from mnemonic import Mnemonic

import hashlib

from solders.keypair import Keypair

import base58

from solders.pubkey import Pubkey

import sys

# Reemplaza con tu frase semilla de 12 palabras

mnemonic\_phrase = "<escriba su frase semilla aqui>"

# Inicializa el generador de mnemónicos para inglés

mnemonic\_generator = Mnemonic('english')

try:

    # Imprimir el sys.path

    print(sys.path)

    print("Solana importado exitosamente")

    # 1. Generar la semilla (con opción de passphrase)

    passphrase = input("Introduce la passphrase (déjala vacía si no tiene): ")

    seed\_bytes = mnemonic\_generator.to\_seed(mnemonic\_phrase, passphrase=passphrase)

    # \*\*Importante:\*\* solders.Keypair.from\_seed() espera 32 bytes

    # La semilla generada por mnemonic.to\_seed() es de 64 bytes.

    # Para Solana, generalmente se usan los primeros 32 bytes.

    seed\_32\_bytes = seed\_bytes[:32]

    # 2. Generar el Keypair de Solana directamente desde los bytes de la semilla

    keypair = Keypair.from\_seed(seed\_32\_bytes)

    # Obtener la clave pública (de solders)

    public\_key\_solders = keypair.pubkey()

    print(f"Clave pública (Pubkey): {public\_key\_solders}")

    print(f"Clave pública en base-58 (solders): {public\_key\_solders}")

    # \*\*Obtener la clave privada completa (64 bytes) directamente del objeto Keypair\*\*

    full\_private\_key\_bytes = bytes(keypair)

    # Codificar la clave privada completa a Base58

    full\_private\_key\_base58 = base58.b58encode(full\_private\_key\_bytes).decode('utf-8')

    print(f"Clave privada completa (64 bytes) en base-58 (solders): {full\_private\_key\_base58}")

except Exception as e:

    print(f"Error al generar el Keypair de Solana: {e}")

    print("Asegúrate de que la frase semilla y la passphrase sean correctas.")