import hashlib  
import time  
  
def proof\_of\_work(data, difficulty=2):  
    """  
    A simple Proof of Work function that finds a nonce  
    such that the hash has 'difficulty' leading zero pairs.  
     
    :param data: The transaction data or block data to be hashed  
    :param difficulty: Number of leading zero pairs (e.g., difficulty=2 means "0000" at the start)  
    :return: (nonce, hash\_value, time\_taken)  
    """  
    nonce = 0  
    prefix = '0' \* (difficulty \* 2)# Two zeros per difficulty level  
    print(prefix)  
    start\_time = time.time()  
  
    while True:  
        text = f"{data}{nonce}".encode()  
        hash\_value = hashlib.sha256(text).hexdigest()  
        print(hash\_value)  
         
        if hash\_value.startswith(prefix):  
            end\_time = time.time()  
            return nonce, hash\_value, round(end\_time - start\_time, 4)  # Return the nonce, hash, and time taken  
  
        nonce += 1  # Increment nonce and retry  
  
# Example Usage  
data = "Shahbaz"  
difficulty = 2  # Require three pairs of leading zeros (i.e., "000000")  
nonce, final\_hash, time\_taken = proof\_of\_work(data, difficulty)  
  
print(f"✅ Proof of Work Completed!")  
print(f"Nonce Found: {nonce}")  
print(f"Hash: {final\_hash}")  
print(f"Time Taken: {time\_taken} seconds")