UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

EJERCICIOS DE CLASE Nº 01

NOMBRE COMPLETO: Aguilar Pérez José Ramón

Nº de Cuenta: 317515048

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 04

SEMESTRE 2025-2

FECHA DE ENTREGA LÍMITE: 11/febrero/2025

CALIFICACIÓN: _____

EJERCICIOS DE SESIÓN:

1. **Actividades realizadas.**

- **Cambiar el color de fondo de la pantalla entre rojo, verde y azul de forma cíclica y solamente mostrando esos 3 colores con un periodo de lapso adecuado para el ojo humano.**

Para realizar este cambio, se añadió la librería *Windows.h,* la cual permite utilizar la función *sleep(),* esto con el fin de que el fondo cambie cada 1.2 segundos para que sea más amigable a la vista. Después, se inicializa en 0 a la variable *fondo,* esta variable servirá como bandera en una serie de condicionales.

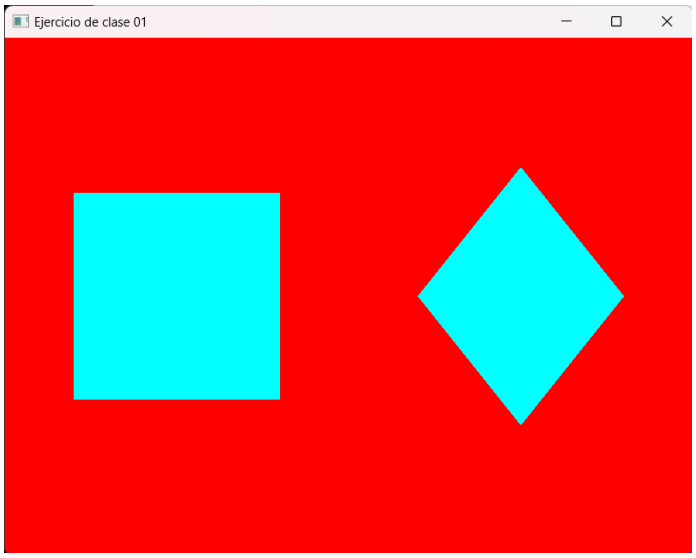
```
#include <glfw3.h>
#include <Windows.h>
```

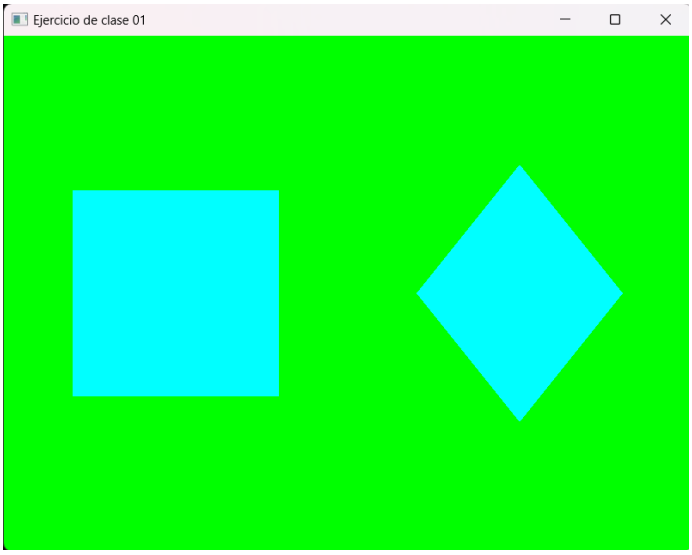
```
int fondo=0;
```

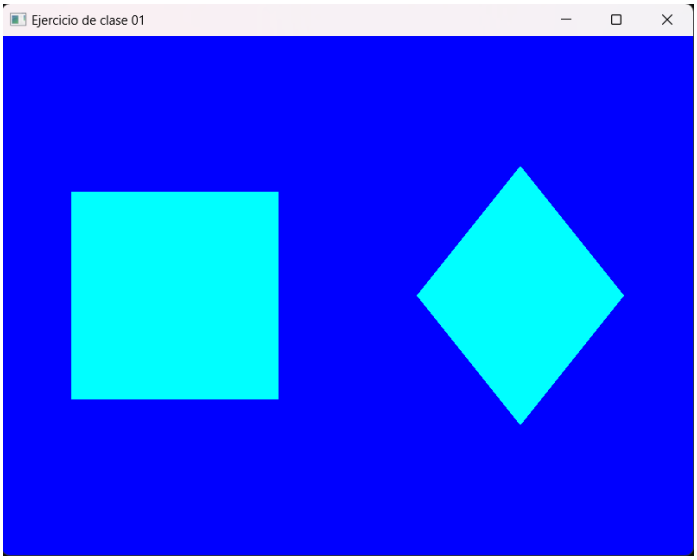
Si *fondo=0,* el fondo será rojo y se aumenta una unidad a la variable bandera. Si *fondo=1,* el fondo será verde y se aumenta otra unidad a la variable bandera. Por último, si *fondo=2,* el fondo será azul y se volverá al valor original de la bandera (es decir, cero) para que se repita el ciclo de colores del fondo.

```
while (!glfwWindowShouldClose(mainWindow))
{
    //Recibir eventos del usuario
    glfwPollEvents();

    //Cambio del fondo a RGB
    if (fondo == 0) {
        glClearColor(1.0f, 0.0f, 0.0f, 1.0f);
        glClear(GL_COLOR_BUFFER_BIT);
        Sleep(1200);
        fondo++;
    }
    else if (fondo == 1) {
        glClearColor(0.0f, 1.0f, 0.0f, 1.0f);
        glClear(GL_COLOR_BUFFER_BIT);
        Sleep(1200);
        fondo++;
    }
    else if (fondo == 2) {
        glClearColor(0.0f, 0.0f, 1.0f, 1.0f);
        glClear(GL_COLOR_BUFFER_BIT);
        Sleep(1200);
        fondo=0;
    }
```

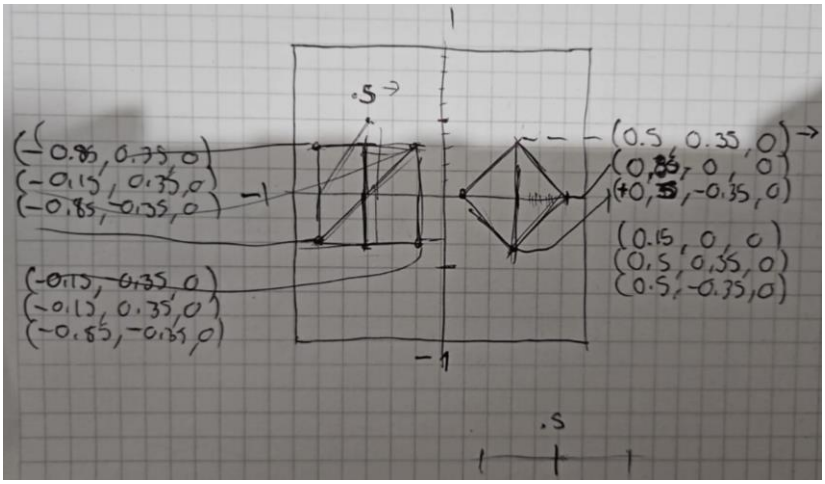

Ejercicio de clase 01


Ejercicio de clase 01


Ejercicio de clase 01

- **Dibujar de forma simultánea en la ventana 1 cuadrado y 1 rombo separados.**

Se hizo un boceto inicial de la distribución general de los triángulos para formar las figuras solicitadas. Como se necesitan dos triángulos por figura (4 en total), se establecieron los 12 vectores a trabajar en el área de trabajo que abarca de [-1,1] en el eje X y [-1,1] en el eje Y. Algunos valores de los vértices fueron modificados en el código por motivos estéticos, porque a la hora de generar las figuras quedaban un poco asimétricas.




```
GLfloat vertices[] = {
    //Cuadrado
    -0.8f,0.4f,0.0f,
    -0.2f,0.4f,0.0f,
    -0.8f,-0.4f,0.0f,
    -0.2f,-0.4f,0.0f,
    -0.2f,0.4f,0.0f,
    -0.8f,-0.4f,0.0f,

    //Rombo
    0.5f,0.5f,0.0f,
    0.8f,0.0f,0.0f,
    0.5f,-0.5f,0.0f,
    0.2f,0.0f,0.0f,
    0.5f,0.5f,0.0f,
    0.5f,-0.5f,0.0f
};
```

Como se van a trabajar con 12 vértices, fue necesario modificar este valor en la función *glDrawArrays()* para que se pudieran generar los 4 triángulos que forman las figuras.

```
glBindVertexArray(VAO);
glDrawArrays(GL_TRIANGLES,0,12);
glBindVertexArray(0);
```

Finalmente, se cambió el color de las figuras a un turquesa, esto con el fin de evitar que se pierdan las figuras a la hora de que el fondo cambie de color.

```
//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,163.0f,163.0f,1.0f);                  \n\
}";
```

2. Problemas presentados.

No se presentaron problemas a la hora de realizar el ejercicio.

3. Conclusión

a. Los ejercicios de la clase: Complejidad, explicación

Los ejercicios solicitados se me hicieron de una complejidad razonable, ya que, aunque sea el primer acercamiento a la computación gráfica, gracias a los conocimientos adquiridos con anterioridad sobre el lenguaje de programación C/C++ se pudieron aplicar las modificaciones necesarias sin problema alguno.

Además, con este ejercicio empecé a tener una noción más clara de algunas de las funciones de las librerías GLFW3 y GLEW.

b. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias.

La explicación general del funcionamiento del programa me pareció buena, ya que fue directo al grano sobre como realizar los triángulos, así como también explicando de manera breve pero precisa el funcionamiento de varas funciones nuevas en este nuevo entorno de computación gráfica.