

## AN TOÀN THÔNG TIN

### BÀI 1 VD MINH HỌA

# An Toán Thông Tin

Date:

Bài 1 Tìm nghịch đảo Euclid

Đề Bài

$$99 \cdot y \equiv 1 \pmod{m} \quad \text{giá trị } m = 101$$

(1) Áp dụng giải thuật

$$101 = 99 \cdot 1 + 2$$

$$99 = 2 \cdot 49 + 1$$

$$2 = 1 \cdot 2 + 0$$

$\Rightarrow$  Dư  $= 0$ , UCLN là số cuối cùng trước đó  
 $\gcd(99, 101) = 1 \Rightarrow$  nghịch đảo 0 tồn tại

(2) Áp dụng Euclid mở rộng để tìm  $x, y$

$$101 \cdot x + 99y = 1$$

$$i = r_i \quad r_i + 1 \quad r_i + 2 \quad q_{i+1} \quad x_i \quad x_{i+1}$$

$$r_i + 2$$

~~$$r_i \quad r_i + 1 \quad r_i + 2 \quad q_{i+1} \quad x_i \quad x_{i+1}$$~~

Date:

$n_i$	$n_{i+1}$	$n_{i+2}$	$q_{i+1}$	$x_i$	$x_{i+1}$	$x_{i+2}$	$y_i$	$y_{i+1}$	$y_{i+2}$
101	99	2	-1	1	0	1	8	1	-1
99	2	1	49	0	1	-49	1	-1	50
2	1	0	2	1	-49	99	-1	50	-101

Khoi tạo  $x_0 = 1, x_1 = 0; y_0 = 0, y_1 = 1$

$$\text{Tacó } x_{i+2} = x_i - q_{i+1} \cdot x_{i+1}$$

$$y_{i+2} = y_i - q_{i+1} \cdot y_{i+1}$$

$$\Rightarrow x = 99 \cdot 49 = -101$$

$$101 \cdot 99 + 99(-101) = 101 \cdot 99 - 99 \cdot 101$$

$$\Rightarrow = 0 + 1 = 1$$

$\Rightarrow$  Đúng ✓

4. Tìm nghịch đảo

Nghịch đảo của 99 modulo 101 là  $y = -101$

$$-101 \bmod 101 = -101 + 101 = 0$$

$$y = 50$$

$$99 \cdot 50 = 4950 \cdot 4950 \bmod 101$$

$$= 4950 - 101 \cdot 49 = 4950 - 4949 = 1$$

$\Rightarrow$  Nghịch đảo của 99 modulo 101 là 50

CODE :

```
main.py
8 y = x1
9 return gcd, x, y
10
11 def mod_inverse(a, m):
12     gcd, x, _ = extended_gcd(a, m)
13     if gcd != 1:
14         return None
15     return (x % m + m) % m
16
17 a, m = 3, 11
18 gcd, x, y = extended_gcd(a, m)
19 inverse = mod_inverse(a, m)
20
21 print(f>Date: {datetime.now().strftime('%Y-%m-%d')}")
22 print(f>Extended GCD của {a} và {m}: GCD = {gcd}, x = {x}, y = {y}")
23 print(f>Nghịch đảo modulo của {a} mod {m}: {inverse}")
Ln: 1, Col: 1
Run Share Command Line Arguments
Date: 2025-05-05
Extended GCD của 3 và 11: GCD = 1, x = 4, y = -1
Nghịch đảo modulo của 3 mod 11: 4
** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

from datetime import datetime

def extended\_gcd(a, b):

if a == 0:

return b, 0, 1

gcd, x1, y1 = extended\_gcd(b % a, a)

x = y1 - (b // a) \* x1

y = x1

return gcd, x, y

def mod\_inverse(a, m):

gcd, x, \_ = extended\_gcd(a, m)

if gcd != 1:

return None

```
return (x % m + m) % m
```

```
a, m = 3, 11
```

```
gcd, x, y = extended_gcd(a, m)
```

```
inverse = mod_inverse(a, m)
```

```
print(f"Date: {datetime.now().strftime('%Y-%m-%d')}")
```

```
print(f"Extended GCD của {a} và {m}: GCD = {gcd}, x = {x}, y = {y}")
```

```
print(f"Ngược đảo modulo của {a} mod {m}: {inverse}")
```

BÀI 2:

Bài 2 MSSV 0542050099 74 <sup>Date:</sup>

$$74 \% 6 = 2$$

⇒ 2 sẽ là mã Affine

Mã Hoại "NG"

Chữ N ( $x = 13$ , vì N là chữ thứ 14 trong BCC 0-based)

- Công thức  $y = (5 \cdot x + 8) \bmod 26$
- Tính  $y = 5 \cdot 13 = 65, 65 + 8 = 73$
- Modulo  $73 \bmod 26 = 73 - 2 \cdot 26 = 73 - 52 = 21$
- Kết quả  $y = 21 \rightarrow V$  (V là chữ thứ 22)

Chữ G ( $x = 6$ , vì G là chữ thứ 7)

- Tính  $5 \cdot 6 = 30, 30 + 8 = 38$
- Modulo:  $38 \bmod 26 = 38 - 1 \cdot 26 = 38 - 26 = 12$
- Kết quả  $y = 12 \rightarrow M$  (M là chữ thứ 13)

⇒ Kết quả của Mã Hoại là VM

⇒ NG  $\rightarrow$  VM



Date:

Giải mã "NM" với  $a=5$ ,  $b=8$

Tìm  $a^{-1}$  (nghịch đảo modulo của 5 mod 26)

• Dùng Euclidean mở rộng để tìm  $x$  sao cho  
 $5x + 26y = 1$

$$26 = 5 \cdot 5 + 1$$

$$1 = 26 - 5 \cdot 5$$

$$\text{Hệ số } x = -5, y = 1$$

$$\text{Nghịch đảo } x = -5 \bmod 26 = -5 + 26 = 21$$

$$\text{Vậy } a^{-1} = 21$$

$$\text{Kiểm tra } 5 \cdot 21 = 105, 105 \bmod 26 = 105 - 4 \cdot 26 \\ = 105 - 104 = 1$$

Chữ V ( $y=21$ )

$$x = 21 \cdot (y - b) \bmod 26$$

$$y - b = 21 - 8 = 13$$

$$21 \cdot 13 = 273$$

$$\Rightarrow \text{Modulo } 273 \bmod 26 = 273 - 10 \cdot 26 \\ = 13$$

$\rightarrow N$

• Chữ M ( $q = 12$ )

Date:

$$\text{Tính } y - b = 12 - 8 = 4$$

$$\cdot 21. 4 = 84$$

$$\text{Modulo } 84 \bmod 26 = 84 - 3 \cdot 26 = 6$$

$$\Rightarrow x = 6 \rightarrow \text{G}$$

KQ giải mã "VM" = "NG"

CODE:



The screenshot shows an online Python IDE with a dark theme. The editor displays a file named `main.py` containing an affine cipher implementation. The code defines `affine_encrypt` and `affine_decrypt` functions, and uses them to encrypt and decrypt the text "NG". A tooltip with the text "Help Us Keep the Code Flowing!!" is visible over the code. Below the editor, there are buttons for "Run", "Share", and "Command Line Arguments". The output panel at the bottom shows the execution results: "Date: 2025-05-05", "Original: NG", "Encrypted: VM", "Decrypted: NG", and "\*\* Process exited - Return Code: 0 \*\*".

```
main.py
39 for char in cipher.upper():
40     if char.isalpha():
41         y = ord(char) - ord('A')
42         x = (a_inv * (y - b)) % 26
43         result += chr(x + ord('A'))
44     else:
45         result += char
46     return result
47
48 # Ví dụ với "NG"
49 text = "NG"
50 a, b = 5, 8
51 encrypted = affine_encrypt(text, a, b)
52 decrypted = affine_decrypt(encrypted, a, b)
53
54 # Print the results
55 print(f"Original: {text}, Encrypted: {encrypted}, Decrypted: {decrypted}")
56 print(f"Date: {datetime.datetime.now().strftime('%Y-%m-%d')}")
```

Ln: 57

Run Share \$ Command Line Arguments

Date: 2025-05-05  
Original: NG  
Encrypted: VM  
Decrypted: NG

\*\* Process exited - Return Code: 0 \*\*

```
from datetime import datetime
```

```
def extended_gcd(a, b):
```

```
    """Tìm GCD và hệ số x, y sao cho ax + by = gcd(a, b)"""
```

```
    if a == 0:
```

```
        return b, 0, 1
```

```
    gcd, x1, y1 = extended_gcd(b % a, a)
```

```
    x = y1 - (b // a) * x1
```

```
    y = x1
```

```
    return gcd, x, y
```

```
def mod_inverse(a, m):
```

```
    """Tìm nghịch đảo modulo của a mod m"""
```

```
    gcd, x, _ = extended_gcd(a, m)
```

```
if gcd != 1:
    return None
return (x % m + m) % m
```

```
def affine_encrypt(text, a, b):
    """Mã hóa Affine:  $y = (ax + b) \bmod 26$ """
    if mod_inverse(a, 26) is None:
        return "a phải nguyên tố cùng 26"
    result = ""
    for char in text.upper():
        if char.isalpha():
            x = ord(char) - ord('A')
            y = (a * x + b) % 26
            result += chr(y + ord('A'))
        else:
            result += char
    return result
```

```
def affine_decrypt(cipher, a, b):
    """Giải mã Affine:  $x = a^{-1} * (y - b) \bmod 26$ """
    a_inv = mod_inverse(a, 26)
    if a_inv is None:
        return "a phải nguyên tố cùng 26"
    result = ""
    for char in cipher.upper():
        if char.isalpha():
```

```
    y = ord(char) - ord('A')
    x = (a_inv * (y - b)) % 26
    result += chr(x + ord('A'))
else:
    result += char
return result
```

# Ví dụ với "NG"

```
text = "NG"
```

```
a, b = 5, 8
```

```
encrypted = affine_encrypt(text, a, b)
```

```
decrypted = affine_decrypt(encrypted, a, b)
```

```
print(f"Date: {datetime.now().strftime('%Y-%m-%d')}")
```

```
print(f"Original: {text}")
```

```
print(f"Encrypted: {encrypted}")
```

```
print(f"Decrypted: {decrypted}")
```

Date:

Bài 3 AES 128-bit vs Khối 128-bit

Mô hình

Mã hóa vs mã Chứa "NG" vs Khối 128-bit  
 "1234567890123456"

Thao tác AES (CBC):

- Văn bản: "NG" (2 byte), thêm (PKCS7) thành 16 byte (thêm 14 byte 0x14)
- Khối: "1234567890123456" (16 byte)
- IV: Ngẫu nhiên (VD: 0x1a2b3c4d5e6f7890abcdef1234567890)
- Mã hóa
- Khối 1: "NG" + 14 byte padding, XOR vs IV

Mã hóa AES (SubBytes, Shift Rows, Mix Columns, Add Round Key)

• K<sub>A</sub>: 16 byte mã

Giải mã: Giải mã khối, XOR vs IV, bỏ padding

Thay từ đầu vào  
 Thay "NG" bằng "74"; "Hi", hoặc  
 "SE"

vs Mã hóa sẽ khác như T toán K<sub>đ</sub>

CODE:

```
padding_len = padded_text[-1]
return padded_text[:-padding_len].decode()

key = b'1234567890123456'
plaintext = "SE"
iv, ciphertext = aes_encrypt(plaintext, key)
decrypted = aes_decrypt(iv, ciphertext, key)

print(f>Date: {datetime.now().strftime('%Y-%m-%d')}")
print(f>Plaintext: {plaintext}")
print(f>IV (hex): {iv.hex()}")
print(f>Ciphertext (hex): {ciphertext.hex()}")
print(f>Decrypted: {decrypted}")
```

```
Date: 2025-05-05
Plaintext: SE
IV (hex): 9240fecc5773aaa6b3c40bb9ec925540
Ciphertext (hex): f8fc55c46c53075f0944fe247d328b82
Decrypted: SE
```

```
from datetime import datetime
```

```
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
```

```
from cryptography.hazmat.backends import default_backend
```

```
import os
```

```
def aes_encrypt(plaintext, key):
```

```
    iv = os.urandom(16)
```

```
    cipher = Cipher(algorithms.AES(key), modes.CBC(iv), backend=default_backend())
```

```
    encryptor = cipher.encryptor()
```

```
    # Đệm thủ công (tương tự PKCS7)
```

```
    block_size = 16
```

```
    padded_text = plaintext.encode()
```

```
    padding_len = block_size - (len(padded_text) % block_size)
```



```
padded_text += bytes([padding_len] * padding_len)
ciphertext = encryptor.update(padded_text) + encryptor.finalize()
return iv, ciphertext
```

```
def aes_decrypt(iv, ciphertext, key):
    cipher = Cipher(algorithms.AES(key), modes.CBC(iv), backend=default_backend())
    decryptor = cipher.decryptor()
    padded_text = decryptor.update(ciphertext) + decryptor.finalize()
    padding_len = padded_text[-1]
    return padded_text[:-padding_len].decode()
```

```
key = b'1234567890123456'
plaintext = "SE"
iv, ciphertext = aes_encrypt(plaintext, key)
decrypted = aes_decrypt(iv, ciphertext, key)
```

```
print(f"Date: {datetime.now().strftime('%Y-%m-%d')}")
print(f"Plaintext: {plaintext}")
print(f"IV (hex): {iv.hex()}")
print(f"Ciphertext (hex): {ciphertext.hex()}")
print(f"Decrypted: {decrypted}")
```

BÀI 4:

Date:

Bài 4 Mã hóa Bài tập Đối Xứng  $74 \% 3$   
 $= 2 \Rightarrow$  RSA

$$74 \% 3 = 2 \Rightarrow \text{RSA}$$

Mã hóa và giải mã  $m = 74$  với khóa công khai  
( $e = 3, n = 100$ ), giá trị  $p = 5, q = 20$

Tạo khóa

- $n = p \cdot q = 5 \cdot 20 = 100$
- $\phi(n) = (p-1)(q-1) = 4 \cdot 19 = 76$
- Chọn  $e = 3$  ( $\text{gcd}(3, 76) = 1$ )
- Tính  $d$ :  $d \cdot 3 \equiv 1 \pmod{76}$
- Euclidean mở rộng  $76 = 25 \cdot 3 + 1, 1 = 76 - 25 \cdot 3$
- $d = -25 \pmod{76} = 51$
- Khóa công khai ( $e = 3, n = 100$ ), khóa riêng ( $d = 51, n = 100$ )

$$[\text{Mã hóa}] \quad c = m^e \pmod{n} = 74^3 \pmod{100}$$

(Tính)

$$74^2 = 5476 \pmod{100} = 76$$
$$74^3 = 74 \cdot 76 = 5624 \pmod{100} = 24$$

$$\text{Kq } c = 24$$

Giải mã

Date:

$$m = c^d \bmod n = 24^{51} \bmod 100$$

Bình phương và nhân

- $24^2 = 576 \bmod 100 = 76$
- $24^4 = 76^2 = 5776 \bmod 100 = 76$
- $24^8 = 76, 76, \dots, 24^{32} = 76$
- $51 = 32 + 16 + 2 + 1$ , nên  $24^{51} = 24^{32} \cdot 24^{16} \cdot 24^2 \cdot 24^1$
- $76 \cdot 76 \cdot 76 \cdot 24 = 76 \cdot 761824$
- $76 \cdot 76 = 5776 \bmod 100 = 76$
- $76 \cdot 1824 = 138624 \bmod 100 = 24$

KQ  $m = 74$

Thay đổi dấu vào

thay 74 bằng "NG" (mã chữ Văn bản)  
như RSA thay mã chữ Số, có thể chuyển  
"NG" thành Số ( $N = 13, G = 6 = 1306$ )

$$c = 1306^3 \bmod 100 \text{ tìm ngược từ}$$

```
main.py +
1  from datetime import datetime
2
3  def rsa_encrypt(m, e, n):
4      return pow(m, e, n)
5
6  def rsa_decrypt(c, d, n):
7      return pow(c, d, n)
8
9  m = 74
10 e, n = 3, 100
11 d = 51
12 c = rsa_encrypt(m, e, n)
13 decrypted = rsa_decrypt(c, d, n)
14
```

Share the code to your friends and colleagues

Run Share \$ Command Line Arguments

Date: 2025-05-05  
Message: 74  
Encrypted: 24  
Decrypted: 24

\*\* Process exited - Return Code: 0 \*\*  
Press Enter to exit terminal

```
from datetime import datetime
```

```
def rsa_encrypt(m, e, n):  
    return pow(m, e, n)
```

```
def rsa_decrypt(c, d, n):  
    return pow(c, d, n)
```

```
m = 74
```

```
e, n = 3, 100
```

```
d = 51
```

```
c = rsa_encrypt(m, e, n)
```

```
decrypted = rsa_decrypt(c, d, n)
```

```
print(f"Date: {datetime.now().strftime('%Y-%m-%d')}")
```

```
print(f"Message: {m}")
```

```
print(f"Encrypted: {c}")
```

```
print(f"Decrypted: {decrypted}")
```

BÀI 5:



Date:

(Task 5) Tạo chữ ký số

$$74 \% 5 = 4 \text{ ~~DSA~~ } \Rightarrow \text{DSA}$$

4 xác định thuật toán chữ ký số là DSA

VD Minh họa

Ký và xác minh thông qua  $m = 74$  với tham số  $p = 23$ ,  $q = 11$ ,  $g = 2$ , khóa riêng  $x = 6$

Tạo khóa

- Khóa công khai:  $y = g^x \bmod p = 2^6 \bmod 23 = 64 \bmod 23 = 18$

- Khóa riêng:  $x = 6$ , công khai  $y = 18$

Ký

$$H(m) = 74 \bmod 11 = 8 \text{ (giá trị băm đơn giản)}$$

Chọn  $k = 7$

- $r = (g^k \bmod p) \bmod q = (2^7 \bmod 23) \bmod 11 = 128 \bmod 23 = 13 \bmod 11 = 2$

- $k^{-1} \bmod q: 7x \equiv 1 \pmod{11}, x = 8$

(Vì  $7 \cdot 8 = 56 \equiv 1 \pmod{11}$ )

$$= S = k^{-1} (H(m) + x \cdot r) \bmod q = 8 (8 + 6 \cdot 2) \bmod 11 = 8 \cdot 20 \bmod 11 = 160 \bmod 11 = 6$$

• Chữ ký ( $r = 2, s = 6$ )

Xác Minh

$$• u = s^{-1} \bmod q = 6^{-1} \bmod 11 = 2 \quad (\text{vì } 6 \cdot 2 = 12 \equiv 1 \pmod{11})$$

$$• u_1 = H(m) \cdot u \bmod q = 8 \cdot 2 \bmod 11 = 16 \bmod 11 = 5$$

$$• u_2 = r \cdot u \bmod q = 2 \cdot 2 \bmod 11 = 4$$

$$• v = (g^{u_1} \cdot y^{u_2} \bmod p) \bmod q$$

$$• g^{u_1} = 2^5 \bmod 23 = 32 \bmod 23 = 9$$

$$• y^{u_2} = 18^4 \bmod 23 = 4$$

$$• g \cdot u = 36 \bmod 23 = 13$$

$$• v = 13 \bmod 11 = 2$$

$\Rightarrow v = r = 2$  Chữ ký hợp lệ  
Thay dấu vào = NG

Thay 74 = "NG" (Chuyển thành số hoặc chữ)

```
main.py +
56     return False
57     h = simple_hash(message)
58     w = mod_inverse(s, q)
59     u1 = (h * w) % q
60     u2 = (r * w) % q
61     v = ((pow(g, u1, p) * pow(y, u2, p)) % p) % q
62     return v == r
63
64 # Tạo khóa với tham số nhỏ
65 p = 23 # Số nguyên tố p
66 q = 11 # Số nguyên tố q, q là ước của p-1
67 g = 2 # g là số nguyên sao cho g^q ≡ 1 (mod p)
68 key = DSAKey(p, q, g)
69 public_key = key
70
Ln: 79, Col: 35
Run Share $ Command Line Arguments
Date: 2025-05-05
Message: 74
Signature: (9, 9)
Verification: True
>
** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

```
from datetime import datetime
```

```
import random
```

```
# Hàm tính GCD và nghịch đảo modulo
```

```
def gcd(a, b):
```

```
    while b:
```

```
        a, b = b, a % b
```

```
    return a
```

```
def mod_inverse(a, m):
```

```
    def extended_gcd(a, b):
```

```
        if a == 0:
```

```
            return b, 0, 1
```

```
        gcd, x1, y1 = extended_gcd(b % a, a)
```

```
        x = y1 - (b // a) * x1
```

```
y = x1
```

```
return gcd, x, y
```

```
_, x, _ = extended_gcd(a, m)
```

```
return (x % m + m) % m
```

```
# Hàm băm đơn giản (thay cho SHA256)
```

```
def simple_hash(message):
```

```
    # Một hàm băm đơn giản: chuyển message thành số và lấy modulo
```

```
    return int(str(message).encode().hex(), 16) % 11 # Giả sử q = 11
```

```
# Triển khai DSA thủ công
```

```
class DSAKey:
```

```
    def __init__(self, p, q, g):
```

```
        self.p = p # Số nguyên tố lớn
```

```
        self.q = q # Số nguyên tố nhỏ, q là ước của p-1
```

```
        self.g = g # g là một số nguyên sao cho  $g^q \equiv 1 \pmod{p}$ 
```

```
        self.x = random.randint(1, q-1) # Khóa riêng
```

```
        self.y = pow(g, self.x, p) # Khóa công khai:  $y = g^x \pmod{p}$ 
```

```
def dsa_sign(message, private_key):
```

```
    p, q, g, x = private_key.p, private_key.q, private_key.g, private_key.x
```

```
    # Băm thông điệp
```

```
    h = simple_hash(message)
```

```
    while True:
```

```
        k = random.randint(1, q-1) # Chọn k ngẫu nhiên
```

```

if gcd(k, q) != 1: # Đảm bảo k có nghịch đảo
    continue

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

if r == 0:
    continue

k_inv = mod_inverse(k, q)

s = (k_inv * (h + x * r)) % q

if s != 0:
    return (r, s)

```

```

def dsa_verify(message, signature, public_key):

    p, q, g, y = public_key.p, public_key.q, public_key.g, public_key.y

    r, s = signature

    if not (0 < r < q and 0 < s < q):
        return False

    h = simple_hash(message)

    w = mod_inverse(s, q)

    u1 = (h * w) % q

    u2 = (r * w) % q

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

    return v == r

```

# Tạo khóa với tham số nhỏ

p = 23 # Số nguyên tố p

q = 11 # Số nguyên tố q, q là ước của p-1

g = 2 # g là số nguyên sao cho  $g^q \equiv 1 \pmod{p}$



```
key = DSAKey(p, q, g)
public_key = key

# Ký và xác minh
message = 74
signature = dsa_sign(message, key)
is_valid = dsa_verify(message, signature, public_key)

print(f'Date: {datetime.now().strftime('%Y-%m-%d')}')
print(f'Message: {message}')
print(f'Signature: {signature}')
print(f'Verification: {is_valid}')
```

BÀI 6:

Date:

Bài 6 Hàm băm SHA - 256

Tính SHA - 256 của "NG"

Chuẩn toán SHA - 256

• Tiền xử lý

- "NG" = 01001110 01000111 (16 bits)
- Thêm bit 1: 01001110 01000111 1
- Thêm 0 để tổng chiều dài mod 512 = 448
- Thêm 429 ~~bits~~ bit 0
- Thêm độ dài 16 = 000...010000
- Nén

- Khởi tạo HC - H7 (chuẩn SHA - 256)
- Xử lý khối 512 - bit qua 64 vòng (Ch, Maj, Sigma)
- Kết quả: 256 - bit băm

• ~~Định ý: SHA - 256 phải tập 1 từ cùng cấp K quá~~

• ~~hi code~~ Thay đầu "74" "NG" = "74" hoặc "SE"

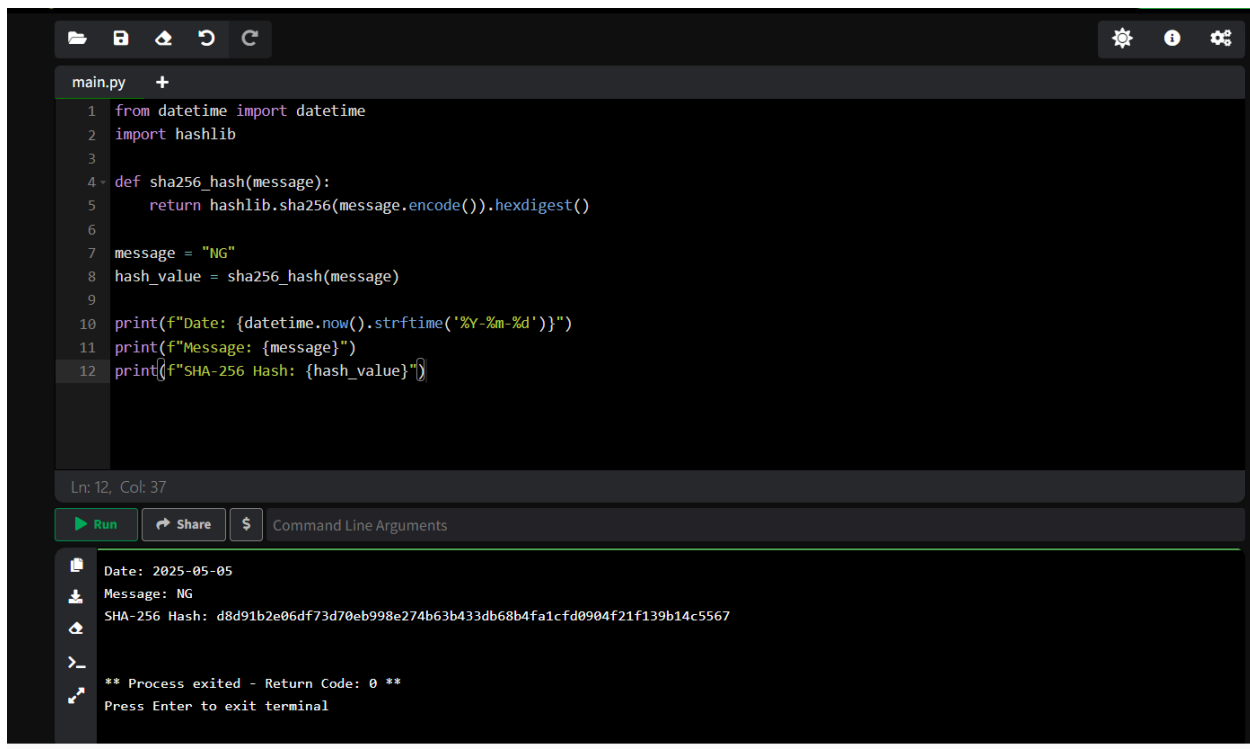
(7 → N, 4 → E) Băm sẽ khác nhưng  
thực ra không đổi

VD băm 74: SHA-256("74")

18d91b2e06d873d70eb998e270

**HUTECH**  
Đại học Công nghệ TP.HCM

CODE:



The screenshot shows a code editor with a dark theme. The top bar contains icons for file operations and settings. The main area displays a Python script in a file named 'main.py'. The script defines a function 'sha256\_hash' and uses it to calculate the SHA-256 hash of the message 'NG'. It also prints the current date and the message. The bottom bar shows the 'Run' button and the output of the script, which includes the date, message, and SHA-256 hash.

```
main.py +
1 from datetime import datetime
2 import hashlib
3
4 def sha256_hash(message):
5     return hashlib.sha256(message.encode()).hexdigest()
6
7 message = "NG"
8 hash_value = sha256_hash(message)
9
10 print(f>Date: {datetime.now().strftime('%Y-%m-%d')}")
11 print(f>Message: {message}")
12 print(f>SHA-256 Hash: {hash_value}")
```

Ln: 12, Col: 37

Run Share \$ Command Line Arguments

Date: 2025-05-05  
Message: NG  
SHA-256 Hash: d8d91b2e06df73d70eb998e274b63b433db68b4fa1cfd0904f21f139b14c5567

\*\* Process exited - Return Code: 0 \*\*  
Press Enter to exit terminal

```
from datetime import datetime
```

```
import hashlib
```

```
def sha256_hash(message):  
    return hashlib.sha256(message.encode()).hexdigest()
```

```
message = "NG"
```

```
hash_value = sha256_hash(message)
```

```
print(f>Date: {datetime.now().strftime('%Y-%m-%d')}")
```

```
print(f>Message: {message}")
```

```
print(f>SHA-256 Hash: {hash_value}")
```