**MSSV:** 23520315

**Họ và tên:** Tào Minh Đức

**Link code của các bài dưới:** https://github.com/Duck4705/Crypto.git

# BÁO CÁO BÀI TẬP TUẦN 3

## **Bài 1:** Code your own chaotic-based stream cipher

-Thuật toán sử dụng Henon map

+Giới thiệu: Henon Map là một hệ động lực hỗn loạn được giới thiệu bởi Michel Hesnon vào năm 1976. Nó là một hệ hai chiều phi tuyến, thường được sử dụng để nghiên cứu sự hỗ loạn trong hệ thống động lực. Trong bài mã hóa stream cipher, thì người ta ứng dụng thuật toán này để tại ra sự hỗn loạn tốt hơn logistic Map mà người ta không thể tìm ra quy luật của khóa

+Lý do chọn Henon map: Vì henon map có hai biến là x và y thay vì một biến như logistic, sẽ làm tăng độ hỗn loạn và khó đoán, Hệ henon map là không tuần hoàn ngắn như logistic, dễ dàng triển khai như logistic và nhanh, bảo mật hơn logistic.

+Công thức của thuật toán:

$$x_{n+1} = 1 - ax_n^2 + y_n$$

$$y_{n+1} = bx_n$$

+Test độ nhạy tạo khóa

```
D:\Project_Pycharm\Mat_ma_hoc\.venv\Scripts\
Enter x0: 0
Enter y0: 0
Enter a: 0.1
Enter b: 0.1
Enter keystream length: 10
x[0] = 1.0000000000000000
x[1] = 0.9000000000000000
x[2] = 1.0190000000000001
x[3] = 0.9861638999999999
x[4] = 1.0046480762336791
x[5] = 0.9976846142919967
x[6] = 1.0009273486638708
x[7] = 0.9995829056988711
x[8] = 1.0001761363298474
x[9] = 0.9999230602015170
```

```
Enter x0: 0
Enter y0: 0
Enter a: 1
Enter b: 0.5
Enter keystream length: 10
x[0] = 1.0000000000000000
x[1] = 0.0000000000000000
x[2] = 1.5000000000000000
x[3] = -1.2500000000000000
x[4] = 0.1875000000000000
x[5] = 0.3398437500000000
x[6] = 0.9782562255859375
x[7] = 0.2129366321023554
x[8] = 1.4437861035018749
x[9] = -0.9780499966139486
```

```
D:\Project_Pycharm\Mat_ma_hoc\.venv\Scripts\pyt
Enter x0: 0
Enter y0: 0
Enter a: 0.5
Enter b: 0.4
Enter keystream length: 10
x[0] = 1.0000000000000000
x[1] = 0.5000000000000000
x[2] = 1.2749999999999999
x[3] = 0.3871875000000001
x[4] = 1.4350429199218748
x[5] = 0.1252009089910499
x[6] = 1.5661795341626572
x[7] = -0.1763788030185589
x[8] = 1.6109170725879332
x[9] = -0.3680784285850617
```

```
Enter x0: 0
Enter y0: 0
Enter a: 1.23
Enter b: 0.23
Enter keystream length: 10
x[0] = 1.0000000000000000
x[1] = -0.2300000000000000
x[2] = 1.1649330000000000
x[3] = -0.7220947402214699
x[4] = 0.6265869889577202
x[5] = 0.3510063664298062
x[6] = 0.9925722802529415
x[7] = -0.1310642054987688
x[8] = 1.2071628985236578
x[9] = -0.8225527514583244
```

Khi tạo xong một số seed thì nhận thấy rằng để tạo sự hỗn loạn tốt nhất thì điều kiện

1.2 < a < 1.4 và 0.2 < b < 0.3 lúc này khóa được tạo ra sẽ không phân kỳ và không có chu kỳ ngắn

-Phần code:

```python
def henon_map_keygen(length: int, x0: float, y0: float, a: float = 1.4, b: float = 0.3) -> bytes:
    """
    Generate a keystream of 'length' bytes using the Henon map.

    :param length: Number of bytes in the keystream
    :param x0: Initial x value (-1 < x0 < 1)
    :param y0: Initial y value (-1 < y0 < 1)
    :param a: Henon map parameter (default 1.4 for chaotic behavior)
    :param b: Henon map parameter (default 0.3 for chaotic behavior)
    :return: Keystream as bytes
    """
    x, y = x0, y0
    key = bytearray(length)

    for i in range(length):
        x, y = 1 - a * x ** 2 + y, b * x  # Henon map iteration
        key[i] = int((x % 1) * 256) & 0xFF  # Normalize x to [0, 255]

    return bytes(key)
```

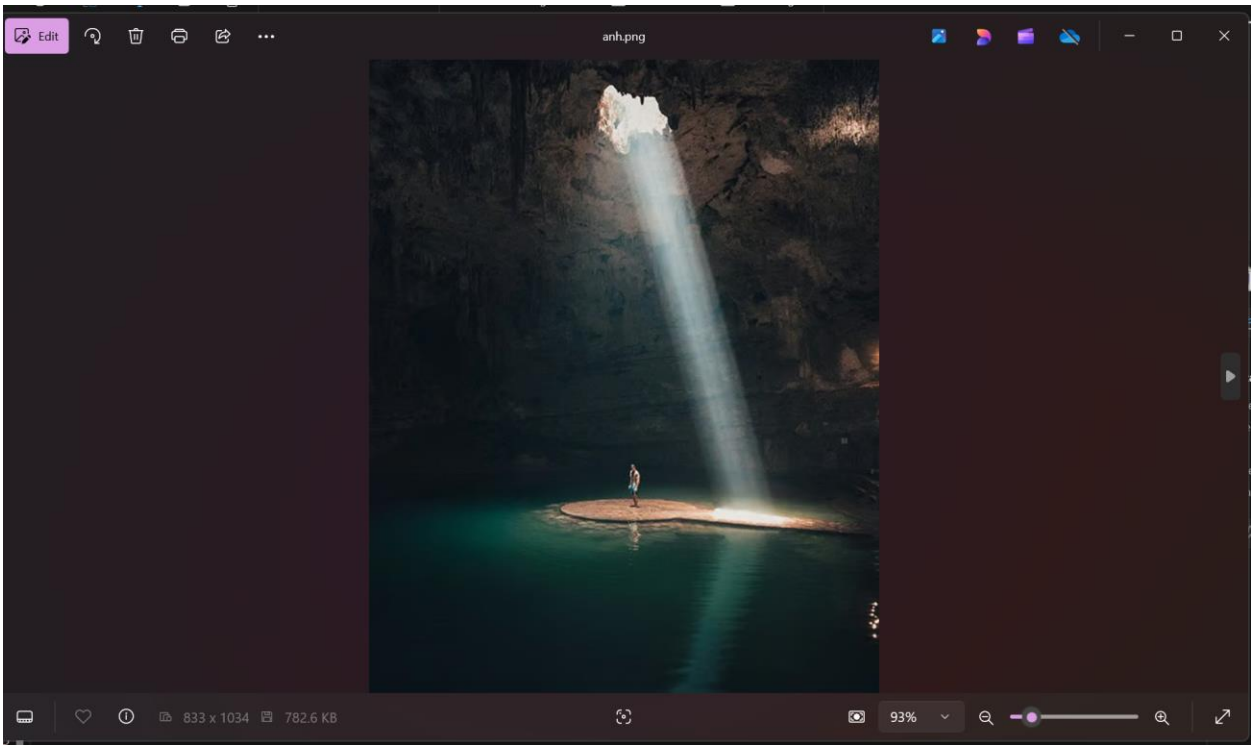
```python
1 usage
def chaotic_encrypt_decrypt(input_path: str, output_path: str, x0: float, y0: float, a: float, b: float) -> bytes:
    """
    Encrypt or decrypt a file (any binary) using a Logistic map-based keystream.
    XOR is used for both encryption and decryption.

    :param input_path: path to the input file
    :param output_path: path to the output file
    :param x0: initial seed for the Logistic map
    :param r: parameter for the Logistic map
    """
    # Read entire file in binary mode
    with open(input_path, 'rb') as f_in:
        data = f_in.read()

    length = len(data)
    # Generate keystream of the same length
    key = henon_map_keygen(length, x0, y0, a, b)

    # XOR each byte
    result = bytes(d ^ k for d, k in zip(data, key))

    # Write result to output
    with open(output_path, 'wb') as f_out:
        f_out.write(result)
```

```python
def main():
    print("=== Chaotic Map (Henon Map) Stream Cipher Demo ===")
    print("This script uses a simple XOR-based scheme with a Henon map keystream.")
    print("Disclaimer: Not secure for real-world cryptography.\n")

    mode = input("Enter mode (encrypt/decrypt): ").strip().lower()
    if mode not in ("encrypt", "decrypt"):
        print("Invalid mode. Use 'encrypt' or 'decrypt'.")
        return

    input_file = input("Enter path to input file: ").strip()
    output_file = input("Enter path to output file: ").strip()

    # Get seed x0 (-1 < x0 < 1), y (-1 < y0 < 1) and parameter a (commonly near 1.4), b (commonly near 0.3
    try:
        x0 = float(input("Enter henon map seed x0 (-1 < x0 < 1): ").strip())
        y0 = float(input("Enter henon map seed y0 (-1 < x0 < 1): ").strip())
        a = float(input("Enter henon map parameter r (e.g., 1.4): ").strip())
        b = float(input("Enter heno map parameter r (e.g., 0.3): ").strip())
    except ValueError:
        print("Invalid x0 or y0 or a or b. Must be float.")
        return

    if not (0 < x0 < 1) and not (0 < y0 < 1):
        print("x0,y0 must be between -1 and 1.")

        return

    # Perform the XOR-based operation
    # (Encryption and decryption are identical in this scheme.)
    chaotic_encrypt_decrypt(input_file, output_file, x0, y0, a, b)

    print(f"\nDone. {'Encrypted' if mode=='encrypt' else 'Decrypted'} file saved to '{output_file}'.")
```

-Test code bằng các mã hóa/giải mã file ảnh(png), file video(mp4), file nhạc(mp3)

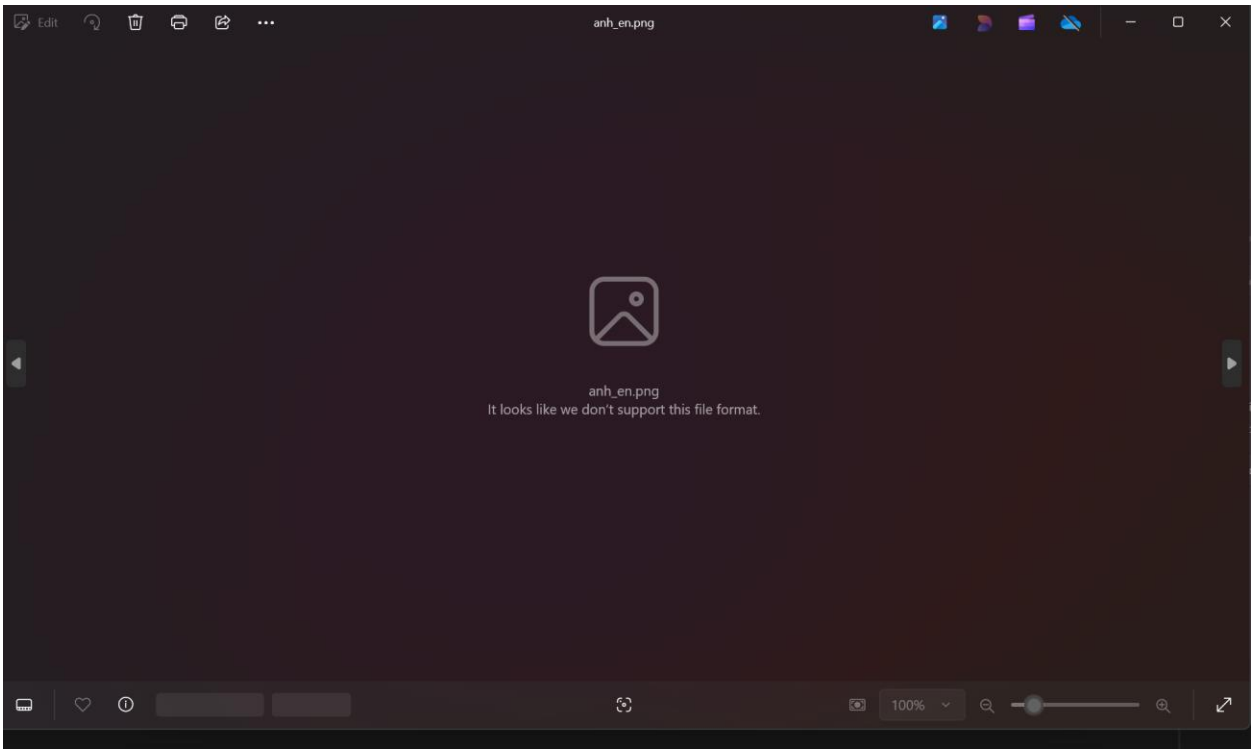
+file ảnh(png)

Ảnh lúc chưa mã hóa



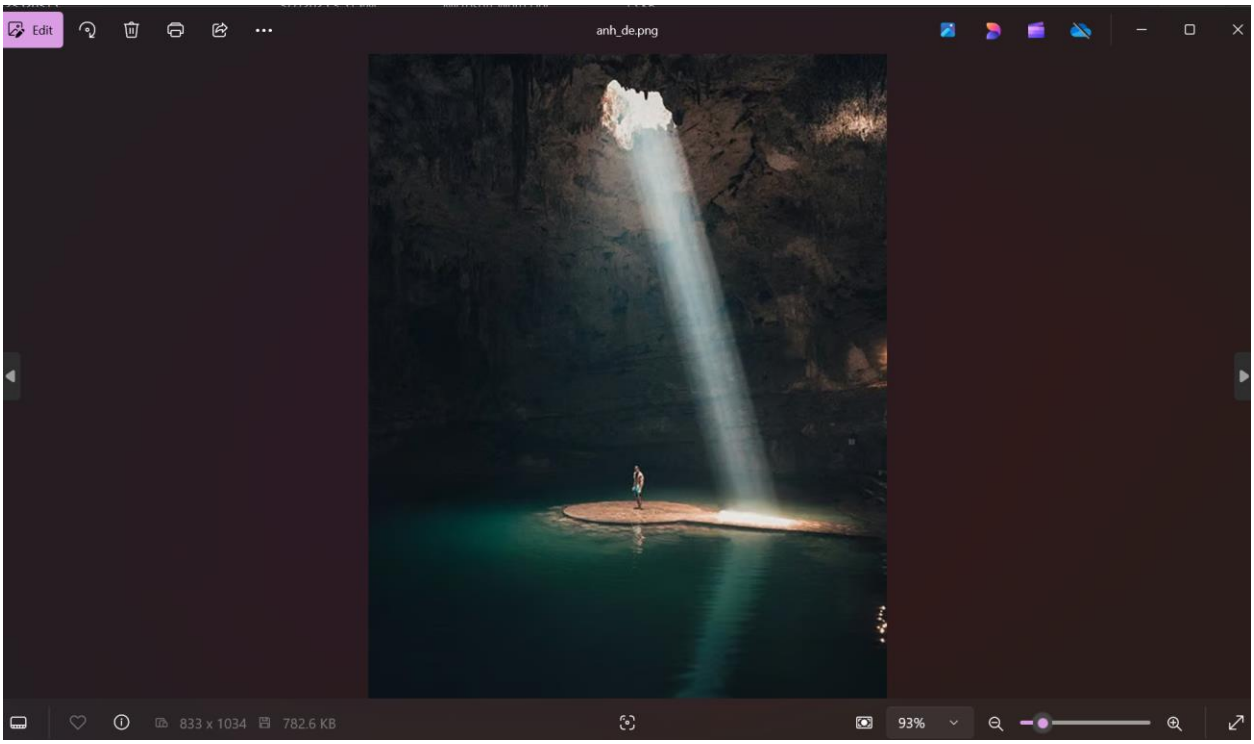

Sau khi mã hóa

```
=== Chaotic Map (Henon Map) Stream Cipher Demo ===
This script uses a simple XOR-based scheme with a Henon map keystream.
Disclaimer: Not secure for real-world cryptography.

Enter mode (encrypt/decrypt): encrypt
Enter path to input file: anh.png
Enter path to output file: anh_en.png
Enter henon map seed x0 (-1 < x0 < 1): 0.5
Enter henon map seed y0 (-1 < x0 < 1): 0.5
Enter henon map parameter r (e.g., 1.4): 1.4
Enter heno map parameter r (e.g., 0.3): 0.3

Done. Encrypted file saved to 'anh_en.png'.
```



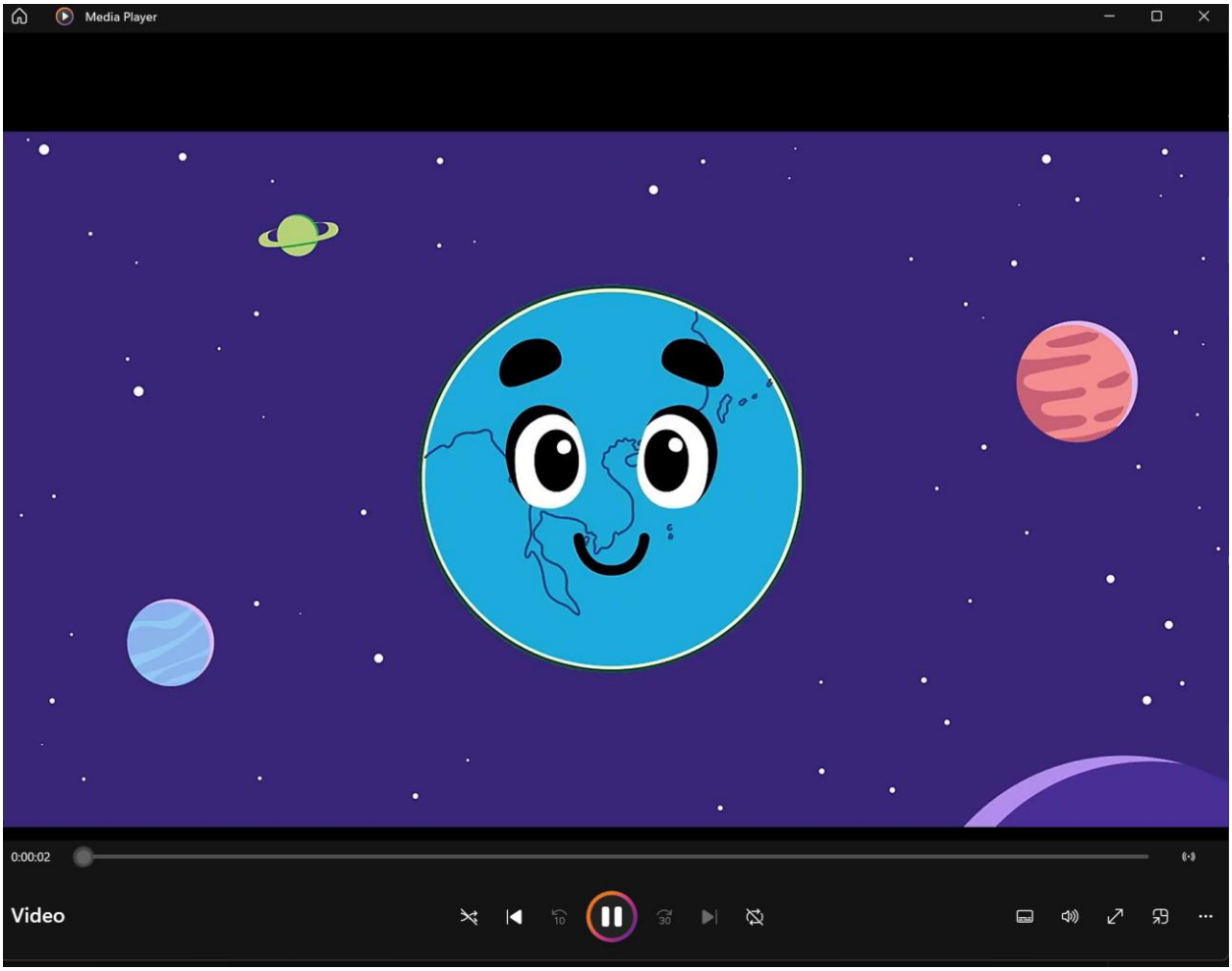

Giải mã ảnh

```
=== Chaotic Map (Henon Map) Stream Cipher Demo ===
This script uses a simple XOR-based scheme with a Henon map keystream.
Disclaimer: Not secure for real-world cryptography.

Enter mode (encrypt/decrypt): decrypt
Enter path to input file: anh_en.png
Enter path to output file: anh_de.png
Enter henon map seed x0 (-1 < x0 < 1): 0.5
Enter henon map seed y0 (-1 < x0 < 1): 0.5
Enter henon map parameter r (e.g., 1.4): 1.4
Enter heno map parameter r (e.g., 0.3): 0.3

Done. Decrypted file saved to 'anh_de.png'.
```



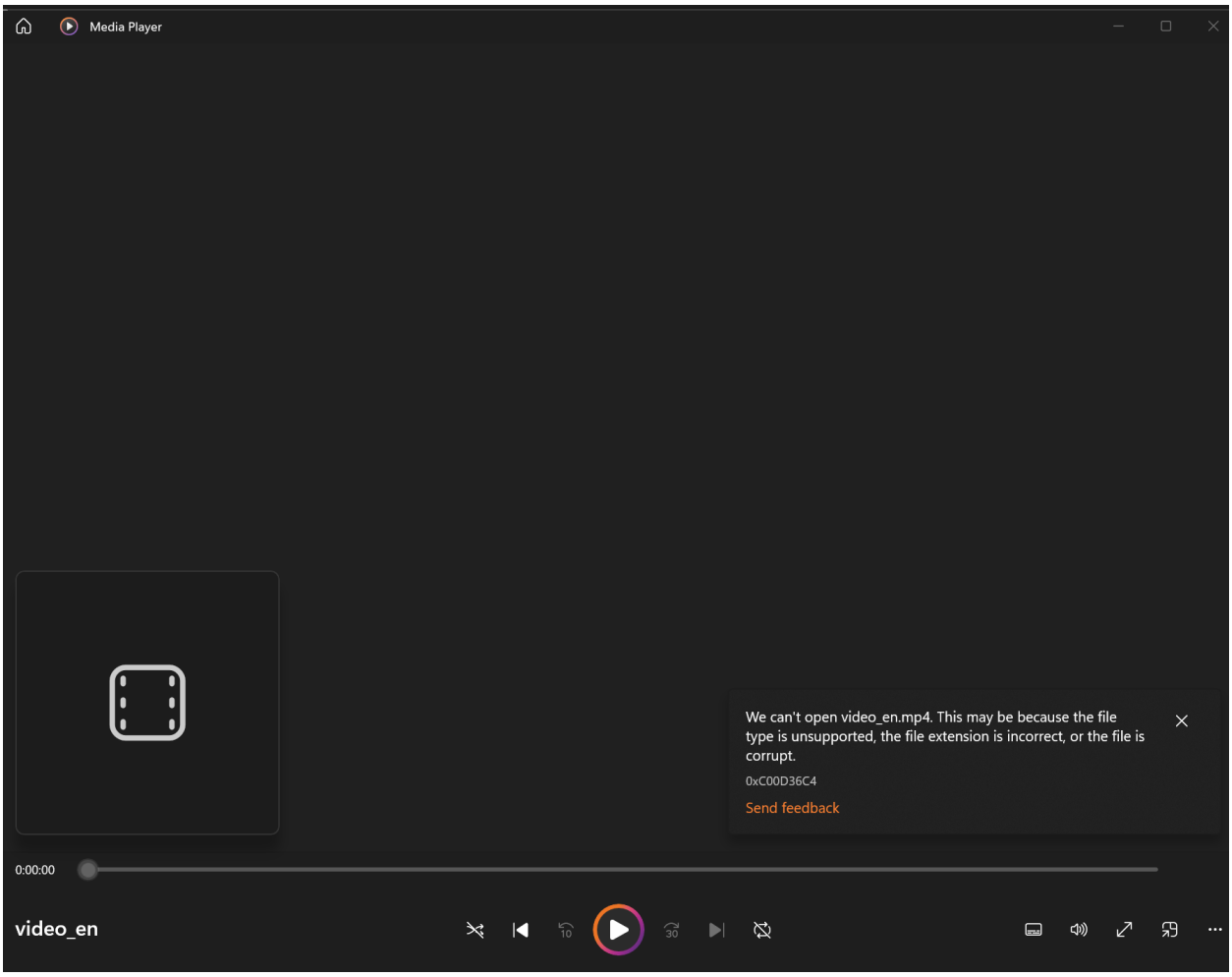

+file video(mp4)

Video trước khi chưa mã hóa



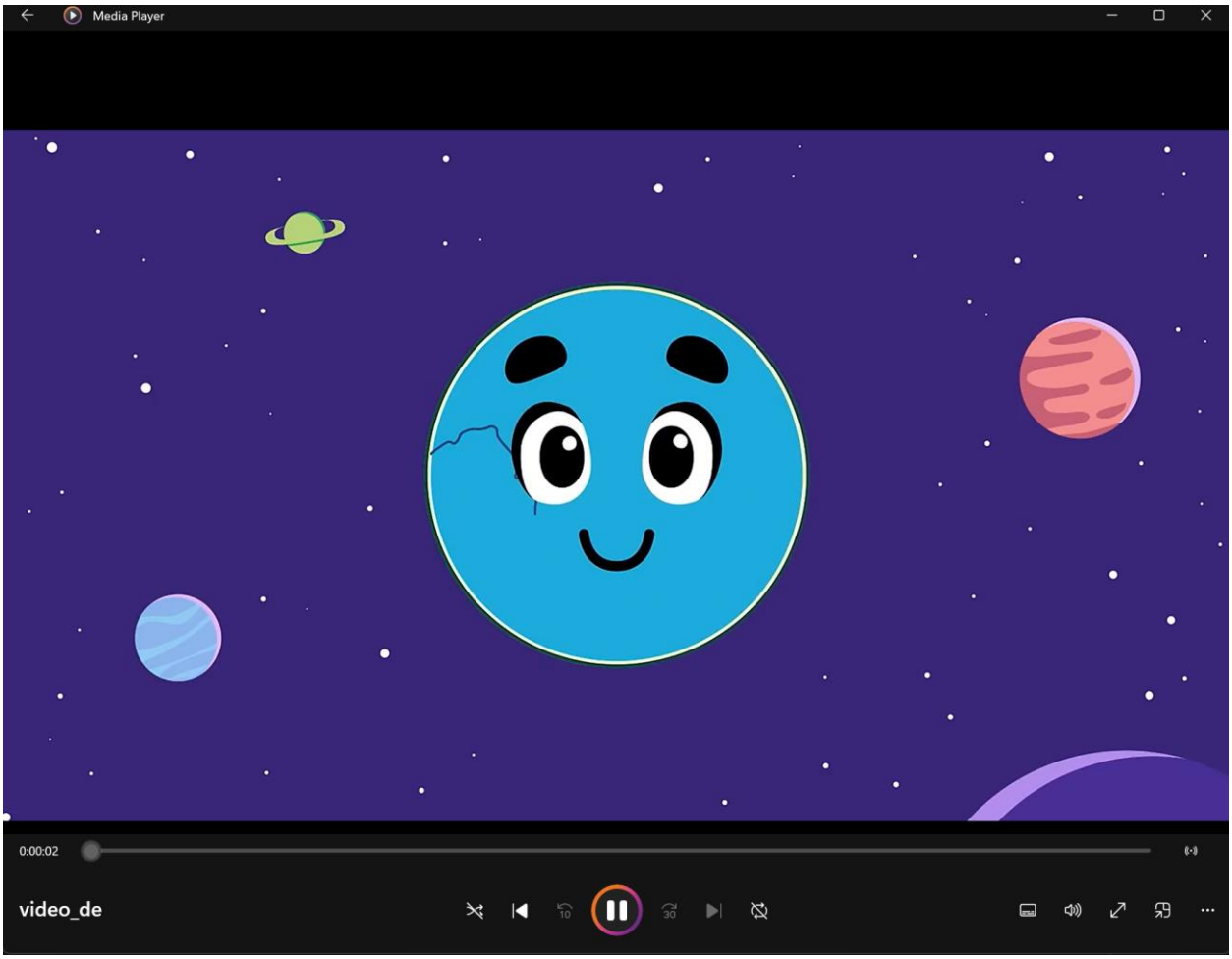

Video sau khi mã hóa

```
D:\Project_Pycharm\Mat_ma_hoc\.venv\Scripts\python.exe D:\Project_Pycharm\Mat_ma_hoc\Task3\own_Chaoticstreamcipher.py
=== Chaotic Map (Henon Map) Stream Cipher Demo ===
This script uses a simple XOR-based scheme with a Henon map keystream.
Disclaimer: Not secure for real-world cryptography.

Enter mode (encrypt/decrypt): encrypt
Enter path to input file: video.mp4
Enter path to output file: video_en.mp4
Enter henon map seed x0 (-1 < x0 < 1): -0.5
Enter henon map seed y0 (-1 < x0 < 1): -0.5
Enter henon map parameter r (e.g., 1.4): 1.398
Enter heno map parameter r (e.g., 0.3): 0.298

Done. Encrypted file saved to 'video_en.mp4'.
```



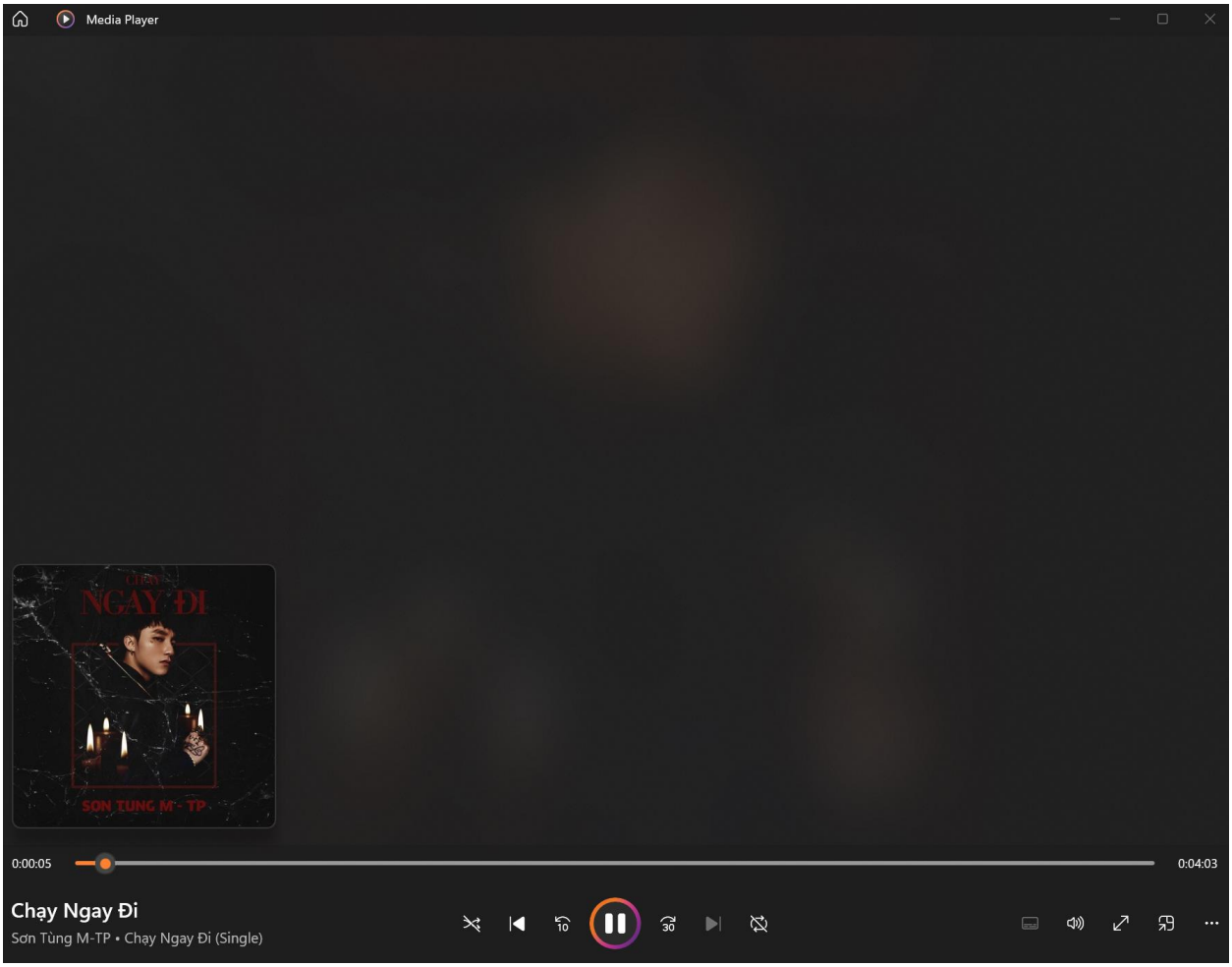

Video sau khi giải mã

```
D:\Project_Pycharm\Mat_ma_hoc\.venv\Scripts\python.exe D:\Project_Pycharm\Mat_ma_hoc\Task3\own_Chaoticstreamcipher.py
=== Chaotic Map (Henon Map) Stream Cipher Demo ===
This script uses a simple XOR-based scheme with a Henon map keystream.
Disclaimer: Not secure for real-world cryptography.

Enter mode (encrypt/decrypt): decrypt
Enter path to input file: video_en.mp4
Enter path to output file: video_de.mp4
Enter henon map seed x0 (-1 < x0 < 1): -0.5
Enter henon map seed y0 (-1 < x0 < 1): -0.5
Enter henon map parameter r (e.g., 1.4): 1.398
Enter heno map parameter r (e.g., 0.3): 0.298

Done. Decrypted file saved to 'video_de.mp4'.
```



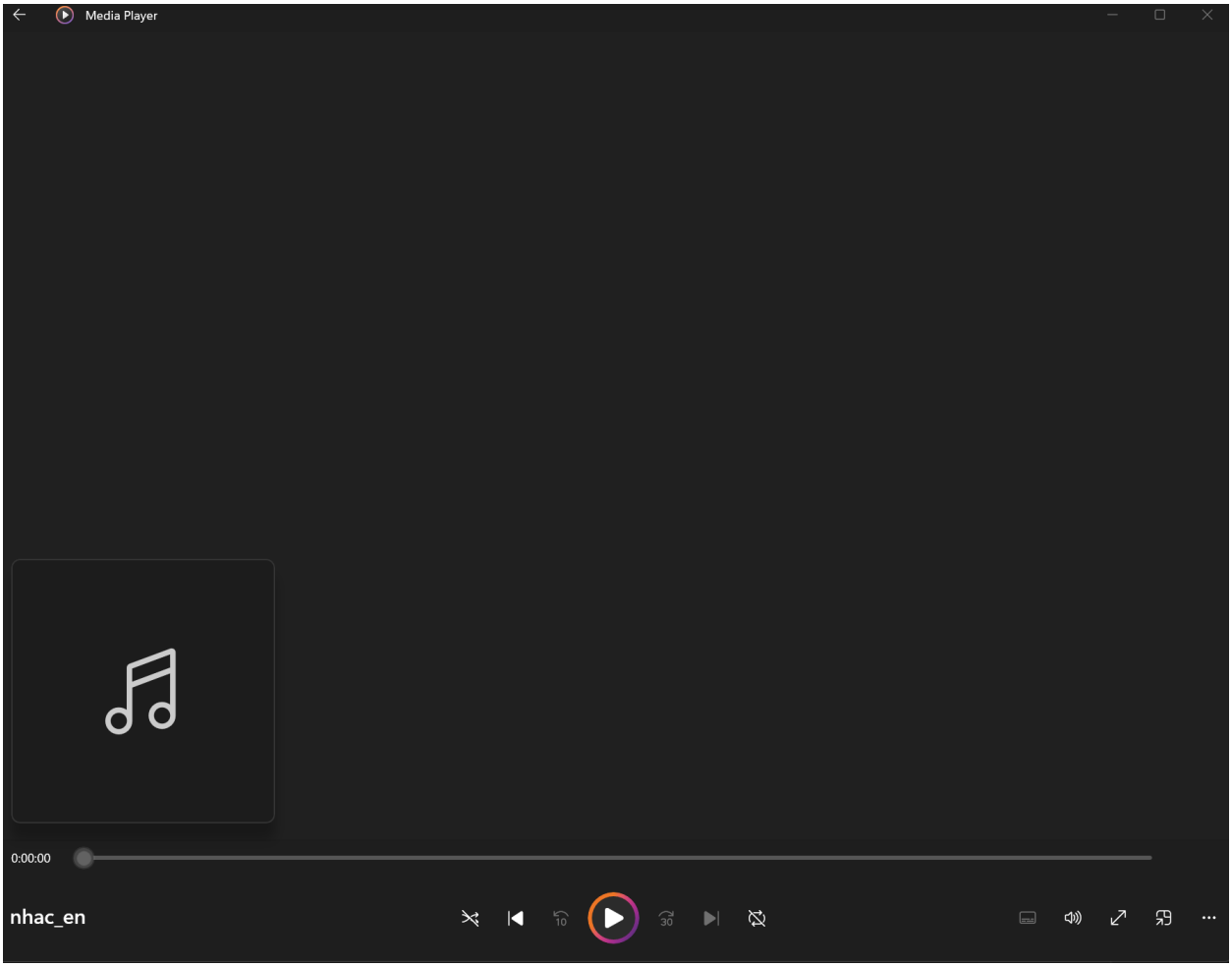

+file nhạc(mp3)

Nhạc trước khi mã hóa



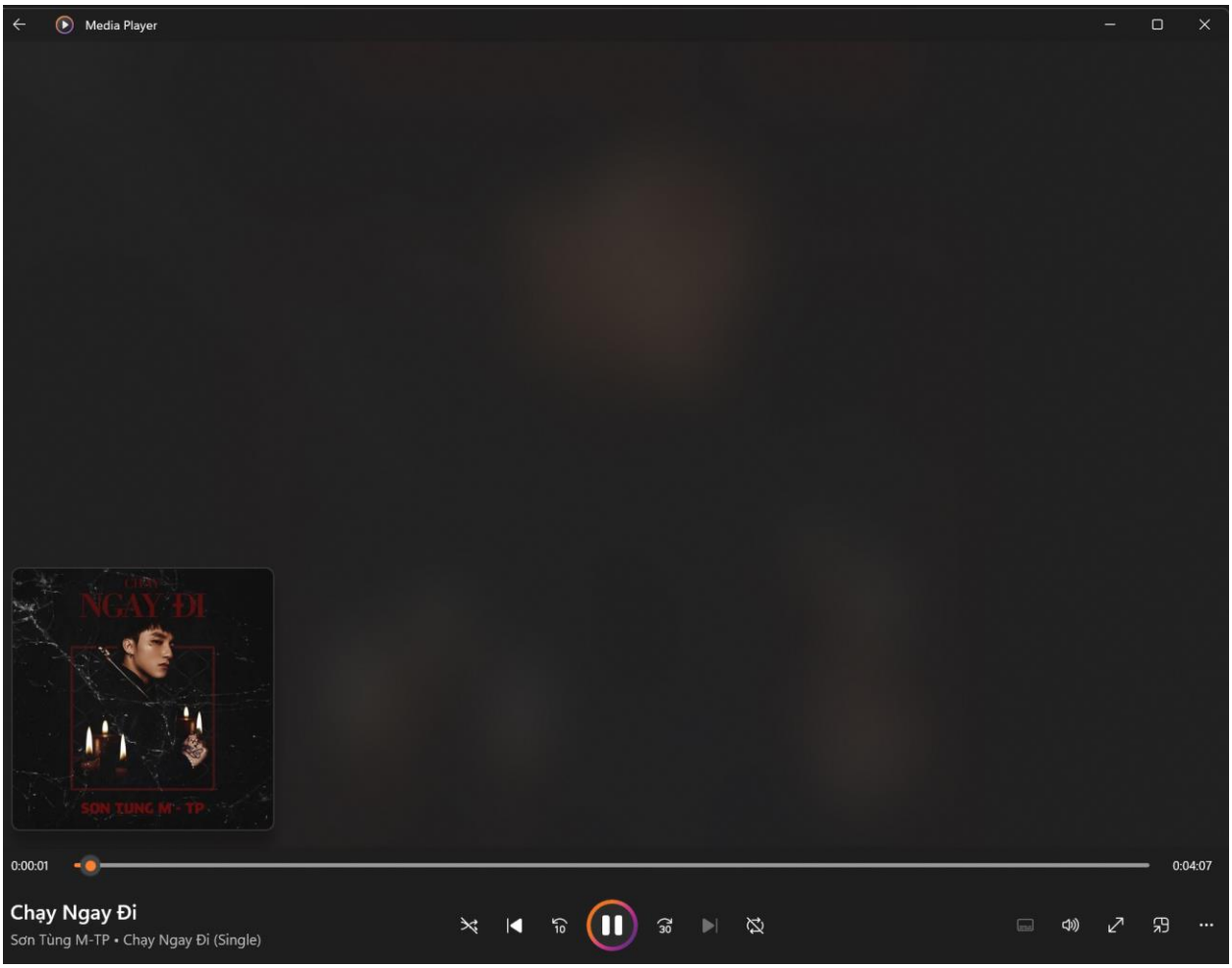

Nhạc sau khi mã hóa

```
D:\Project_Pycharm\Mat_ma_hoc\.venv\Scripts\python.exe D:\Project_Pycharm\Mat_ma_hoc\Task
=== Chaotic Map (Henon Map) Stream Cipher Demo ===
This script uses a simple XOR-based scheme with a Henon map keystream.
Disclaimer: Not secure for real-world cryptography.

Enter mode (encrypt/decrypt): encrypt
Enter path to input file: nhac.mp3
Enter path to output file: nhac_en.mp3
Enter henon map seed x0 (-1 < x0 < 1): 0
Enter henon map seed y0 (-1 < x0 < 1): 0
Enter henon map parameter r (e.g., 1.4): 1.4
Enter heno map parameter r (e.g., 0.3): 0.3

Done. Encrypted file saved to 'nhac_en.mp3'.
```




Nhạc sau khi giải mã

```
D:\Project_Pycharm\Mat_ma_hoc\.venv\Scripts\python.exe D:\Project_Pycharm\Mat_ma_hoc\Task3
=== Chaotic Map (Henon Map) Stream Cipher Demo ===
This script uses a simple XOR-based scheme with a Henon map keystream.
Disclaimer: Not secure for real-world cryptography.

Enter mode (encrypt/decrypt): decrypt
Enter path to input file: nhac_en.mp3
Enter path to output file: nhac_de.mp3
Enter henon map seed x0 (-1 < x0 < 1): 0
Enter henon map seed y0 (-1 < x0 < 1): 0
Enter henon map parameter r (e.g., 1.4): 1.4
Enter heno map parameter r (e.g., 0.3): 0.3

Done. Decrypted file saved to 'nhac_de.mp3'.

Process finished with exit code 0
```




## **Bài 2:** Cryptanalysis  Stream Cipher

Trả lời câu hỏi: Does your  Stream Cipher is secure of one use the same key to encrypt many files?

-Stream cipher hay thuật toán Henon Map nói riêng đều không an toàn khi sử dụng chung một khóa cho mã hóa nhiều file. Dù nó có tại ra độ hỗn loạn mạnh tới đâu hay là xử lý vấn đề tuần hoàn ngắn thì chúng ta cũng không nên sử dụng một khóa cho nhiều file để mã hóa.

+**Thứ nhất:** khi hacker biết được hai file đó sử dụng cùng một key. Thì chúng có thể xor hai cipher text lại và chúng cần biết thêm một trong hai plaintext thì hoàn toán có thể suy ra

một trong hai plaintext còn lại  bởi vì thuật toán mã hóa chung của stream cipher là phép xor

Ví dụ: C1= P1 ⊕ K,  C2 = P2 ⊕ K  // Đây là hai ciphertext đã được mã hóa chung một khóa K

Hacker tiến hành xor hai Ciphertext lại

C1⊕C2=(P1 ⊕ K) ⊕ (P2 ⊕ K) = P1 ⊕ P2 // Vì phép xor có tính giao hoán nên khi K ⊕ K sẽ bằng 0 mà A ⊕ 0 = A
Qua công thức trên hacker chỉ cần biết P1 hoặc P2 là hoàn toàn có thể biết plaintext còn lại

+**Thứ hai:** hacker có thể suy ra khóa K nếu biết trước plaintext và ciphertext và áp dụng K đó để mã hóa cho tất cả các file còn lại
Ví dụ: C = P ⊕ K

Suy ra K = C ⊕ P

-Kết luận không nên xài một khóa để mã hóa cho các thuật toán ciphertext nói chung và heno map nói riêng. Chúng ta phải kết hợp nhiều khóa để mã hóa cho nhiều file