
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:** Vázquez Gómez Carlos Iván

**Nº de Cuenta: 4200551851**

**GRUPO DE LABORATORIO:** 03

**GRUPO DE TEORÍA: 04**

**SEMESTRE 2025-1**

**FECHA DE ENTREGA LÍMITE: 17/08/2024**

**CALIFICACIÓN: _____**

REPORTE DE PRÁCTICA:

## 1.- Ejercicios planteados

### Primer ejercicio

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

Para resolver estos ejercicios, se uso la base de los ejercicios hechos en clase, en donde se habían creado 4 variables.

```
float r, g, b; // variables para el cambio de color
int aux; // indice de color
```

En este caso, se incluye la librería “stdlib.h” para usar “rand” y “srand”, también se incluye la librería “time.h” para utilizar la palabra de definida “time”.

```
#include <time.h>
#include <stdlib.h>
```

Para inicializar los números en aleatorios es necesario incluir “srand(time(NULL))”, “srand” será la encargada de inicializar los números aleatorios esto es necesario colocar en el parámetro de srand “time” puede ser time(0) o time(NULL) como en este caso. Se podría decir que es una semilla para generar números aleatorios.

```
srand(time(NULL)); // se inicializan los valores aleatorios
```

Ahora para poder ponerle un límite a los números aleatorios, porque en mi índice usado es de 0 a , se usó la siguiente línea en donde se da a entender que va de 0 a 3, los necesarios para hacer que se vean los colores en modo aleatorio.

```
aux = rand() % 4; // se especifica que vaya de 0 a 3
```

Al tener los numeros aleatorios funcionando correctamente, ahora solo tuve que modificar mi else if, ya que en mi ejercicio de clase la variable “aux” sumaba uno para poder ir recorriendo la gama de colores, ahora ya no es necesario hacer esa suma, porque los numeros aleatorios van de 0 a 2.

```
if (aux == 0) {
    r = 1.0f, g = 0.0f, b = 0.0f; // color rojo

}else if(aux == 1) {
    r = 0.0f, g = 1.0f, b = 0.0f; // color verde

}else if(aux == 2) {
    r = 0.0f, g = 0.0f, b = 1.0f; // color azul


}
else {
    // el numero 3 no hace ningun cambio de color
}
```

Agregue una impresión de pantalla para ver que se estaban generando aleatoriamente.

```
printf("num: %d\n", aux);
```

```
num: 0
num: 3
num: 1
num: 0
num: 2
num: 1
```

El ejercicio pedía 2 segundos, por lo tanto, con la investigación que realice para resolver mi ejercicio de clase ya contaba con 2 segundos, porque 2000 en el parámetro de “Sleep” equivale a 2 segundos.

```
Sleep(2000); //Su parametro son milisegundos por lo tanto 2000 milisegundos equivale a 2 segundos
```

Las evidencias de que funciona correctamente, se incluirán en la entrega de documentos en un video para que se vea el ciclo de 2 segundos y de manera aleatorio los colores

**Segundo ejercicio**

**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**

Para resolver este ejercicio, fui siguiendo la lógica del ejercicio en clase.

Mi primera letra a poner fue “C”, la cual ocupo el lado negativo del cuadrante para que cupieran las 3 letras. Las coordenadas fueron las siguientes:

```
// LETRA C

// Primer Triangulo
-0.9f,0.6f,0.0f,
-0.9f,0.3f,0.0f,
-0.45f,0.6f,0.0f,

// Segundo Triangulo
-0.9f,0.3f,0.0f,
-0.45f,0.6f,0.0f,
-0.45f,0.3f,0.0f,

// Tercer Triangulo
-0.9f,0.3f,0.0f,
-0.9f,-0.4f,0.0f,
-0.7f,0.3f,0.0f,

// Cuarto Triangulo
-0.9f,-0.4f,0.0f,
-0.7f,-0.4f,0.0f,
-0.7f,0.3f,0.0f,

// Quinto Triangulo
-0.9f,-0.6f,0.0f,
-0.9f,-0.3f,0.0f,
-0.45f,-0.6f,0.0f,

// Sexto Triangulo
-0.9f,-0.3f,0.0f,
-0.45f,-0.6f,0.0f,
-0.45f,-0.3f,0.0f,
```

La segunda letra a dibujar fue “i” mayúscula, y las coordenadas fueron las siguientes:

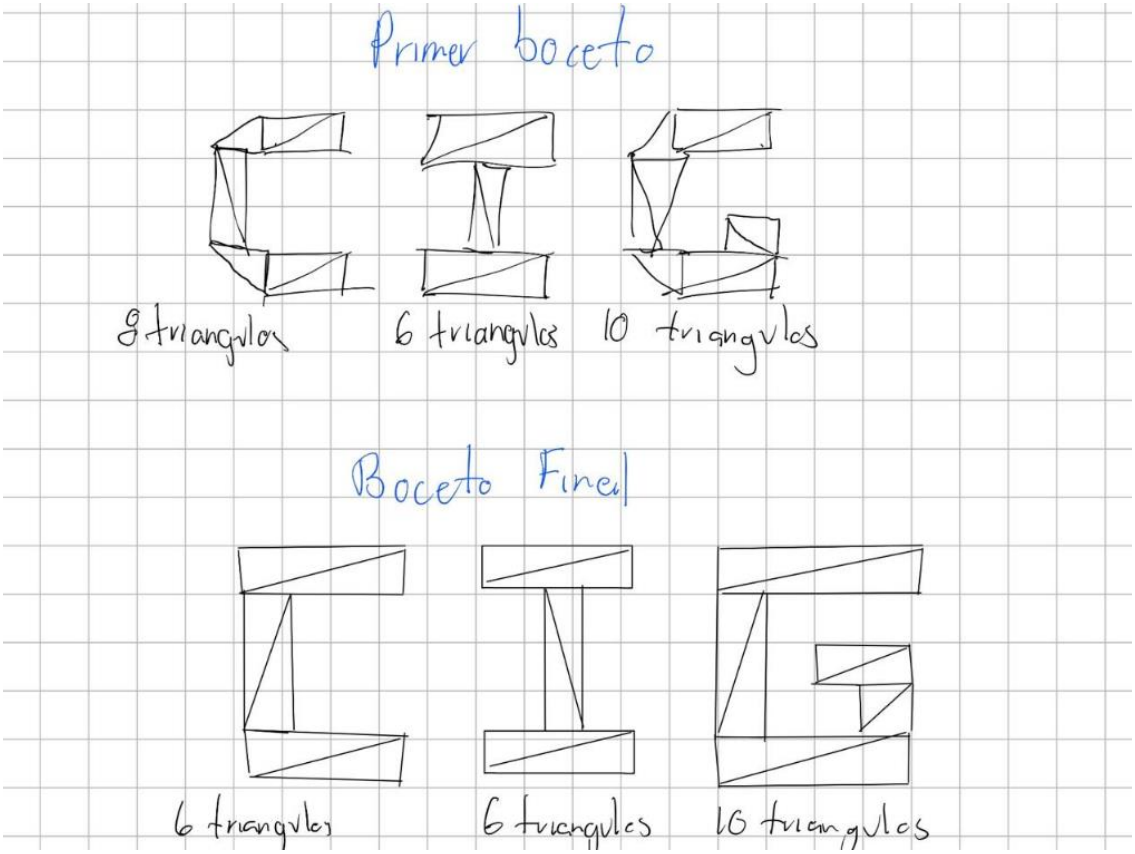
```
// LETRA I

// Primer Triangulo
-0.4f,0.6f,0.0f,
-0.4f,0.3f,0.0f,
0.1f,0.6f,0.0f,

// Segundo Triangulo
-0.4f,0.3f,0.0f,
0.1f,0.6f,0.0f,
0.1f,0.3f,0.0f,

// Tercer Triangulo
-0.25f,0.3f,0.0f,
-0.05f,0.3f,0.0f,
-0.05f,-0.3f,0.0f,

// Cuarto Triangulo
-0.25f,0.3f,0.0f,
-0.25f,-0.3f,0.0f,
-0.05f,-0.3f,0.0f,

// Quinto Triangulo
-0.4f,-0.6f,0.0f,
-0.4f,-0.3f,0.0f,
0.1f,-0.6f,0.0f,

// Sexto Triangulo
-0.4f,-0.3f,0.0f,
0.1f,-0.6f,0.0f,
0.1f,-0.3f,0.0f,
```

La tercera letra que dibuje fue “G”

```
//Letra G

// Primer Triangulo
0.15f,0.6f,0.0f,
0.15f,0.3f,0.0f,
0.85f,0.6f,0.0f,

// Segundo Triangulo
0.15f,0.3f,0.0f,
0.85f,0.6f,0.0f,
0.85f,0.3f,0.0f,

// Tercer Triangulo
0.15f,0.3f,0.0f,
0.15f,-0.4f,0.0f,
0.35f,0.3f,0.0f,

// Cuarto Triangulo
0.15f,-0.4f,0.0f,
0.35f,-0.4f,0.0f,
0.35f,0.3f,0.0f,

// Quinto Triangulo
0.15f,-0.6f,0.0f,
0.15f,-0.3f,0.0f,
0.85f,-0.6f,0.0f,

// Sexto Triangulo
0.15f,-0.3f,0.0f,
0.85f,-0.6f,0.0f,
0.85f,-0.3f,0.0f,
```

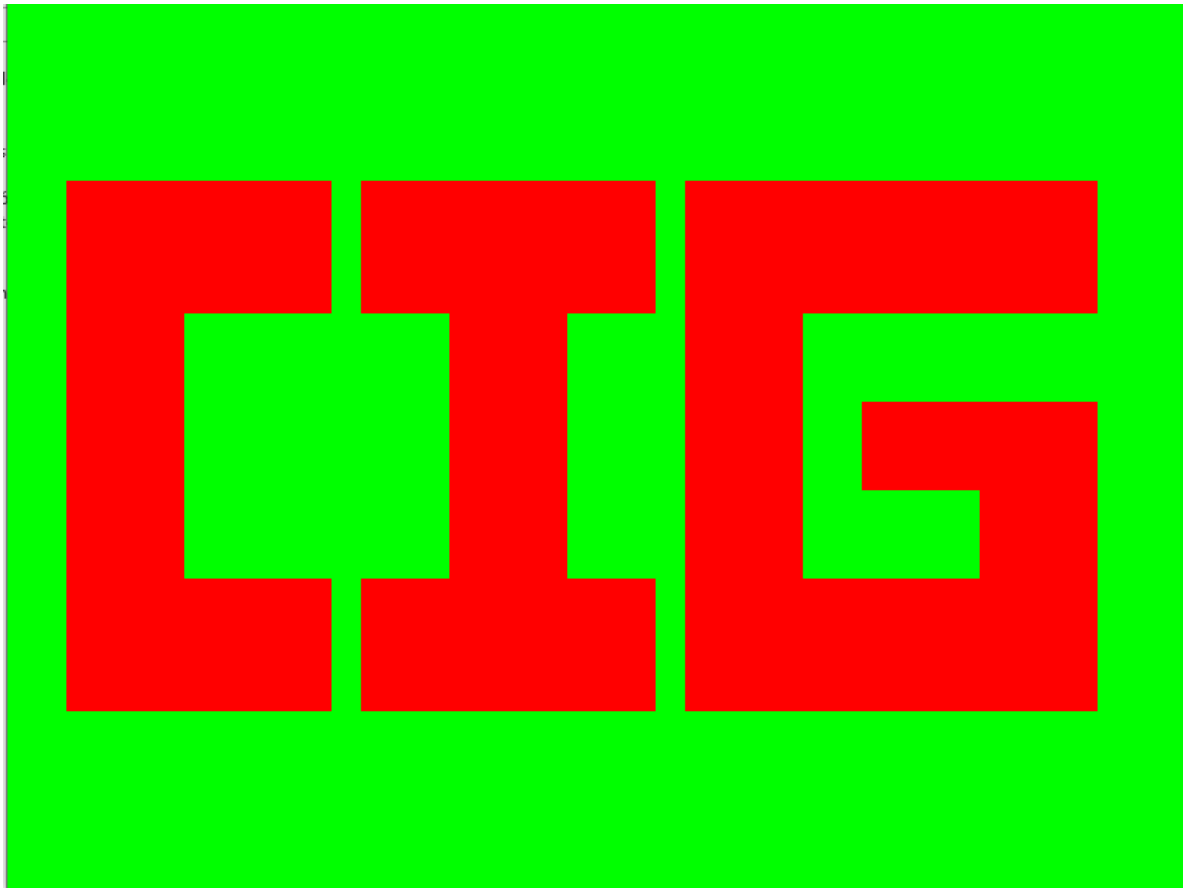
```
// Septimo Triangulo
0.65f,-0.3f,0.0f,
0.85f,-0.1f,0.0f,
0.85f,-0.3f,0.0f,

// Octavo Triangulo
0.65f,-0.1f,0.0f,
0.85f,-0.1f,0.0f,
0.65f,-0.3f,0.0f,

// Noveno Triangulo
0.45f,-0.1f,0.0f,
0.85f,0.1f,0.0f,
0.85f,-0.1f,0.0f,

// Decimo Triangulo
0.45f,-0.1f,0.0f,
0.45f,0.1f,0.0f,
0.85f,0.1f,0.0f
```

Para realizar este ejercicio no fue tan complicado como me lo imagine, me ayudo bastante realizar un boceto de cuantos triángulos iba a necesitar para poder lograr el objetivo.




Al haber hecho el primer boceto, me di cuenta que estaba en lo incorrecto al pensar que los triángulos obligatoriamente tendrían que ser 3,4 y 5 de hipotenusa para tener el mismo lado. Al percatarme de eso, logre facilitarme el trabajo y solo hacer más largo los triángulos para no tener que usar tantos a la vez.

Finalmente, se tienen los 2 ejercicios ejecutándose al mismo tiempo, este fue el resultado.



## 2.- Problemas presentados

Un “error” que presente dentro de la práctica fue los números aleatorios, ya que al poner que solo fueran 3 me salían las probabilidades como 0 0, 1 1 , 2 2 y no se podía ver el cambio de color como me gustaría, por eso la solución que implemente fue colocar 1 número más para que existan más probabilidades.

Ejemplo de los resultados al solo poner 3 números

```
num: 1
num: 1
num: 2
num: 0
num: 0
num: 1
num: 1
num: 2
num: 2
num: 0
num: 0
num: 1
num: 1
```

Otro “problema” que presente en esta practica fue que no recordaba como incluir de manera correcta los datos aleatorios, sabía que existía la función “rand”, pero no sabia como ejecutarla en su totalidad, y es por eso que investigue como incluirla de una manera correcta.

## 3.- Conclusión

La complejidad de los ejercicios presentados en esta practica fueron muy semejantes a los ejercicios hechos en clase, y creo que gracias a esos ejercicios fue que se me facilito bastante resolver la práctica. A mi parecer, que nos proporcionen esos ejercicios es de bastante ayuda, porque nos ofrecen las herramientas necesarias para resolver este tipo de problemas. Al momento de hacer los triángulos, me percate que una herramienta de gran ayuda seria GeoGebra, porque nos daríamos cuenta de modo grafico cuales serian las coordenadas para dibujar los triángulos de una forma más rápida, porque creo que estar dibujando a cada rato un boceto me haría perder tiempo en un futuro.

## 4.- Bibliografía


- TylerMSFT. (2024b, agosto 3). *srand*. Microsoft Learn. https://learn.microsoft.com/es-es/cpp/c-runtime-library/reference/srand?view=msvc-170
- TylerMSFT. (2023, 12 octubre). *rand*. Microsoft Learn. https://learn.microsoft.com/es-es/cpp/c-runtime-library/reference/rand?view=msvc-170
- *IBM i 7.5*. (s. f.). https://www.ibm.com/docs/es/i/7.5?topic=files-timeh