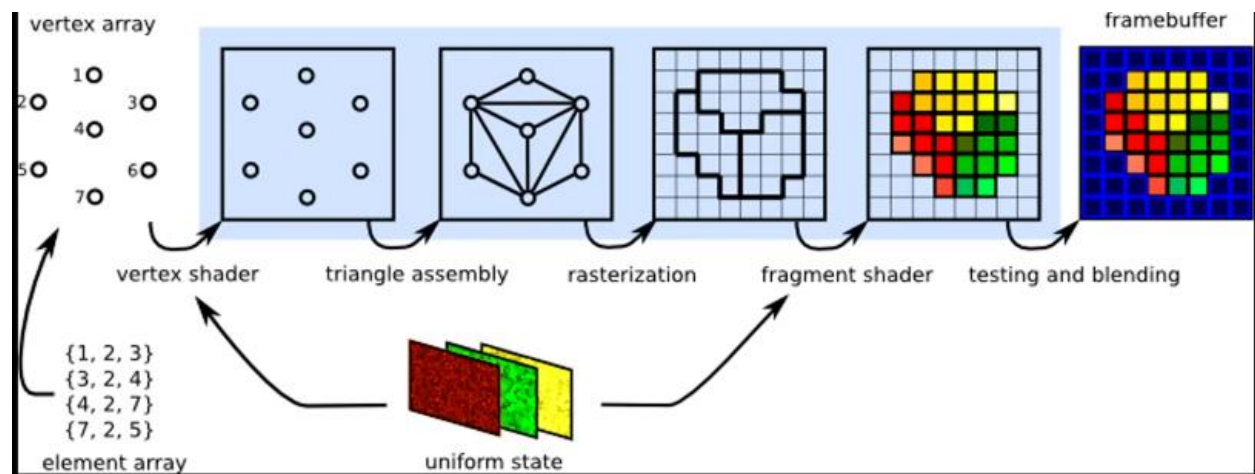


Introducción

En este material vamos a aprender a como dibujar primitivas en 2D utilizando shaders en Open GL, exploraremos el código base y entenderemos como cada parte contribuye a la renderización final en pantalla y de igual manera vamos a ver como modificar el proyecto de la sesión anterior, para poder integrar estos nuevos archivos.

Desarrollo

Es importante repasar el Pipeline de renderizado en Open GL



Los datos provienen de los vértices, que pasan primero al shader de vértices y luego se ensamblan para pasar a la etapa de rasterización y posteriormente al shader de fragmentos. Posteriormente, estos entran en el buffer de shade y de ahí llegamos a renderizar las primitivas en 2D.

Agregamos el nuevo código base y una librería que es el shader en nuestro proyecto de Visual Studio.

Ahora agregamos los shaders como elementos por separado para tener un mejor orden y estructura al momento de generar nuestros códigos.

Tenemos las bibliotecas que vamos a estar ocupando

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
// Shaders
```

Tenemos aquí la creación de la ventana, el tamaño que vamos a ocupar

```
void resize(GLFWwindow* window, int width, int height);  
  
const GLint WIDTH = 800, HEIGHT = 600;
```

Y el nombre que va a tener la ventana dibujo de primitivas 2D, posteriormente inicializamos la parte de las librerías para la creación de la ventana

```
//Verificación de errores de creacion ventana  
if (window== NULL)  
{  
    std::cout << "Failed to create GLFW window" << std::endl;  
    glfwTerminate();  
    return EXIT_FAILURE;  
}  
  
glfwMakeContextCurrent(window);  
glewExperimental = GL_TRUE;
```

Aquí tenemos información referente al software grafico que estamos ocupando.

Uno de los cambios significativos es el siguiente:

```
Shader ourShader("Shader/core.vs", "Shader/core.frag");
```

Aquí lo que hacemos es cargar la información de los archivos que tenemos por separado.

Colocamos una variable tipo shader una variable que va a tener el shader y la ruta en donde se encuentran los archivos.

Posteriormente tenemos la definición de los vértices que estaremos ocupando, esto nos va a permitir trabajar con la información que vamos a utilizar.

Aquí tenemos la posición en 3 coordenadas x,y,z.

```
Shader ourShader("Shader/core.vs", "Shader/core.frag");  
  
// Set up vertex data (and buffer(s)) and attribute pointers  
float vertices[] = {  
    0.5f,  0.5f, 0.0f,  1.0f, 0.0f, 0.0f, // top right  
    0.5f, -0.5f, 0.0f,  1.0f, 1.0f, 0.0f, // bottom right  
    -0.5f, -0.5f, 0.0f,  1.0f, 0.0f, 1.0f, // bottom left  
    -0.5f,  0.5f, 0.0f,  1.0f, 1.0f, 0.0f, // top left  
};  
unsigned int indices[] = { // note that we start from 0!
```

Pero en este caso, como solo vamos a trabajar en 2 dimensiones dejamos en 0 la componente z que representa la componente de la profundidad.

A agregamos el color que van a tener cada una de la información extra o atributos/vértices y están en el modelo de color RGB, cada una de las componentes que van de 0 a 1.

Y aquí están definidos cada uno de los vértices que vamos a tener, en este caso que seria definida como superior derecha/izquierda, la inferior derecha/izquierda.

Posteriormente tenemos los índices que tienen cada uno de los vértices los vamos a ocupar para dibujar cada uno de estos vértices.

```
};  
unsigned int indices[] = { // note that we start from 0!  
    3,2,1, // second Triangle  
    0,1,3,
```

Cargamos la información de los vértices a los buffers, para poderlos enlazar y posteriormente enviarlos al shader.

Frames por segundo, son los ciclos o velocidades es lo que vamos a estar haciendo para estas primitivas básicas

Las primitivas básicas

1.-Limpiar el buffer de color

2.-Dibujar cada uno de los elementos

```
while (!glfwWindowShouldClose(window))  
{  
    // Check if any events have been activated (key pressed, mouse moved etc.) and call corresponding response  
    glfwPollEvents();  
  
    // Render  
    // Clear the colorbuffer  
    glClearColor(0.0f, 0.0f, 0.0f, 1.0f);  
    glClear(GL_COLOR_BUFFER_BIT);  
  
    // Draw our first triangle  
    ourShader.Use();  
    glBindVertexArray(VAO);  
  
    glPointSize(1);  
    glDrawArrays(GL_POINTS, 0, 1);  
  
    //glDrawArrays(GL_LINES, 0, 2);  
    //glDrawArrays(GL_LINE_LOOP, 0, 4);  
  
    //glDrawArrays(GL_TRIANGLES, 0, 3);  
    //glDrawElements(GL_TRIANGLES, 3, GL_UNSIGNED_INT, 0);  
}
```

¿Cómo funciona la comunicación entre shaders?

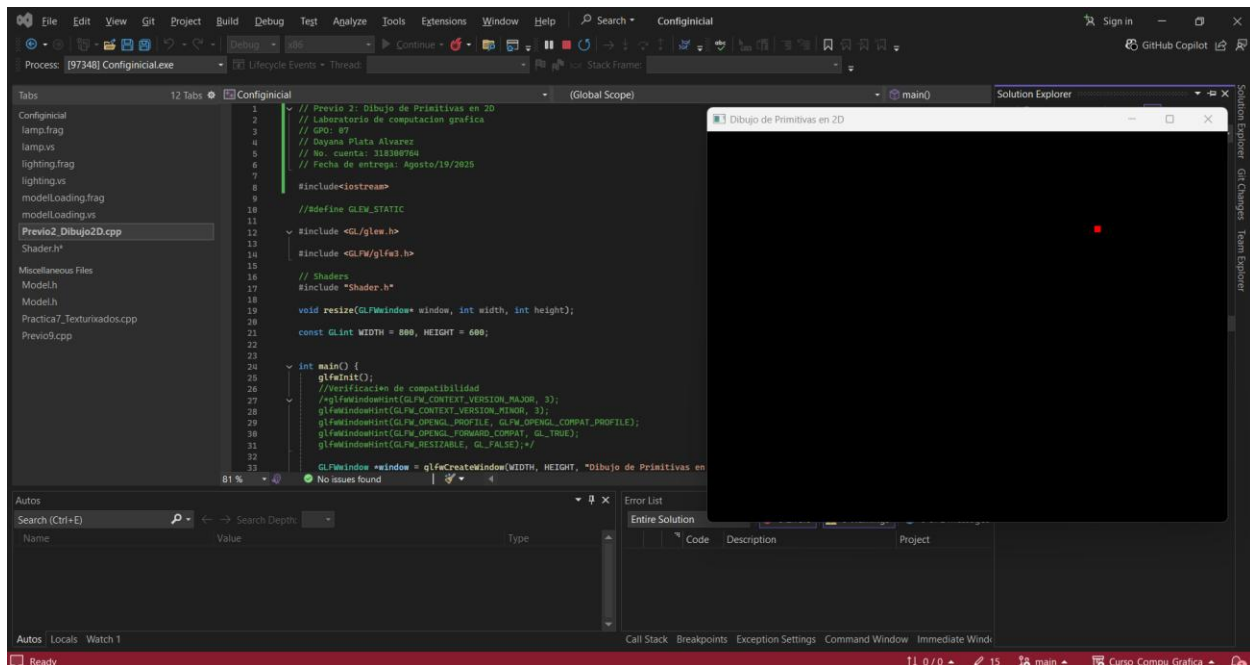
Primero se manda a llamar el shader de vértices, lo que nosotros hacemos es tener definido la localidad de memoria en donde se van a estar almacenando cada uno de estos elementos.

Dibujo

Lo más básico que vamos a poder dibujar es un punto.

Si ejecutamos este código, podemos observar que tenemos una pantalla con fondo negro, y hay un pequeño vertice de color rojo.

Si modificamos el tamaño ya podemos observar el punto rojo.



La ventana va de -1 a 1 en el caso de la componente horizontal/vertical.

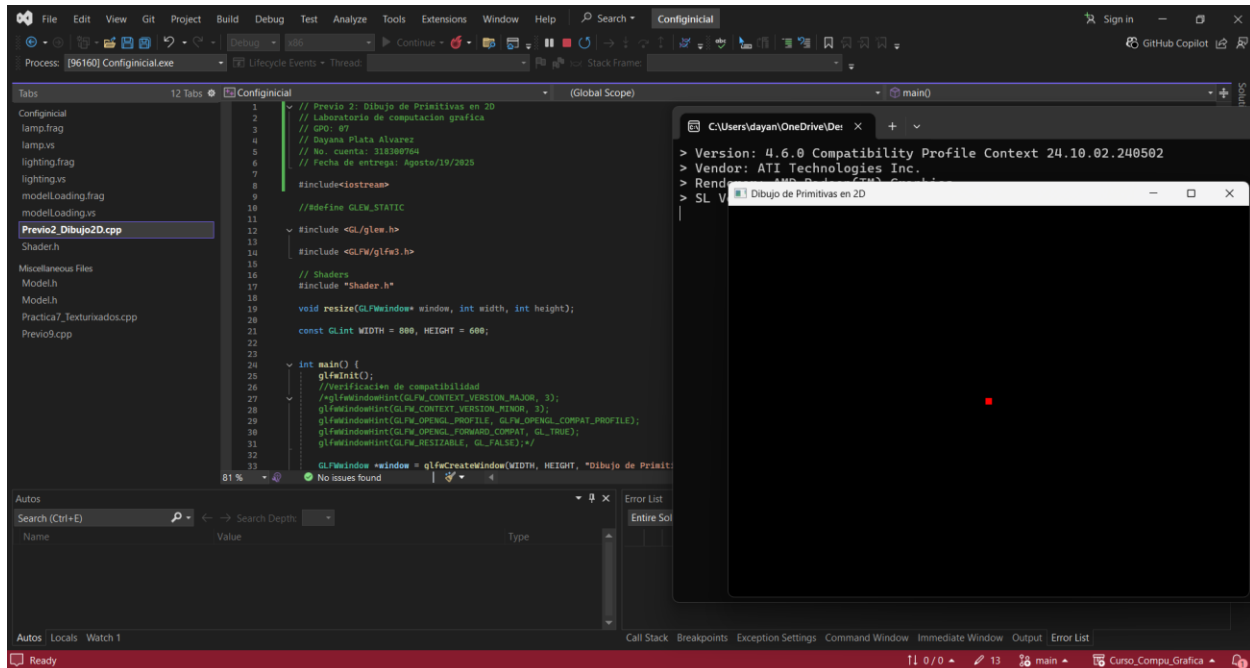
Si quisiéramos tener el vertice en el origen, cambiamos los parámetros del vertice:

```
// Set up vertex data (and buffer(s)) and attribute pointers
float vertices[] = {
    0.0f, 0.0f, 0.0f, 1.0f, 0.0f, 0.0f, // top right
    0.5f, -0.5f, 0.0f, 1.0f, 1.0f, 0.0f, // bottom right
    -0.5f, -0.5f, 0.0f, 1.0f, 0.0f, 1.0f, // bottom left
    -0.5f, 0.5f, 0.0f, 1.0f, 1.0f, 0.0f, // top left
}
```

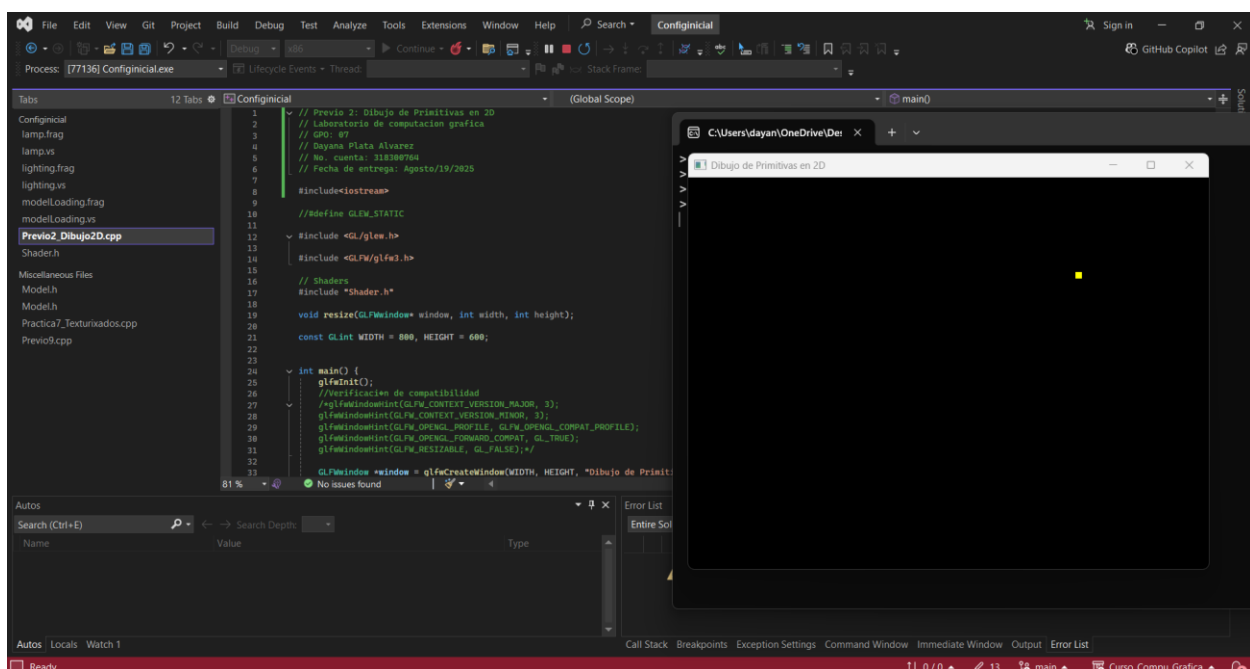
Previo 2: Dibujo de primitivas en 2D
Fecha de entrega: 19 de agosto del 2025

Alumna: Plata Alvarez Dayana
No. Cuenta: 318300764

Aquí podemos observar que lo estamos dibujando en el centro de la ventana



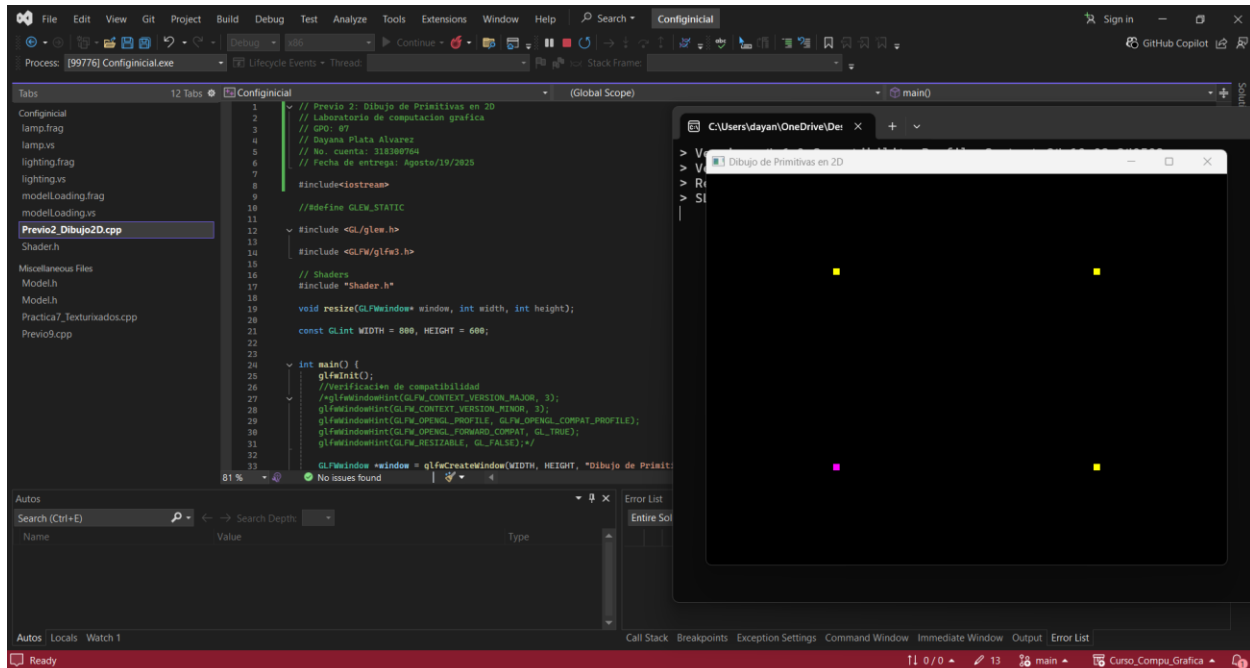
Para cambiar los colores



Previo 2: Dibujo de primitivas en 2D
Fecha de entrega: 19 de agosto del 2025

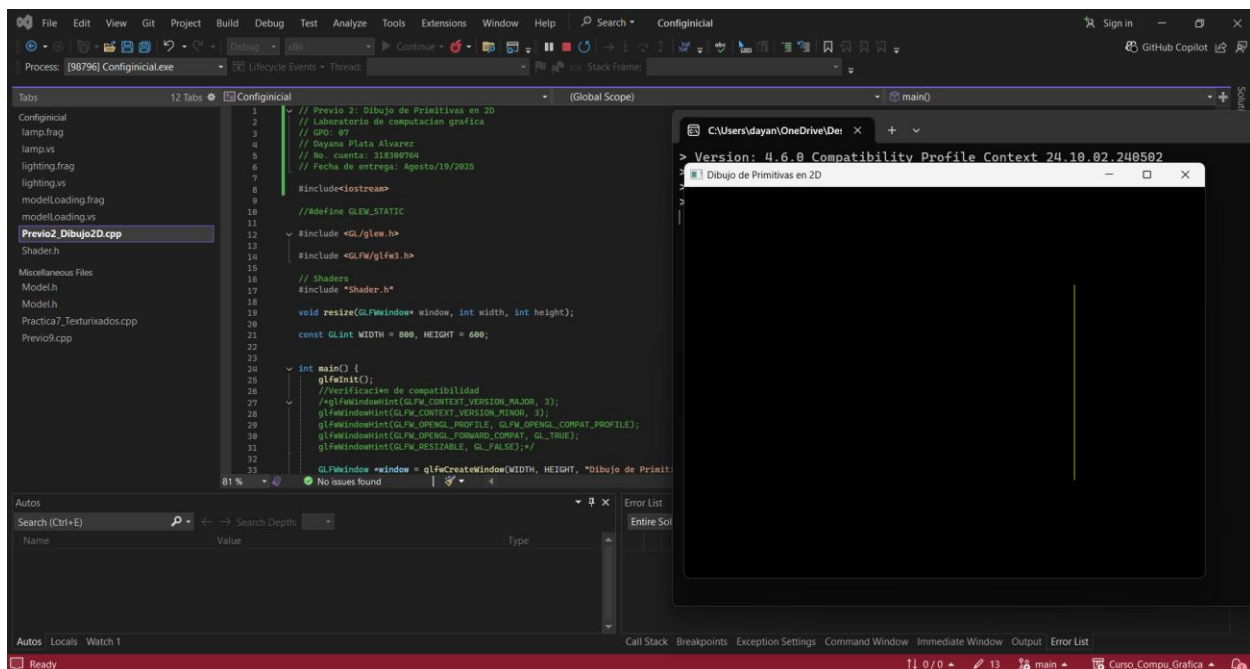
Alumna: Plata Alvarez Dayana
No. Cuenta: 318300764

Ahora para tomar los 4 elementos modificamos los parámetros y obtenemos lo siguiente:



Y ya obtenemos los 4 vértices de la ventana, Posteriormente lo que podemos dibujar como primitivas básicas son las líneas.

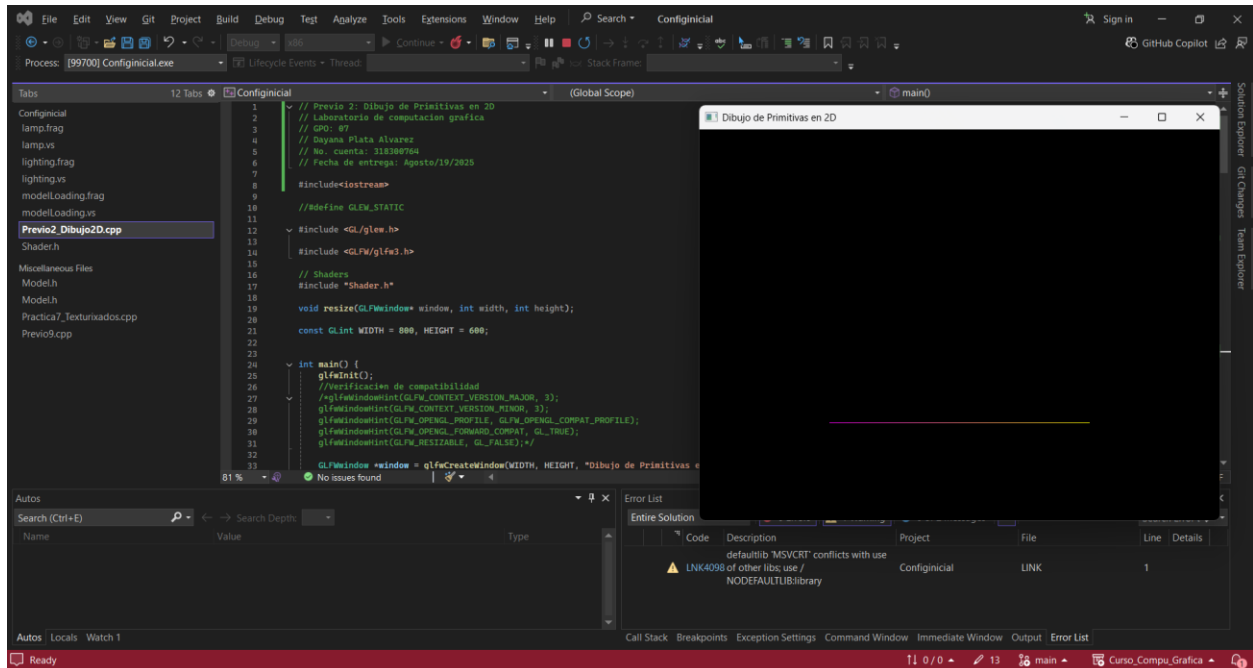
Aquí ya estamos uniendo las 2 aristas



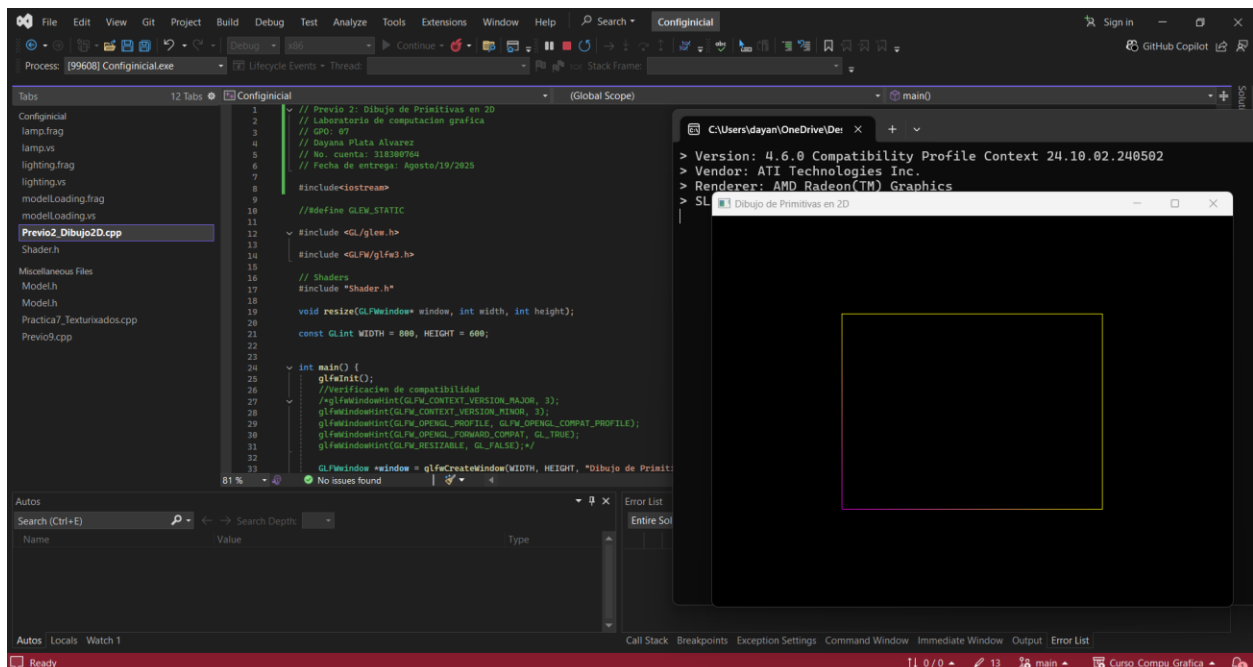
Previo 2: Dibujo de primitivas en 2D
Fecha de entrega: 19 de agosto del 2025

Alumna: Plata Alvarez Dayana
No. Cuenta: 318300764

Agregamos otra línea, que seria el siguiente elemento de mi arreglo.



Podemos ir manipulando parte por parte, pero si ahora queremos dibujar el cuadrado completo, tenemos otra instrucción.

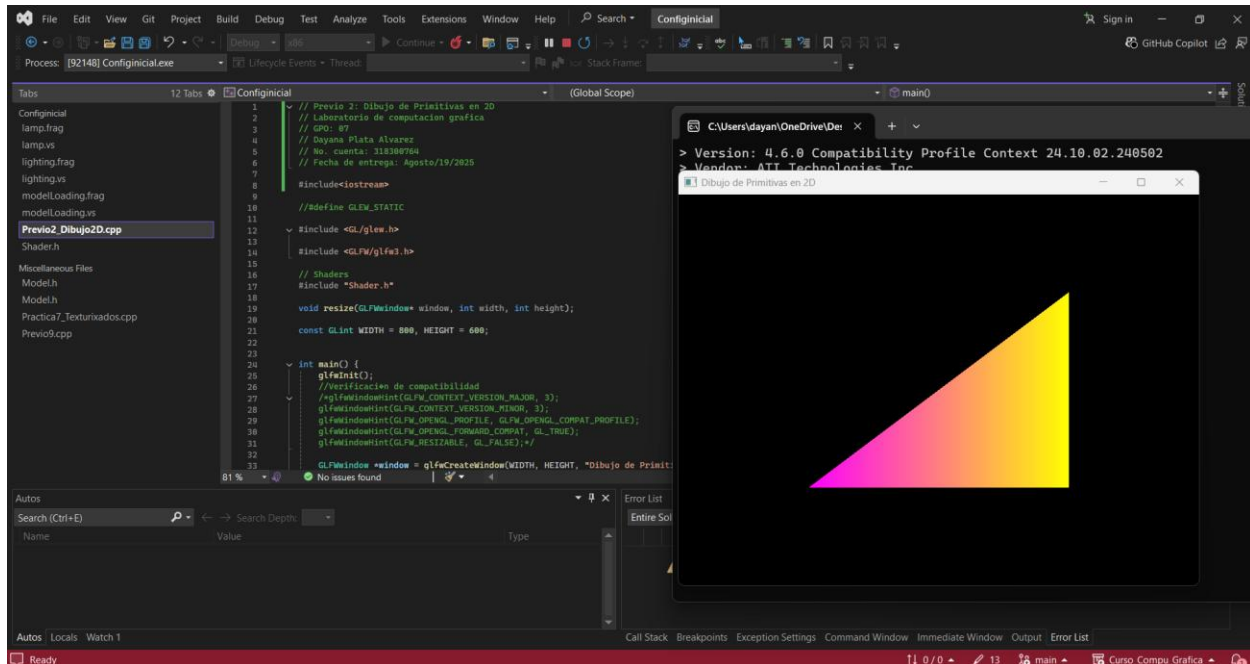


Esta instrucción, permite poder cerrar cada uno de estos vértices por medio de las líneas.

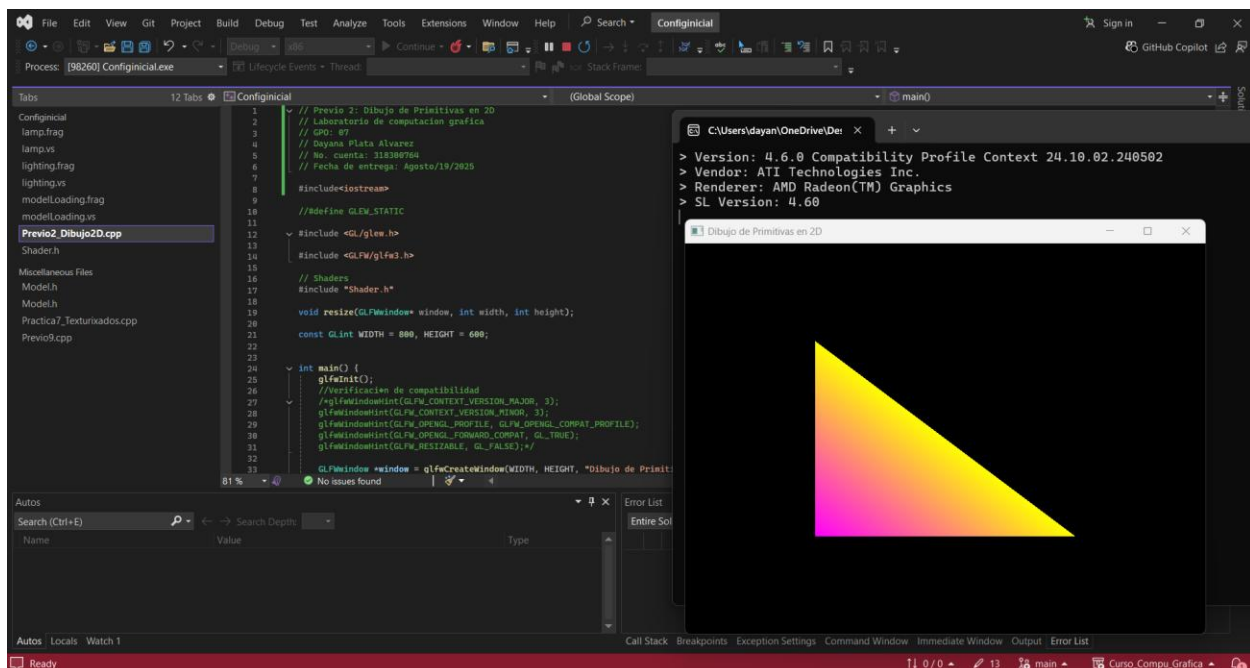
Previo 2: Dibujo de primitivas en 2D
Fecha de entrega: 19 de agosto del 2025

Alumna: Plata Alvarez Dayana
No. Cuenta: 318300764

La siguiente instrucción, es para dibujar triángulos.



Esta instrucción lo que hace es tomar del arreglo el primer elemento y los 3 vértices posteriores a ese arreglo y podemos ir modificando cada uno de los argumentos para obtener las variantes que pueda ir creando.



Alumna: Plata Alvarez Dayana
No. Cuenta: 318300764

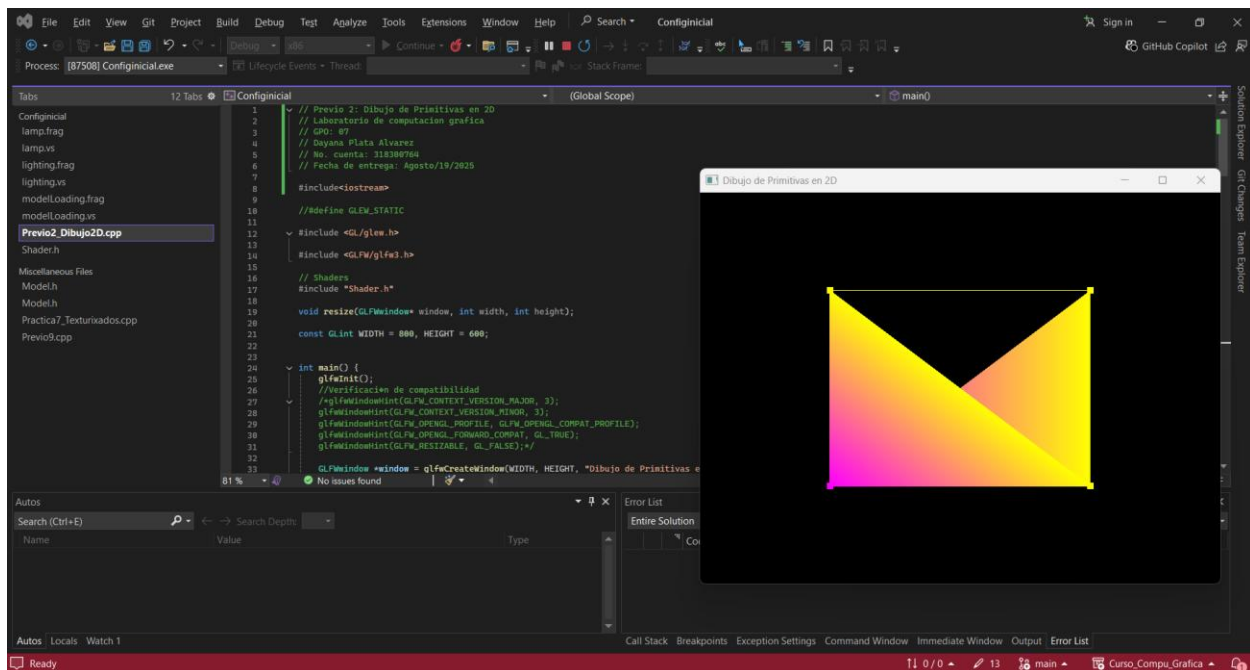
Para eso podemos utilizar nuestra última función, que en lugar de dibujar el arreglo directamente, dibuja los elementos que se encuentran en el arreglo, por medio de los índices.

The image shows a screenshot of a C++ development environment, likely Visual Studio, with the following components:

- Top Bar:** Displays standard application menus (File, Edit, View, etc.) and a search bar.
- Toolbar:** Contains icons for file operations, debugging, and navigation.
- Process:** Shows 'Process: [98260] Configurational.exe'.
- Stack Frame:** Displays the current execution context.
- File Explorer:** Shows the project structure with files like 'Configurational', 'lamp.frag', 'lamp.vs', 'lighting.frag', 'lighting.vs', 'modelLoading.frag', 'modelLoading.vs', 'Shader.h', 'Miscellaneous Files', 'Model.h', 'Model.h', 'Practica7_Texturizados.cpp', and 'Previo9.cpp'. The file 'Previo2_Dibujo2D.cpp' is selected.
- Code Editor:** Displays the source code for 'Previo2_Dibujo2D.cpp'. The code includes comments in Spanish, GLFW headers, and a main function that creates a window and draws a yellow triangle. The code is as follows:

```
1 // Previo 2: Dibujo de Primitivas en 2D
2 // Laboratorio de computacion grafica
3 // GPU: 67
4 // Dayana Plata Alvarez
5 // No. cuenta: 318389764
6 // Fecha de entrega: Agosto/19/2025
7
8 #include <iostream>
9
10 #define GLEW_STATIC
11
12 #include <GL/glew.h>
13 #include <GLFW/glfw3.h>
14
15 // Shaders
16 #include "Shader.h"
17
18 void resize(GLFWwindow* window, int width, int height);
19
20 const GLint WIDTH = 800, HEIGHT = 600;
21
22
23
24 int main() {
25     glfwInit();
26     //Verificacion de compatibilidad
27     #if defined(__APPLE__)
28     glewExperimental = GL_TRUE;
29     #endif
30     glfwWindowHint(GLFW_CONTEXT_VERSION_MAJOR, 3);
31     glfwWindowHint(GLFW_CONTEXT_VERSION_MINOR, 3);
32     glfwWindowHint(GLFW_OPENGL_PROFILE, GLFW_OPENGL_CORE_PROFILE);
33     glfwWindowHint(GLFW_OPENGL_FORWARD_COMPAT, GL_TRUE);
34     glfwWindowHint(GLFW_RESIZABLE, GL_FALSE);
35
36     GLFWwindow *window = glfwCreateWindow(WIDTH, HEIGHT, "Dibujo de Primitivas en 2D", NULL, NULL);
37     if (!window) {
38         return -1;
39     }
40     glfwMakeContextCurrent(window);
41     glewInit();
42     glfwSetWindowUserPointer(window, (void*)0);
43     glfwSetWindowSizeCallback(window, resize);
44     glfwSetKeyCallback(window, key_callback);
45     glfwSetMouseButtonCallback(window, mouse_button_callback);
46     glfwSetScrollCallback(window, scroll_callback);
47     glfwSetCharCallback(window, char_callback);
48     glfwSetCharModsCallback(window, char_mods_callback);
49     glfwSetJoystickCallback(window, joystick_callback);
50     glfwSetJoystickUserPointerCallback(window, joystick_user_pointer_callback);
51     glfwSetJoystickNameCallback(window, joystick_name_callback);
52     glfwSetJoystickIdCallback(window, joystick_id_callback);
53     glfwSetJoystickPathCallback(window, joystick_path_callback);
54     glfwSetJoystickSerialNumberCallback(window, joystick_serial_number_callback);
55     glfwSetJoystickVendorIdCallback(window, joystick_vendor_id_callback);
56     glfwSetJoystickProductIdCallback(window, joystick_product_id_callback);
57     glfwSetJoystickVersionCallback(window, joystick_version_callback);
58     glfwSetJoystickRevisionCallback(window, joystick_revision_callback);
59     glfwSetJoystickReleaseDateCallback(window, joystick_release_date_callback);
60     glfwSetJoystickWeightCallback(window, joystick_weight_callback);
61     glfwSetJoystickLengthCallback(window, joystick_length_callback);
62     glfwSetJoystickWidthCallback(window, joystick_width_callback);
63     glfwSetJoystickHeightCallback(window, joystick_height_callback);
64     glfwSetJoystickDepthCallback(window, joystick_depth_callback);
65     glfwSetJoystickAreaCallback(window, joystick_area_callback);
66     glfwSetJoystickVolumeCallback(window, joystick_volume_callback);
67     glfwSetJoystickPowerCallback(window, joystick_power_callback);
68     glfwSetJoystickTemperatureCallback(window, joystick_temperature_callback);
69     glfwSetJoystickHumidityCallback(window, joystick_humidity_callback);
70     glfwSetJoystickPressureCallback(window, joystick_pressure_callback);
71     glfwSetJoystickTensionCallback(window, joystick_tension_callback);
72     glfwSetJoystickTorqueCallback(window, joystick_torque_callback);
73     glfwSetJoystickForceCallback(window, joystick_force_callback);
74     glfwSetJoystickMomentCallback(window, joystick_moment_callback);
75     glfwSetJoystickEnergyCallback(window, joystick_energy_callback);
76     glfwSetJoystickPowerFactorCallback(window, joystick_power_factor_callback);
77     glfwSetJoystickReactivePowerCallback(window, joystick_reactive_power_callback);
78     glfwSetJoystickComplexPowerCallback(window, joystick_complex_power_callback);
79     glfwSetJoystickApparentPowerCallback(window, joystick_apparent_power_callback);
80     glfwSetJoystickRealPowerCallback(window, joystick_real_power_callback);
81     glfwSetJoystickComplexPowerFactorCallback(window, joystick_complex_power_factor_callback);
82     glfwSetJoystickReactivePowerFactorCallback(window, joystick_reactive_power_factor_callback);
83     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
84     glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
85     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
86     glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
87     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
88     glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
89     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
90     glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
91     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
92     glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
93     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
94     glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
95     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
96     glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
97     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
98     glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
99     glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
100    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
101    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
102    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
103    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
104    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
105    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
106    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
107    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
108    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
109    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
110    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
111    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
112    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
113    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
114    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
115    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
116    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
117    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
118    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
119    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
120    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
121    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
122    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
123    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
124    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
125    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
126    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
127    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
128    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
129    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
130    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
131    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
132    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
133    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
134    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
135    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
136    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
137    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
138    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
139    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
140    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
141    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
142    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
143    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
144    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
145    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
146    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
147    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
148    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
149    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
150    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
151    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
152    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
153    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
154    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
155    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
156    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
157    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
158    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
159    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
160    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
161    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
162    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
163    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
164    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
165    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
166    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
167    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
168    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
169    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
170    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
171    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
172    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
173    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
174    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
175    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
176    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
177    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
178    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
179    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
180    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
181    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
182    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
183    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
184    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
185    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
186    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
187    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
188    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
189    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
190    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
191    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
192    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
193    glfwSetJoystickComplexPowerFactorAngleCallback(window, joystick_complex_power_factor_angle_callback);
194    glfwSetJoystickReactivePowerFactorAngleCallback(window, joystick_reactive_power_factor_angle_callback);
195    glfw
```

Si descomentamos todas las funciones obtenemos:



Conclusión

En esta práctica comprendimos cómo manipular los vértices y las líneas mediante la modificación de los valores en el arreglo, y cómo es posible alterar el orden de ejecución invirtiendo la secuencia de estos elementos.

Referencias:

NTecnologías Interactivas y Computación Gráfica. (2024, 9 de agosto). Dibujo de Primitivas en 2D con Shaders en OpenGL [Video]. YouTube.

<https://www.youtube.com/watch?v=UWcnvhSDYYA>

GitHub:

https://github.com/dayanapa45/Curso_Compu_Grafica