Índice General

Solución Directa

Nota Rápida

Se dará solución al código de nombre `B03_FigsPrecargadas02.py` brindado por el maestro Padilla, además, esta solución aplica para "HASTA AHORA" Windows 11, pues es el sistema operativo empleado para la solución al problema, si su ordenador emplea otro SO, comentarlo a la clase.

¿Qué Modifico Del Código?

Solo véase la función `def iniciar_ventana()` dentro del código del maestro, actualmente se encuentra de la siguiente forma:

```
def iniciar_ventana():
    if not glfw.init():
        raise Exception("No se pudo iniciar GLFW")
    ventana = glfw.create_window(800, 600, "Formas Básicas con transformaciones", None, None)
    if not ventana:
        glfw.terminate()
        raise Exception("No se pudo crear la ventana")
    glfw.make_context_current(ventana)

    # Activar buffer de profundidad, necesario para trabajar en 3D y calcular la profundidad de los
    # objetos en la escena.
    glEnable(GL_DEPTH_TEST)
    glutInit()                      # Inicializar GLUT para usar figuras precargadas
    return ventana
```

PARTE DEL CAUSANTE DEL ERROR, EL "glutInit()"

Lo único que deberá de realizar es eliminar la inicialización de GLUT anterior e iniciar primero en la función (`glutInit()`) y después una ventana mediante GLUT mediante…

```
glutCreateWindow(b"inserte aqui el título de su Ventana ")
```

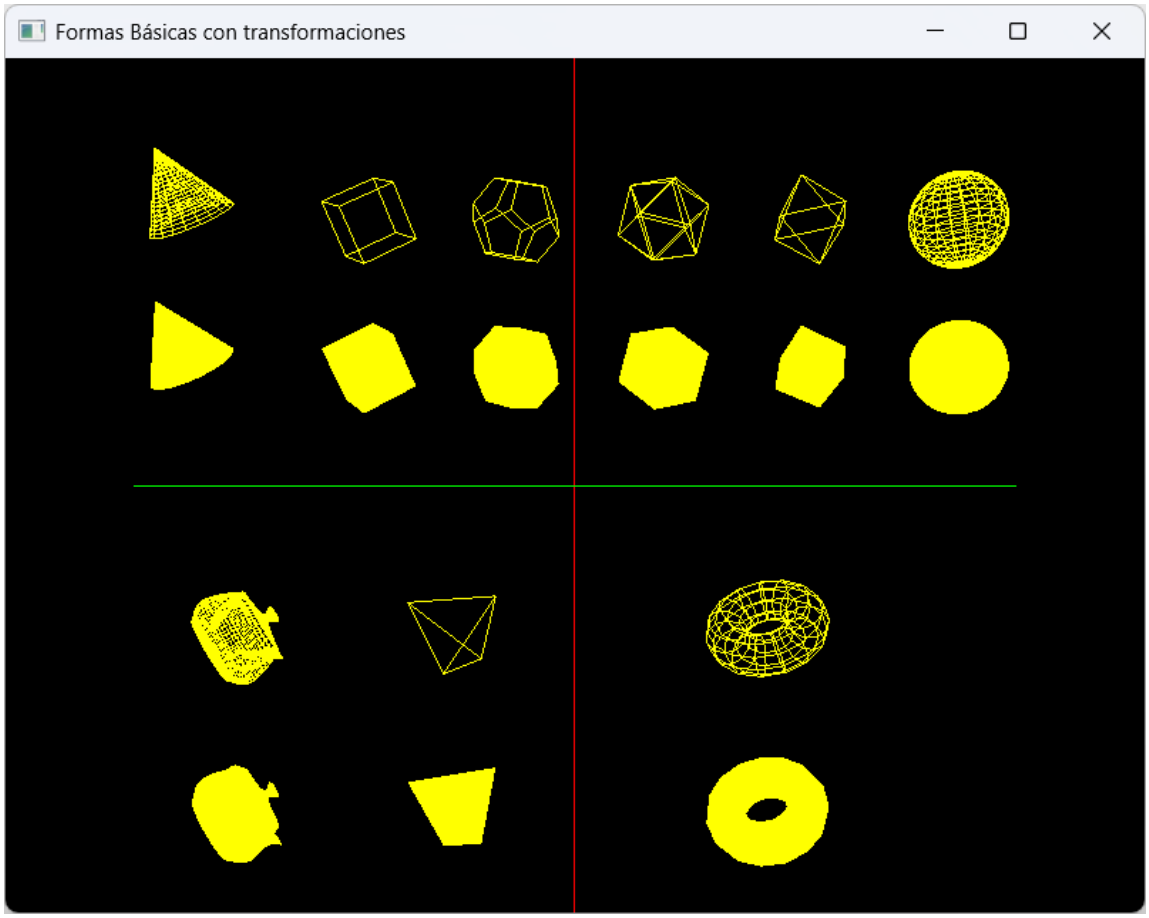
Obsérvese la siguiente imagen:

```
def iniciar_ventana():
    glutInit()                          # Inicializar GLUT para usar figuras precargadas
    glutCreateWindow(b"Ventana inutil de GLUT")

    if not glfw.init():
        raise Exception("No se pudo iniciar GLFW")
    ventana = glfw.create_window(800, 600, "Formas Básicas con transformaciones", None, None)
    if not ventana:
        glfw.terminate()
        raise Exception("No se pudo crear la ventana")
    glfw.make_context_current(ventana)

    # Activar buffer de profundidad, necesario para trabajar en 3D y calcular la profundidad de los
    # objetos en la escena.
    glEnable(GL_DEPTH_TEST)
    return ventana
```

Lo encerrado ahora en el recuadro rojo es el código nuevo que inserto (cabe indicar que debe eliminar el código anterior mencionado). Listo, ahora ejecute el programa y deberá salir algo como lo siguiente:



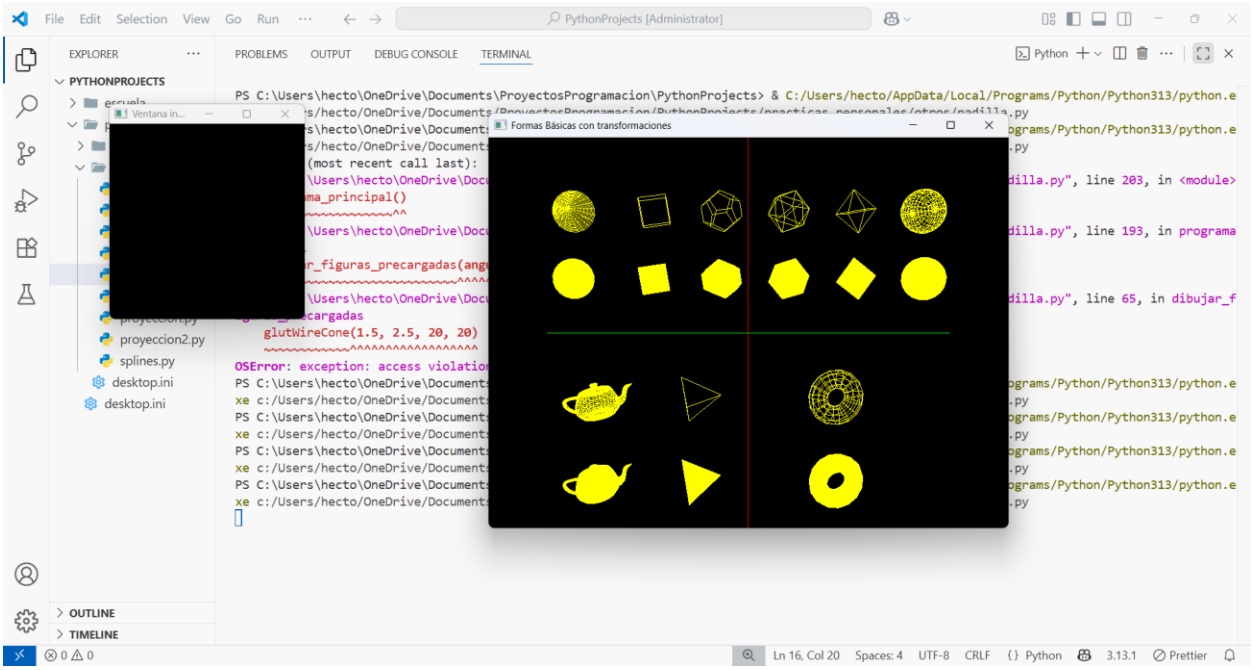
Explicación Breve

A lo que pudo entender el autor de este documento, es necesario inicializar la ventana de GLUT para que fuerce la inicialización de los recursos de dibujo de GLUT, por ende, si usted nunca dibuja ninguna figura 3D (mediante GLUT) ¡nunca le marcará error! Pero si se le ocurre dibujar un cono, cuadrado, etc., mediante GLUT, lo hará, le dará un error.

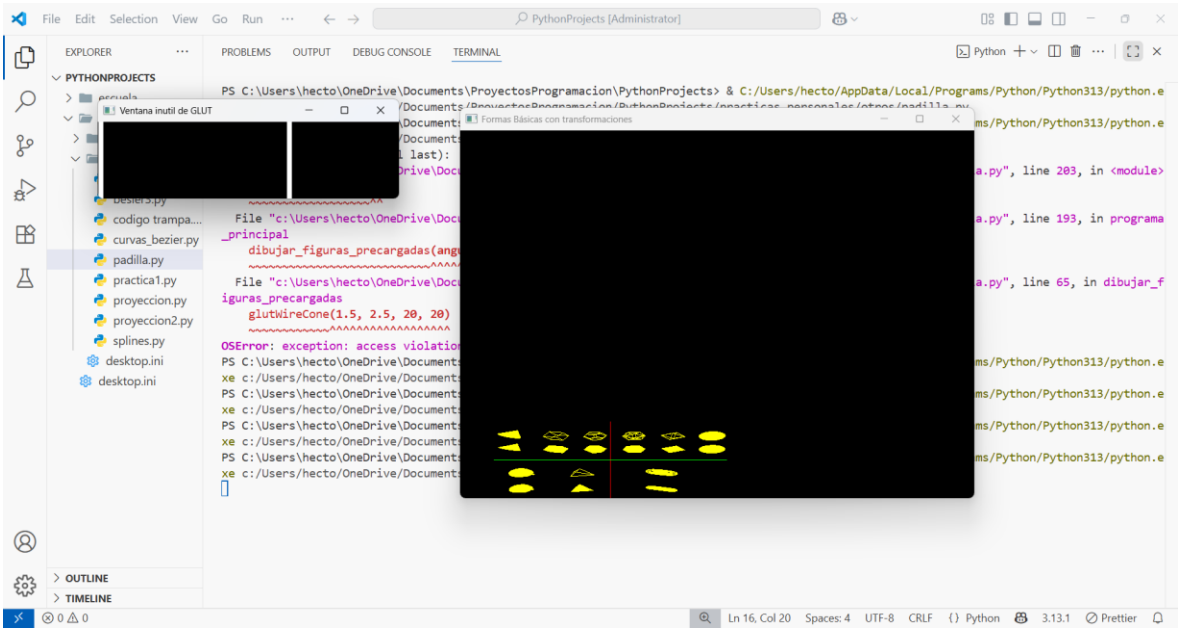
Desventajas

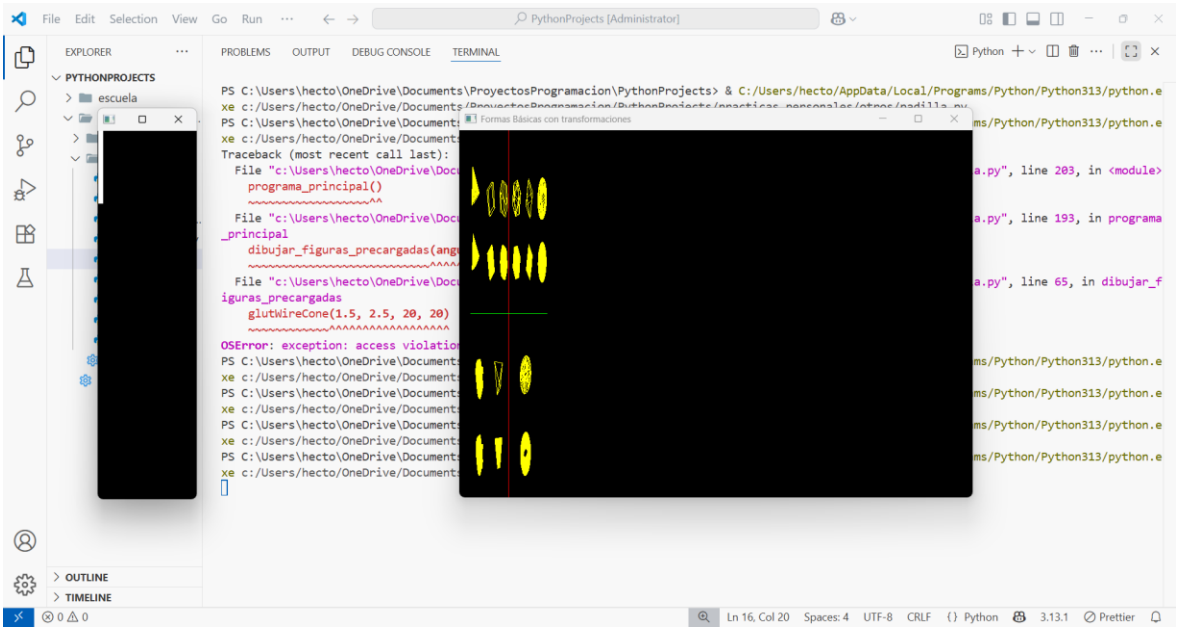
A continuación se enlista las desventajas que se encontraron:

1. Creación de una pequeña ventana: Como se indica a GLUT que cree su propia ventana, la hace y ahora aperen 2 ventanas (pero solo se dibuja dentro de la ventana hecha con GLFW. Se anexa imagen del error:



2. Redimensionamiento de ventanas: Si usted intenta cambiar el tamaño de la ventana hecha por GLFW no pasará nada, pero si cambia el tamaño de la ventana hecha por GLUT, el dibujo que contiene la ventana GLFW se deformará, vea la siguiente imagen, en donde se cambió el ancho y alto de la ventana GLUT, (pareciera redibujarse el contenido de la ventana GLFW a las dimensiones de la ventana GLUT:





3. Cierre de ventanas: Si cierra la ventana GLUT o la ventana GLFW se cierran ambas ventanas (sin importar cual cierre primero).

Solución Para La Ventana Inútil De GLUT

Realmente no hay solución, intente varias alternativas como el esconder la ventana mediante `glutHideWindow()`, después posicionar la ventana fuera de las dimensiones de la pantalla (la laptop del autor en este caso) con `glutInitWindowPosition()`, después cerrar la ventana GLUT antes de crear la ventana GLFW con `glutDestroyWindow(identificador_de_ventana)` y tampoco, pero algo que pude notar es que sí funciona al declararse antes de la creación de la ventana (aquellas que tienen "init" en su nombre nada más, que son funciones que alteran la ventana antes de que se inicialicen), entonces el código quedo de la siguiente forma:

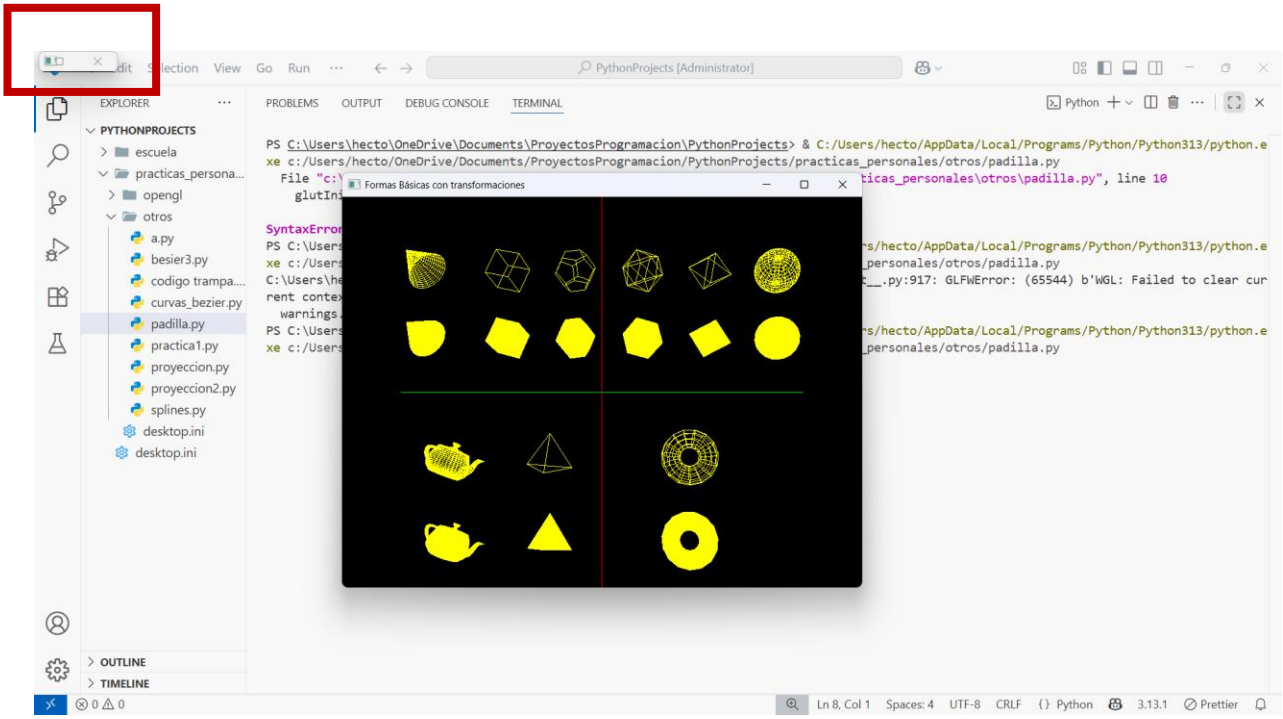
```
def iniciar_ventana():
    glutInit()                      # Inicializar GLUT para usar figuras precargadas
    glutInitWindowSize(1, 1)        # Menciona que el tamaño de la ventana es de 1x1 (ancho, alto) pixeles
    glutInitWindowPosition(0,0)     # Posiciona la ventana en las coordenadas (0,0)
    glutCreateWindow(b"Ventana inutil de GLUT")    # Crea la ventana GLUT

    if not glfw.init():
        raise Exception("No se pudo iniciar GLFW")
    ventana = glfw.create_window(800, 600, "Formas Básicas con transformaciones", None, None)
    if not ventana:
        glfw.terminate()
        raise Exception("No se pudo crear la ventana")
    glfw.make_context_current(ventana)

    # Activar buffer de profundidad, necesario para trabajar en 3D y calcular la profundidad de los
    # objetos en la escena.
    glEnable(GL_DEPTH_TEST)
    return ventana
```

El recuadro rojo refiere a la actualización.

Se intento borrar los bordes de la ventana, en java se le conoce como “undecorated” (vaya, con borde se refiere a el titulo por ejemplo, eliminar dicha barra), pero igual ni le funciono al autor. Al ejecutar verá la ventana en pequeño y arriba a la izquierda:

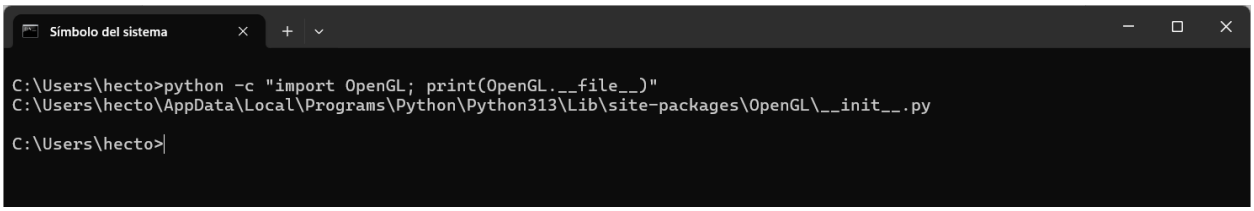


¿Como se llegó a la solución?

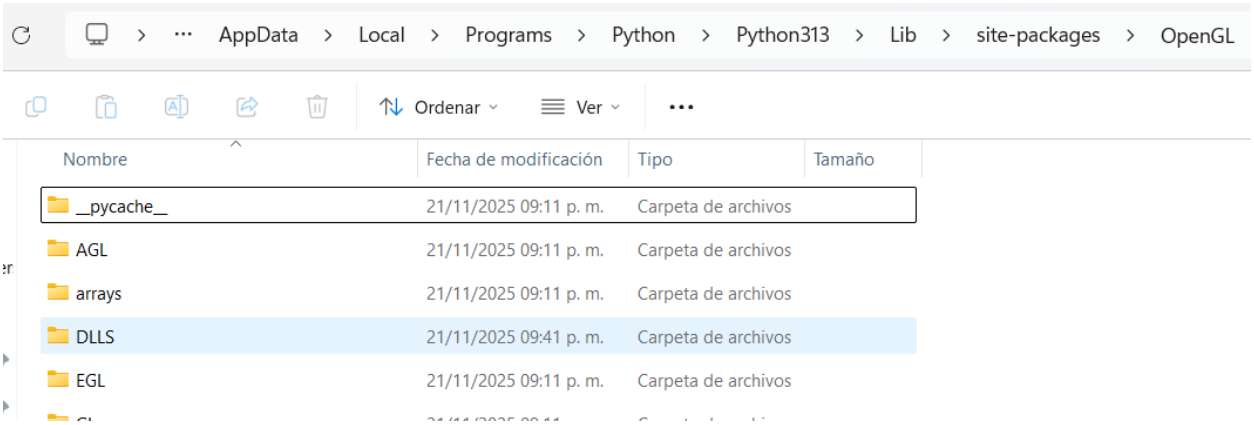
Menciona el autor: “Realmente intente la instalación de Free GLUT, descargando archivos de .dll, reinstalando OpenGl, etc, pero simplemente descubrí que todo el tiempo estaba instalada la librería”, el autor descubrió esto accediento directamente a las librerías de OpenGL, para que pueda acceder Introduzca en CMD el siguiente comando:

python -c "import OpenGL; print(OpenGL.__file__)"

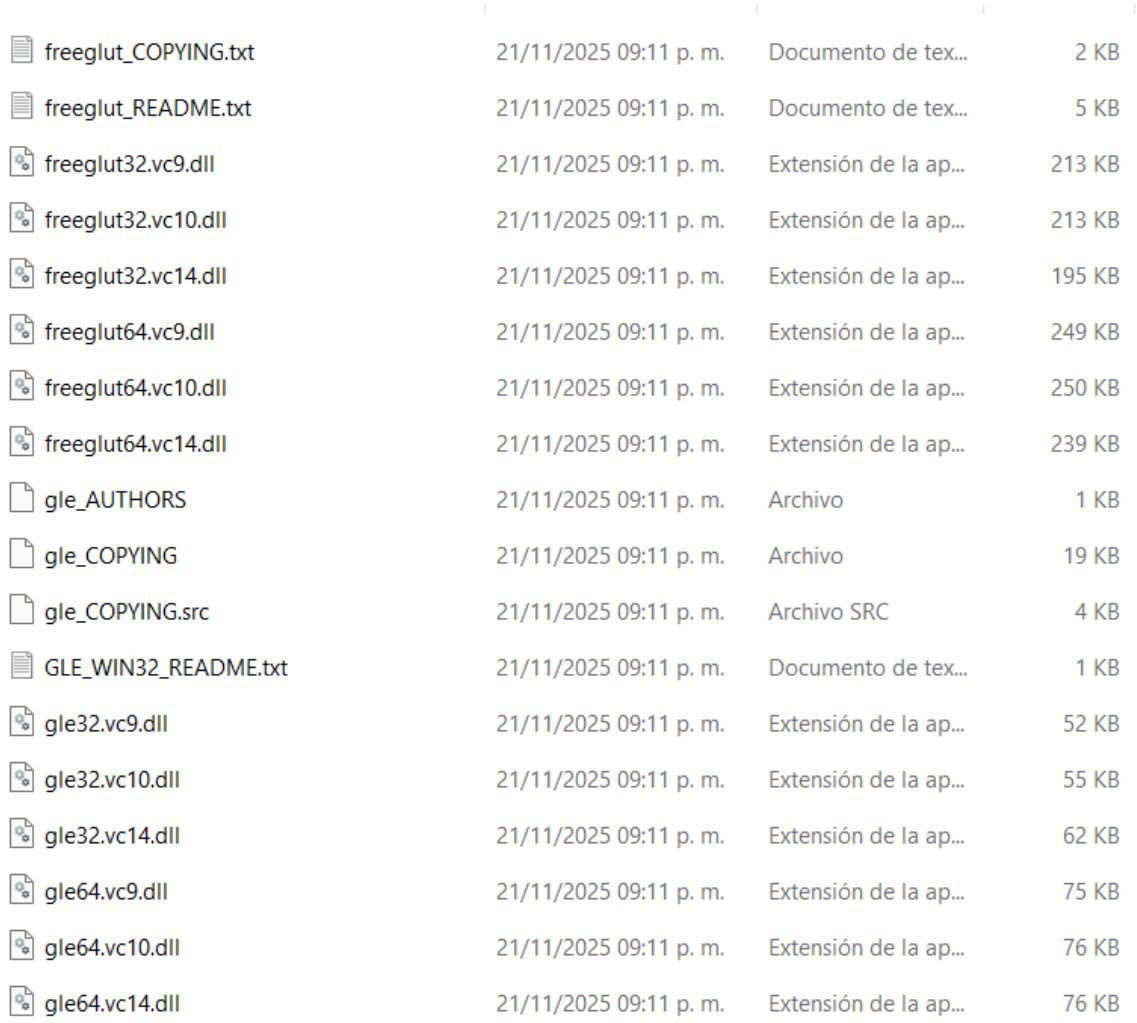
Esto le Brinda lo siguiente:



Una ruta, acceda a ella a través de su explorador de archivos (pero eliminando la última parte de “__init__.py”, notará lo siguiente:



Acceda a “DLLS” y vera todas las DLL que tiene su OpenGL, como en la siguiente imagen:



freeglut_COPYING.txt	21/11/2025 09:11 p. m.	Documento de tex...	2 KB
freeglut_README.txt	21/11/2025 09:11 p. m.	Documento de tex...	5 KB
freeglut32.vc9.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	213 KB
freeglut32.vc10.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	213 KB
freeglut32.vc14.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	195 KB
freeglut64.vc9.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	249 KB
freeglut64.vc10.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	250 KB
freeglut64.vc14.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	239 KB
gle_AUTHORS	21/11/2025 09:11 p. m.	Archivo	1 KB
gle_COPYING	21/11/2025 09:11 p. m.	Archivo	19 KB
gle_COPYING.src	21/11/2025 09:11 p. m.	Archivo SRC	4 KB
GLE_WIN32_README.txt	21/11/2025 09:11 p. m.	Documento de tex...	1 KB
gle32.vc9.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	52 KB
gle32.vc10.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	55 KB
gle32.vc14.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	62 KB
gle64.vc9.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	75 KB
gle64.vc10.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	76 KB
gle64.vc14.dll	21/11/2025 09:11 p. m.	Extensión de la ap...	76 KB

Ahí obsérvese que existen archivos “freeglut”. ¿Entonces que hizo el autor? Nada más genero un código con unicamente GLUT y OpenGL (por que el autor no sabe usar GLUT), donde fue observando cómo funcionaba el código sin GLFW, entonces fue reemplazando líneas y etc., hasta encontrarse con la creación de la ventana, y listo ¡funciono!. A continuación se anexa el código generado el cual dibuja una esfera de alambre, si interesa su análisis:

```
import glfw
import OpenGL.GL as gl
import OpenGL.GLUT as glut
import OpenGL.GLU as glu
import time

# Variable global para la rotación animada (opcional, para que la veas
girar)
rotation_angle = 0.0

# 1. Función de Visualización
def dibujar():
    """Función principal de dibujo llamada por GLUT."""
    global rotation_angle

    gl.glClear(gl.GL_COLOR_BUFFER_BIT | gl.GL_DEPTH_BUFFER_BIT) #
Limpia buffers
    gl.glLoadIdentity() # Reinicia la matriz ModelView

    # Traslación de la cámara/objeto
    # Mueve la cámara hacia atrás para ver la figura
    gl.glTranslatef(0.0, 0.0, -5.0)

    # Rotación animada de la esfera (para verla en 3D)
    gl.glRotatef(rotation_angle, 0, 1, 0) # Rota sobre el eje Y

    # **Configuración del color para las líneas**
    gl.glColor3f(0.0, 0.8, 1.0) # Color azul claro

    # **Dibuja la Esfera Alambrada**
    # Parametros: glutWireSphere(radio, rebanadas, anillos)
    # radio: 1.0
    # rebanadas (slices): 20 (número de divisiones alrededor del eje
Z)
    # anillos (stacks): 20 (número de divisiones a lo largo del eje Z)
    glut.glutWireSphere(1.0, 20, 20)

    glut.glutSwapBuffers() # Intercambia los buffers para mostrar la
imagen

    # Actualiza el ángulo para la animación
    rotation_angle += 0.5 # Incrementa la rotación
    time.sleep(0.01) # Pequeña pausa para controlar la velocidad

# 2. Configuración Inicial
def inicializar():
```

```
    """Configuración inicial de OpenGL."""
    # Habilita el test de profundidad
    gl.glEnable(gl.GL_DEPTH_TEST)

    # Color de fondo de la ventana (negro)
    gl.glClearColor(0.0, 0.0, 0.0, 1.0)

    # Configuración de la matriz de proyección (perspectiva)
    gl.glMatrixMode(gl.GL_PROJECTION)
    gl.glLoadIdentity()
    # Perspectiva: 45 grados FOV, relación de aspecto, 0.1 cerca, 50.0
lejos
    glu.gluPerspective(45, 800/600, 0.1, 50.0)

    # Vuelve a la matriz de ModelView
    gl.glMatrixMode(gl.GL_MODELVIEW)
    gl.glLoadIdentity()

# 3. Función principal (Main)
if __name__ == "__main__":
    glut.glutInit() # Inicializa GLUT

    # Modo de visualización: doble buffer, RGB y test de profundidad
    glut.glutInitDisplayMode(glut.GLUT_DOUBLE | glut.GLUT_RGB |
glut.GLUT_DEPTH)

    # Tamaño de la ventana
    glut.glutInitWindowSize(800, 600)

    # Crea la ventana con un título
    glut.glutCreateWindow(b"Esfera 3D con glutWireSphere")
    # Llama a la función de inicialización de OpenGL
    inicializar()

    # Registra la función 'dibujar' como la función de visualización
    glut.glutDisplayFunc(dibujar)

    # Llama a la función 'dibujar' continuamente para la animación
    glut.glutIdleFunc(dibujar)

    # Entra en el bucle principal de GLUT
    glut.glutMainLoop()
```