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: RENDÓN HERNÁNDEZ ROBERTO CARLOS

Nº de Cuenta: 420052603

GRUPO DE LABORATORIO: 01

GRUPO DE TEORÍA: 04

SEMESTRE 2024-2

FECHA DE ENTREGA LÍMITE: 17/02/2024

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

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

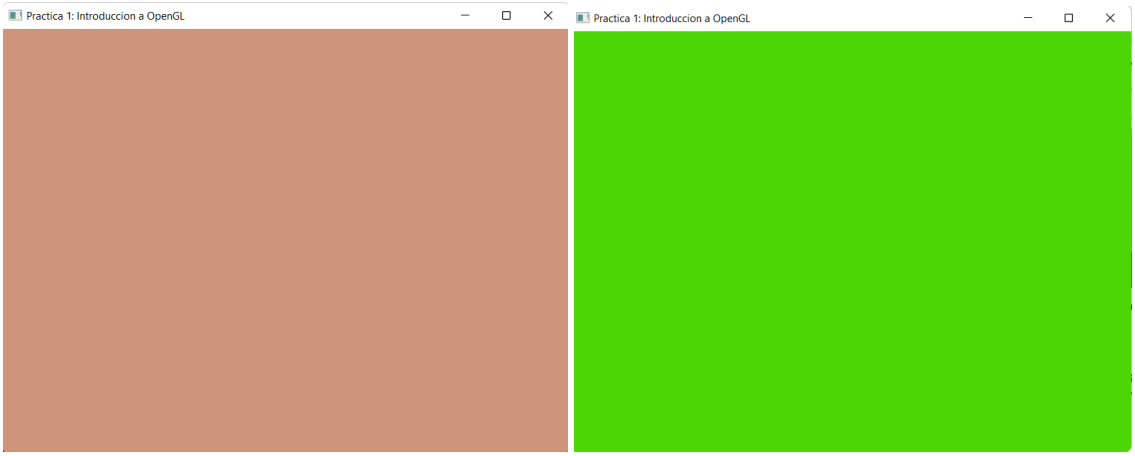
Se generan tres números aleatorios r, g y b entre 0 y 1 utilizando la función rand(), se divide por RAND_MAX para normalizar los valores entre 0 y 1 y este dentro de la gama RGB. Estos valores se utilizan como componentes de color para establecer el color de fondo de OpenGL utilizando glClearColor().

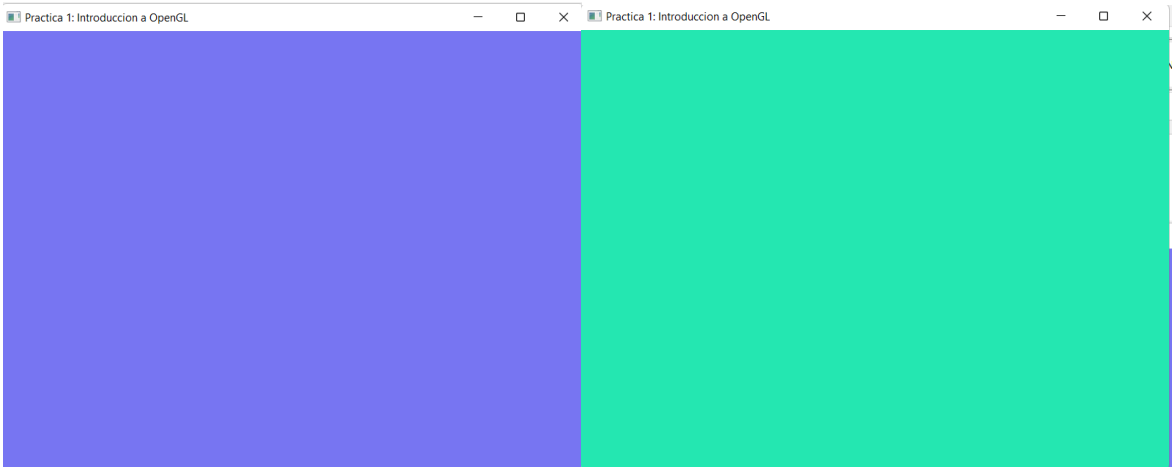
```
//Generaramos números aleatorios entre 0 y 1
double r= (double)rand() / RAND_MAX;
double g = (double)rand() / RAND_MAX;
double b = (double)rand() / RAND_MAX;

glClearColor(r, g, b, 1.0f);
```

Ahora para tener una periodicidad de 3 segundos agregamos la función Sleep():

```
Sleep(2000); //Esperar 2 segundos
```

Ejecución del código:





2.- 3 letras iniciales de sus nombres creadas a partir de triángulos, todas las letras son del mismo color.

Se realizará a partir de triángulos las iniciales RCH de mi nombre para ello definiremos los vértices que forman las letras "R", "C" y "H" utilizando triángulos. Estos vértices están definidos en coordenadas tridimensionales (x, y, z), donde la coordenada z es siempre 0 ya que estamos trabajando en el plano XY.

Cada letra está dividida en triángulos que forman su estructura. Por ejemplo, la letra "R" se divide en 10 triángulos, la letra "C" en 8 triángulos y la letra "H" en 6 triángulos.

En la función CrearTriangulo() agregamos nuestras coordenada de los vértices a nuestra variable vértices:

```
// Triangulos para la letra R
//triangulo 1
-0.8f, 0.5f, 0.0f,
-0.8f, 0.4f, 0.0f,
-0.6f, 0.4f, 0.0f,
//triangulo 2
-0.8f, 0.5f, 0.0f,
-0.6f, 0.5f, 0.0f,
-0.6f, 0.4f, 0.0f,
//triangulo 3
-0.8f, 0.4f, 0.0f,
-0.8f, 0.0f, 0.0f,
-0.7f, 0.0f, 0.0f,
//triangulo 4
-0.8f, 0.4f, 0.0f,
-0.7f, 0.4f, 0.0f,
-0.7f, 0.0f, 0.0f,
//triangulo 5
-0.7f, 0.3f, 0.0f,
-0.7f, 0.2f, 0.0f,
-0.6f, 0.2f, 0.0f,
//triangulo 6
-0.7f, 0.3f, 0.0f,
-0.6f, 0.3f, 0.0f,
-0.6f, 0.2f, 0.0f,
//triangulo 7
-0.6f, 0.2f, 0.0f,
-0.6f, 0.5f, 0.0f,
-0.5f, 0.4f, 0.0f,
//triangulo 8
-0.6f, 0.2f, 0.0f,
-0.5f, 0.3f, 0.0f,
-0.5f, 0.4f, 0.0f,
//triangulo 9
-0.6f, 0.0f, 0.0f,
-0.7f, 0.2f, 0.0f,
-0.6f, 0.2f, 0.0f,
//triangulo 10
-0.6f, 0.0f, 0.0f,
-0.5f, 0.0f, 0.0f,
-0.6f, 0.2f, 0.0f,
```

```
// Triangulos para la letra C
//triangulo 1
-0.3f, 0.0f, 0.0f,
-0.4f, 0.4f, 0.0f,
-0.4f, 0.1f, 0.0f,
//triangulo 2
-0.3f, 0.5f, 0.0f,
-0.4f, 0.4f, 0.0f,
-0.3f, 0.0f, 0.0f,
//triangulo 3
-0.3f, 0.5f, 0.0f,
-0.3f, 0.3f, 0.0f,
-0.25f, 0.4f, 0.0f,
//triangulo 4
-0.3f, 0.5f, 0.0f,
-0.2f, 0.5f, 0.0f,
-0.2f, 0.3f, 0.0f,
//triangulo 5
-0.1f, 0.4f, 0.0f,
-0.2f, 0.5f, 0.0f,
-0.2f, 0.3f, 0.0f,
//triangulo 6
-0.3f, 0.0f, 0.0f,
-0.3f, 0.2f, 0.0f,
-0.25f, 0.1f, 0.0f,
//triangulo 7
-0.3f, 0.0f, 0.0f,
-0.2f, 0.0f, 0.0f,
-0.2f, 0.2f, 0.0f,
//triangulo 8
-0.1f, 0.1f, 0.0f,
-0.2f, 0.0f, 0.0f,
-0.2f, 0.2f, 0.0f,
```

```
// Triangulos para la letra H
//triangulo 1
0.0f, 0.0f, 0.0f,
0.0f, 0.5f, 0.0f,
0.1f, 0.0f, 0.0f,
//triangulo 2
0.1f, 0.5f, 0.0f,
0.0f, 0.5f, 0.0f,
0.1f, 0.0f, 0.0f,
//triangulo 3
0.1f, 0.2f, 0.0f,
0.2f, 0.2f, 0.0f,
0.1f, 0.3f, 0.0f,
//triangulo 4
0.2f, 0.3f, 0.0f,
0.2f, 0.2f, 0.0f,
0.1f, 0.3f, 0.0f,
//triangulo 5
0.2f, 0.5f, 0.0f,
0.3f, 0.5f, 0.0f,
0.2f, 0.0f, 0.0f,
//triangulo 6
0.3f, 0.0f, 0.0f,
0.3f, 0.5f, 0.0f,
0.2f, 0.0f, 0.0f,
```

Por ultimo modificamos la función glDrawArrays(); agregando los vértices necesarios para las iniciales:

```
glDrawArrays(GL_TRIANGLES, 0, 112);
```

Ejecución del código:

Conclusión

El ejercicio de dibujar las iniciales "R", "C" y "H" proporcionó una comprensión de la importancia de definir vértices con precisión y configurar programas de sombreado para renderizar formas tridimensionales. Se demostró cómo los triángulos pueden combinarse para formar formas complejas, dando una base en el proceso de renderizado en OpenGL.

Por otro lado, el ejercicio de cambiar el color de fondo aleatoriamente entre los colores RGB introdujo la manipulación dinámica de gráficos. Al generar valores aleatorios para los componentes de color y aplicarlos al fondo, se exploró cómo ajustar la apariencia visual en tiempo real, lo que brinda una visión práctica de la personalización visual dinámica en OpenGL.

Bibliografía

cplusplus.com. (s. f.). rand - C++ Reference. Recuperado el 17 de febrero de 2024, de https://cplusplus.com/reference/cstdlib/rand/

GlGenVertexArrays and GlGenBuffers Arguments. (s. f.). Stack Overflow. . Recuperado el 17 de febrero de 2024 de https://stackoverflow.com/questions/45860198/glgenvertexarrays-and-glgenbuffers-argument