necesarios en la funcion que genera el dibujo. Finalmente, se eligió una coordenada donde poner cada letra y apartir de ahí empezar a construirlas con triángulos, para finalmente ejecutar el programa y corroborar que salieran.

Esbozo



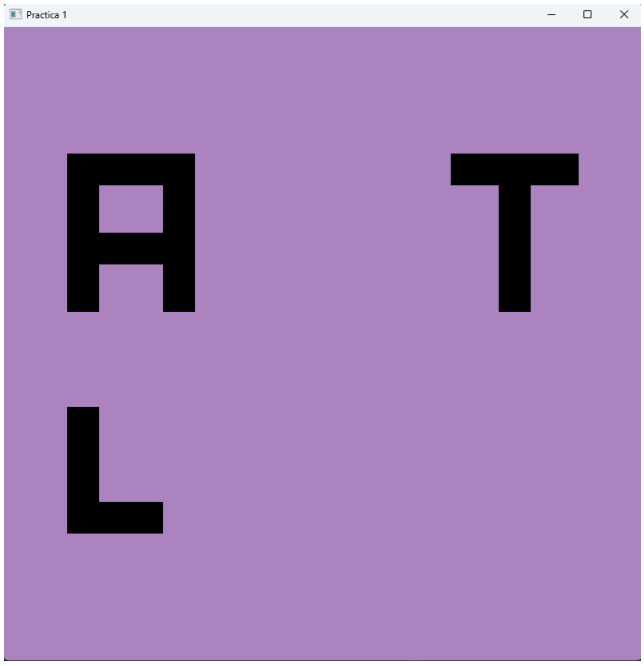
Código generado

```
//Fragment Shader
//recibir Vcolor y dar de salida color
static const char* fShader = "                  \n\
#version 330                                    \n\
out vec4 color;                                 \n\
void main()                                     \n\
{                                               \n\
    color = vec4(0.0f,0.0f,0.0f,1.0f);          \n\
}";
```

```
void CrearTriangulo()
{
    GLfloat vertices[] = {
        // A
        -0.7f, 0.6f,0.0f,
        -0.8f,0.6f, 0.0f,
        -0.8f,0.1f,0.0f,
        -0.8f,0.1f,0.0f,
        -0.7f,0.1f,0.0f,
        -0.7f, 0.6f,0.0f,
        -0.5f, 0.35f,0.0f,
        -0.7f, 0.35f,0.0f,
        -0.7f, 0.25f,0.0f,
        -0.5f, 0.35f,0.0f,
        -0.7f, 0.25f,0.0f,
        -0.5f, 0.25f,0.0f,
        -0.5f, 0.1f,0.0f,
        -0.4f, 0.1f,0.0f,
        -0.4f, 0.6f,0.0f,
        -0.4f, 0.6f,0.0f,
        -0.5f, 0.6f,0.0f,
        -0.5f, 0.1f,0.0f,
        -0.5f, 0.6f,0.0f,
        -0.7f, 0.6f,0.0f,
        -0.7f, 0.5f,0.0f,
        -0.7f, 0.5f,0.0f,
        -0.5f, 0.5f,0.0f,
        -0.5f, 0.6f,0.0f,
        // T
        0.8f, 0.6f,0.0f,
        0.4f, 0.6f,0.0f,
        0.4f, 0.5f,0.0f,
        0.4f, 0.5f,0.0f,
        0.8f, 0.5f,0.0f,
        0.8f, 0.6f,0.0f,
        0.55f, 0.5f,0.0f,
        0.55f, 0.1f,0.0f,
        0.65f, 0.1f,0.0f,
        0.55f, 0.5f,0.0f,
        0.65f, 0.5f,0.0f,
        0.65f, 0.1f,0.0f,
        // L
        -0.8f, -0.2f,0.0f,
        -0.8f, -0.6f,0.0f,
        -0.7f, -0.6f,0.0f,
        -0.8f, -0.2f,0.0f,
        -0.7f, -0.2f,0.0f,
        -0.7f, -0.6f,0.0f,
        -0.5f, -0.5f,0.0f,
        -0.7f, -0.5f,0.0f,
```

```
        -0.7f, -0.6f,0.0f,
        -0.5f, -0.5f,0.0f,
        -0.7f, -0.5f,0.0f,
        -0.7f, -0.6f,0.0f,
        -0.5f, -0.5f,0.0f,
        -0.5f, -0.6f,0.0f,
        -0.7f, -0.6f,0.0f,
    };
```

Ejecución:



Problemas presentados

En el caso de la práctica, no hubo ningún problema, con el ejercicio realizado en clase, se pudo tener una buena aproximación, ya que era algo similar en cuanto a lo del color, solo que aquí era hacerlo aleatorio, y la mayor dificultad es en el punto 2 en el cual había que construir las letras, sin embargo, realizar el esbozo ayudó mucho a facilitar esto, lo único tardado fue ir checando en que coordenadas poner los puntos correctamente para que estuvieran del mismo tamaño, alineados y no saliera algo raro.

Conclusión:

Esta práctica fue bastante buena para reforzar los conocimientos obtenidos en el laboratorio con respecto a cómo dibujar las figuras con triángulos en este caso, y comprender más sobre los vértices y cómo manipular en general el código para hacer la parte gráfica. La práctica fue bastante clara con lo que había que hacer, sin embargo, el único comentario al respecto que tengo fue que en la descripción del problema 1 se decía 2 segundos de delay, pero en la rúbrica de la práctica dice 3 segundos, entonces ahí si hay una inconsistencia, pero esta práctica se realizó con 2 segundos. En cuanto a la dificultad, siento que estuvo bien, el haber realizado el ejercicio 1 ma ayudó mucho a que se me facilitara poder realizar la práctica.
Creo que fue una práctica bastante completa, clara, con un nivel bueno para lo que se vió en lab y entretenida para poder repasar y reforzar los conceptos vistos, además de que el trabajar en poner todos los vertice, hace que nos vayamos acostumbrando a las coordenadas bajo las cuales se está trabajando.

Bibliografía

- Geeksforgeeks.(Enero, 2023). “Chrono in c++”. Recuperado el 20 de agosto de 2025 de: https://www.geeksforgeeks.org/cpp/chrono-in-c/