



Universidad Nacional
Autónoma de México



Practica No. 1

Alumno: Arroyo Llanes Miguel Alejandro.

No. Cuenta: 319218460

Materia: Lab. Computación Grafica e Interacción Humano-Computadora

Grupo Laboratorio: 3

Grupo teoría: 6

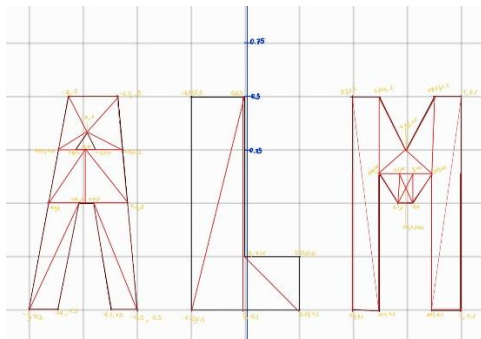
Fecha de Entrega: 15/02/2025

Nombre profesor: Ing. Jose Roque Roman Guadarrama

Comentario:

Primeramente, se buscó hacer el cambio de fondo en la ejecución por el RGB de forma aleatoria, se implementaron las bibliotecas `stdlib.h` y `time.h` para cambiar de forma aleatoria, también se implemento el cambio de color de forma aleatoria cada 2 segundos.

Posteriormente se dibujaron las letras con triángulos, primeramente, se dibujaron de manera digital utilizando una Tablet para poder visualizar cuantos triángulos se necesitarían por letra y en que coordenadas se debían colocar los puntos para formar el triángulo. Una vez hechas las letras a mano y se dibujaran las coordenadas necesarias se escribió en el código las líneas suficientes para dibujar cada triángulo y formar así las letras.



Bloque de código:

Líneas de código que muestran el dibujo de los vértices (triángulos) de las letras “A”, “L” y “M”.

```
Archivo Editar Ver Git Proyecto
Compilar Depurar Prueba Analizar
Herramientas Extensiones Ventana Ayuda
Debug
Practica1
E01-319218460.cpp:2
Practica1 (Ámbito global)
void CrearFiguras()
{
    GLfloat letra_A[] = {
        // Triángulo lateral izquierdo inferior izquierdo
        -1.0f, -0.5f, 0.0f,
        -0.9f, 0.0f, 0.0f,
        -0.75f, -0.0f, 0.0f,
        // Triángulo lateral izquierdo inferior izquierdo derecho
        -1.0f, -0.5f, 0.0f,
        -0.8f, -0.5f, 0.0f,
        -0.75f, -0.0f, 0.0f,
        // Triángulo lateral derecho inferior derecho
        -0.7f, -0.5f, 0.0f,
        -0.74f, 0.0f, 0.0f,
        -0.5f, -0.5f, 0.0f,
        // Triángulo lateral derecho inferior derecho izquierdo
        -0.6f, -0.0f, 0.0f,
        -0.74f, 0.0f, 0.0f,
        -0.5f, -0.5f, 0.0f,
        //Partes superiores del triángulo de la sección 2
        -0.9f, -0.0f, 0.0f,
        -0.85f, 0.25f, 0.0f,
        -0.75f, 0.25f, 0.0f,
        //Texto para separar las secciones
        -0.9f, -0.0f, 0.0f,
        -0.75f, 0.0f, 0.0f,
        -0.75f, 0.25f, 0.0f,
        //Partes superiores del triángulo de la sección 2
        -0.6f, -0.0f, 0.0f,
        -0.65f, 0.25f, 0.0f,
        -0.75f, 0.25f, 0.0f,
    };

    //Partes superiores del triángulo de la sección 3
    -0.85f, 0.25f, 0.0f,
    -0.76f, 0.25f, 0.0f,
    -0.75f, 0.3f, 0.0f,
    //Texto para separar las secciones
    -0.65f, 0.25f, 0.0f,
    -0.74f, 0.25f, 0.0f,
    -0.75f, 0.3f, 0.0f,
    //Partes superiores del triángulo de la sección 4
    -0.85f, 0.25f, 0.0f,
    -0.8f, 0.5f, 0.0f,
    -0.75f, 0.3f, 0.0f,
    //Texto para separar las secciones
    -0.65f, 0.25f, 0.0f,
    -0.7f, 0.5f, 0.0f,
    -0.75f, 0.3f, 0.0f,
    //Partes superiores del triángulo de la sección 5
    -0.7f, 0.5f, 0.0f,
    -0.8f, 0.5f, 0.0f,
    -0.75f, 0.3f, 0.0f,
    // Vértices de la letra L
    GLfloat letra_L[] = {
        //Partes del triángulo de la sección 1
        -0.25f, -0.5f, 0.0f,
        -0.8f, 0.5f, 0.0f,
        -0.25f, 0.5f, 0.0f,
        //Texto para separar las secciones
        -0.25f, -0.5f, 0.0f,
        0.8f, 0.5f, 0.0f,
        -0.8f, -0.5f, 0.0f,
        //Partes del triángulo de la sección 2
        -0.8f, -0.25f, 0.0f,
    };
}
```

```

// E01-319218460.cpp2
// (Ámbito global)
// CrearFiguras()

120 0.25f, -0.5f, 0.0f,
121 -0.0f, -0.5f, 0.0f,
122
123 //texto para separar las secciones
124 -0.0f, -0.25f, 0.0f,
125 0.25f, -0.5f, 0.0f,
126 0.25f, -0.25f, 0.0f,
127
128
129
130
131 // Vértices de la letra M
132 GLfloat letra_M[] = {
133 //Partes del triángulo de la sección 1 izquierda
134 0.5f, -0.5f, 0.0f,
135 0.5f, 0.5f, 0.0f,
136 0.625f, -0.5f, 0.0f,
137
138 //texto para separar las secciones
139 0.625f, 0.5f, 0.0f,
140 0.5f, 0.5f, 0.0f,
141 0.625f, -0.5f, 0.0f,
142
143 //Partes del triángulo de la sección 1 derecha
144 0.875f, 0.5f, 0.0f,
145 0.875f, -0.5f, 0.0f,
146 1.0f, 0.5f, 0.0f,
147
148 //texto para separar las secciones
149 1.0, -0.5f, 0.0f,
150 0.875f, -0.5f, 0.0f,
151 1.0f, 0.5f, 0.0f,
152
153 //Partes del triángulo de la sección central inferior
154 0.625f, 0.125f, 0.0f,
155 0.7f, 0.0f, 0.0f,
156 0.7f, 0.125f, 0.0f,
157
158
159
160
161
162
163
164 //texto para separar las secciones
165 0.75f, -0.0f, 0.0f,
166 0.7f, 0.0f, 0.0f,
167 0.8f, 0.0f, 0.0f,
168
169 //texto para separar las secciones
170 0.75f, 0.0625f, 0.0f,
171 0.7f, 0.125f, 0.0f,
172 0.7f, 0.125f, 0.0f,
173
174 //texto para separar las secciones
175 0.75f, 0.0625f, 0.0f,
176 0.8f, 0.125f, 0.0f,
177 0.8f, 0.0f, 0.0f,
178
179 //texto para separar las secciones
180 0.875f, 0.125f, 0.0f,
181 0.8f, 0.125f, 0.0f,
182 0.8f, 0.0f, 0.0f,
183
184 //Partes del triángulo de la sección central superior
185 0.625f, 0.125f, 0.0f,
186 0.875f, 0.125f, 0.0f,
187 0.75f, 0.25f, 0.0f,
188
189 //texto para separar las secciones
190 0.625f, 0.125f, 0.0f,
191 0.625f, 0.5f, 0.0f,
192 0.75f, 0.25f, 0.0f,
193
194 //texto para separar las secciones
195 0.875f, 0.5f, 0.0f,
196 0.875f, 0.125f, 0.0f,
197 0.75f, 0.25f, 0.0f,
198
199
200
201
202
203
204 numVertices = (sizeof(letra_A) + sizeof(letra_L) + sizeof(letra_M)) / (3 * sizeof(GLfloat)); //

```

```

numVertices = (sizeof(letra_A) + sizeof(letra_L) + sizeof(letra_M)) / (3 * sizeof(GLfloat)); // Calcular la cantidad correcta de vértices

glGenVertexArrays(1, VAO); // Generar 1 VAO
glGenBuffers(1, VBO); // Generar 1 VBO

glBindVertexArray(VAO[0]); // Asignar VAO
glBindBuffer(GL_ARRAY_BUFFER, VBO[0]);
glBufferData(GL_ARRAY_BUFFER, sizeof(letra_A) + sizeof(letra_L) + sizeof(letra_M), NULL, GL_STATIC_DRAW);
glBufferSubData(GL_ARRAY_BUFFER, 0, sizeof(letra_A), letra_A);
glBufferSubData(GL_ARRAY_BUFFER, sizeof(letra_A), sizeof(letra_L), letra_L);
glBufferSubData(GL_ARRAY_BUFFER, sizeof(letra_A) + sizeof(letra_L), sizeof(letra_M), letra_M);

glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 3 * sizeof(GL_FLOAT), (GLvoid*)0);
glEnableVertexAttribArray(0);

glBindBuffer(GL_ARRAY_BUFFER, 0);
glBindVertexArray(0);

```

Cambio de color de fondo de la pantalla de la ejecución en colores R (rojo), G (verde) y B (azul), los cuales cambiaran aleatoriamente cada 2 segundos.

```

//Limpiar la ventana
//glClearColor(0.0f,0.0f,0.0f,1.0f);
glClear(GL_COLOR_BUFFER_BIT);

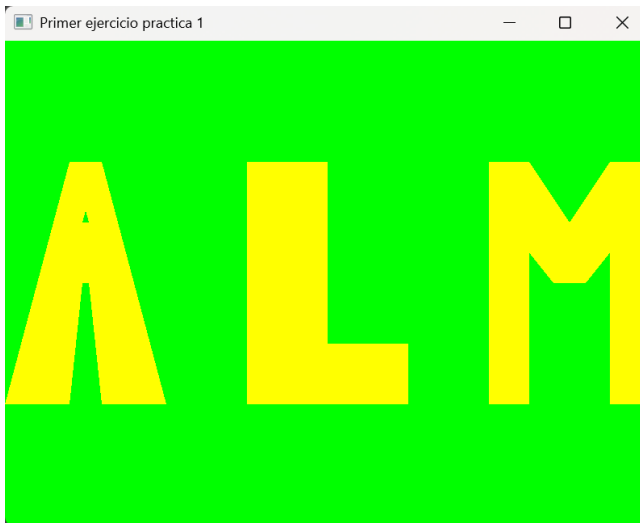
double currentTime = glfwGetTime();
if (currentTime - lastTime >= 2.0) { // Cambiar color cada 2 segundos
    int colorIndex = rand() % 3; // Elegir un número aleatorio entre 0 y 2

    if (colorIndex == 0) {
        glClearColor(1.0f, 0.0f, 0.0f, 1.0f); // Rojo
    }
    else if (colorIndex == 1) {
        glClearColor(0.0f, 1.0f, 0.0f, 1.0f); // Verde
    }
    else {
        glClearColor(0.0f, 0.0f, 1.0f, 1.0f); // Azul
    }

    lastTime = currentTime;
}

```

Ejecución:



Conclusión:

Con lo aprendido en la clase y en el previo he aprendido el como hacer dibujos con vértices al menos de momento en un espacio de 2 dimensiones, así como aprendí a optimizarlo de la mejor manera cada letra para evitar desperdiciar recursos, aunque aun se puede mejorar y aun es una introducción al tema, se ha logrado absorber el conocimiento suficiente y reafirmar el que se tenia anteriormente con cursos pasados.

Bibliografía:

1. En esta practica no se hizo uso de fuentes alternas a las que ya se habían aprendido en el laboratorio y previo.