import random

import hashlib

# Modular inverse function

def mod\_inverse(a, m):

for x in range(1, m):

if (a \* x) % m == 1:

return x

return None

# Hash function using SHA-1

def hash\_message(message):

h = hashlib.sha1()

h.update(message.encode('utf-8'))

return int(h.hexdigest(), 16)

# Step 1: Parameters (in real-world use large primes!)

p = 23 # prime modulus

q = 11 # prime divisor of p-1

g = 2 # generator

# Step 2: Key generation

x = random.randint(1, q-1) # private key

y = pow(g, x, p) # public key

print(f"Public Key (y): {y}")

print(f"Private Key (x): {x}")

# Step 3: Message input

message = input("\nEnter message to sign: ")

hm = hash\_message(message) % q

# Step 4: Signing

k = random.randint(1, q-1)

r = pow(g, k, p) % q

if r == 0:

raise Exception("Invalid r, choose another k")

k\_inv = mod\_inverse(k, q)

s = (k\_inv \* (hm + x \* r)) % q

if s == 0:

raise Exception("Invalid s, choose another k")

print(f"\nSignature: (r={r}, s={s})")

# Step 5: Verification

w = mod\_inverse(s, q)

u1 = (hm \* w) % q

u2 = (r \* w) % q

v = ((pow(g, u1, p) \* pow(y, u2, p)) % p) % q

print("\nVerifying signature...")

if v == r:

print("✅ Signature is valid!")

else:

print("❌ Signature is invalid.")