

## Lab01 - IS355.Q11

MSSV: 22520335

Họ và tên: Nguyễn Trung Duy

### Kết quả bài tập tại lớp

#### B. Chữ ký số lên dữ liệu trong blockchain

1. Bạn thử đóng vai trò của Tèo viết một đoạn chương trình Python để cố gắng sửa thông điệp của Tý thành “Ty hates cat”. Bạn chụp hình lại kết quả chạy.

Code:

```
tysignature.py  teo_attack.py x message.sig  ty.pub
teo_attack.py > ...
1  from cryptography.hazmat.primitives import hashes
2  from cryptography.hazmat.primitives.asymmetric import padding
3  from cryptography.hazmat.primitives import serialization
4
5  # --- Load public key (công khai của Tý) ---
6  with open("ty.pub", "rb") as f:
7      public_key = serialization.load_pem_public_key(f.read())
8
9  # --- Giả mạo thông điệp ---
10 tampered_message = b"Ty hates cat"
11
12 # --- Load chữ ký thật của Tý (đã ký trên "Ty likes cat") ---
13 with open("message.sig", "rb") as f:
14     signature = f.read()
15
16 # --- Thử xác minh chữ ký ---
17 print("🔒 Tèo đang cố xác minh thông điệp đã bị sửa...")
18 try:
19     public_key.verify(
20         signature,
21         tampered_message,
```

Kết quả:

```
• (.venv) PS D:\BlockChain\Thuchanh1> python teo_attack.py
🔒 Tèo đang cố xác minh thông điệp đã bị sửa...
❌ Xác minh thất bại!
```

2. Bạn chạy TySignature.py, chụp lại kết quả dữ liệu được mã hoá dựa trên thông điệp “Ty likes cat”. Sau đó bạn thay thông điệp thành “Ty hates cat” rồi chạy lại chương trình đồng thời chụp kết quả và so sánh 2 giá trị này.

- Kết quả chạy thông điệp “Ty likes cat”

```
(.venv) PS D:\BlockChain\Thuchanh1> python TySignature.py
--- Generating new RSA key pair ---
✅ Private key saved to ty.pem
✅ Public key saved to ty.pub

🔥 Signing message: 'Ty likes cat'
🔑 Signature (hex):
489ba742bf78f74376df1b6e5b78b98feecde3c0b06d03e50bd82de8693a53f58c1d02fa82c8a73db37ff8e502d08621da3bc996d0862d65d153f3caa7a0671f95d983fdb16e8893d687037
50b0b4f5b4f895ebd865be088c1e8805a1dca8710a3660490125b252e381a4b0a1c5d8e92acea954b80514cd8f12652ff80a7aa62a3700bbbf2ac8010750403e44bcabac413d699ec6d0593b7
1d92c02b50d583a1757be52d75ef4d1495da7b08ce6d49467cb2555e3997dd52700a64f3f4562b0347f55d0ac57f618fb39e50d9f0f3c11b102a9319940ef929f696008a10b41abd803214cf
d88e59d280bebf5e8721759e6b2006f68332b1c8a5f00822f695
📁 Signature saved to message.sig
```

- Kết quả chạy thông điệp “Ty hates cat”

```
(.venv) PS D:\BlockChain\Thuchanh1> python TySignature.py
🔑 Keys loaded successfully.

🔥 Signing message: 'Ty hates cat'
🔑 Signature (hex):
147bfeb59976094a9878b717351aa05d4490a9c8e0cd9693a21c99d681165774db642ee5d7b73f6e70f8fad6097aef617708a59f79d8c92ff58255b8ca4cf9ae0f12dc4f0d83318d6d75c9c
70138d3c4af131fe7c19cba5cb3d8bf2f28d7372207aacaac6a44befd569e507d20b7d02c94142205416e28dfe27e5edc0c7432c03db7b7ef9e6f9ad77e7d4b07723e00fdc50795891730818
c968271dd0615c443d8c64a244eb191c7255dd7b0b461f859d0381cb40e0f4ec56fd40bf636c516255622ef52baa52b975e0372ad54dfb5f940c9ac3924becbe459a33e4acd2f6dde78d35db9
b682b34d3e8ff4cbe59c7f2e2c4316079ae5874f862e7d4b0b8df
📁 Signature saved to message.sig
```

⇒ Hai giá trị chữ ký số hoàn toàn khác nhau

- Nhận xét: hai giá trị chữ ký số thu được hoàn toàn khác nhau mặc dù chỉ thay đổi một từ trong thông điệp. Điều này chứng tỏ chữ ký số **phụ thuộc chặt chẽ vào nội dung thông điệp**. Chỉ cần một thay đổi nhỏ trong thông điệp cũng dẫn đến sự thay đổi hoàn toàn của chữ ký. Do đó, nếu có ai đó (như Tèo) cố gắng sửa đổi nội dung thông điệp, chữ ký số sẽ **không còn hợp lệ**, và thông điệp sẽ bị phát hiện là **đã bị thay đổi**.

3. \* Bạn viết 1 chương trình để xác thực xem có phải Tý là người đã viết “Ty likes cat” hay không?

- Code

```

verify_signature.py > ...
1  from cryptography.hazmat.primitives import hashes
2  from cryptography.hazmat.primitives.asymmetric import padding
3  from cryptography.hazmat.primitives import serialization
4
5  MESSAGE = b"Ty likes cat"
6
7  # --- Load public key của Tý ---
8  with open("ty.pub", "rb") as f:
9      public_key = serialization.load_pem_public_key(f.read())
10
11  # --- Load chữ ký ---
12  with open("message.sig", "rb") as f:
13      signature = f.read()
14
15  # --- Kiểm tra ---
16  print("🔍 Đang xác minh xem có phải Tý đã ký thông điệp không...")
17  try:
18      public_key.verify(
19          signature,
20          MESSAGE,
21          padding.PSS(
22              mgf=padding.MGF1(hashes.SHA256()),
23              salt_length=padding.PSS.MAX_LENGTH
24          ),
25          hashes.SHA256()
26      )
27      print("✅ Đúng! Thông điệp 'Ty likes cat' được ký bởi Tý và không bị thay đổi.")
28  except Exception:
29      print("❌ Sai! Đây KHÔNG phải là chữ ký hợp lệ từ Tý.")
30

```

- Kết quả

```

(.venv) PS D:\Blockchain\Thuchanh1> python verify_signature.py
🔍 Đang xác minh xem có phải Tý đã ký thông điệp không...
✅ Đúng! Thông điệp 'Ty likes cat' được ký bởi Tý và không bị thay đổi.

```

## E. Ngôn ngữ lập trình Vyper

### i. Kiểu dữ liệu

Viết một hợp đồng đặt tên DataType.vy để kiểm thử các kiểu dữ liệu như hình 13.

Sau đó bạn biên dịch và chụp kết quả để nộp báo cáo.\

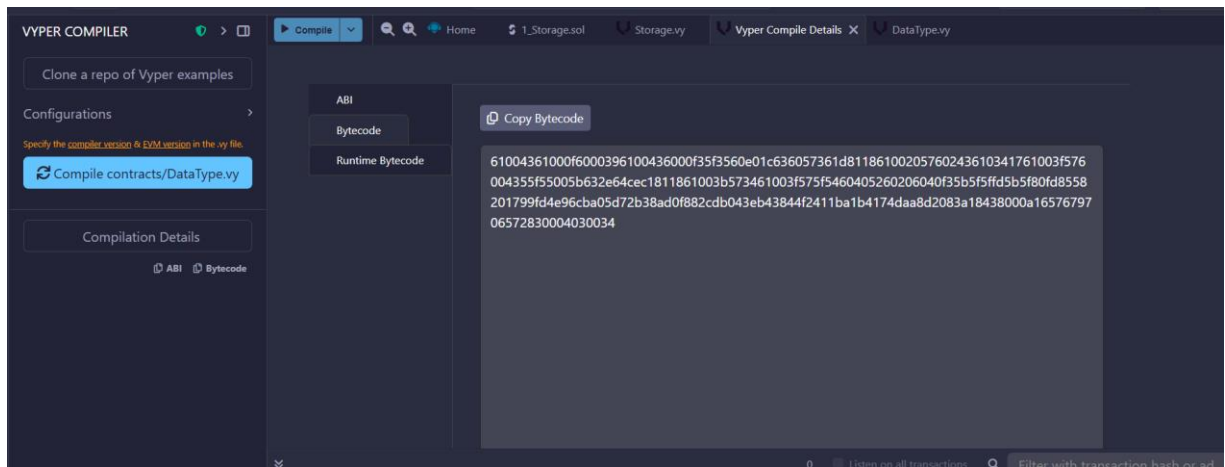
- Code:

```

39  @deploy
40  def __init__():
41      self.life_is_beautiful = True
42      self.var_int1 = -8
43      self.var_int2 = -64
44      self.var_int3 = -128
45      self.var_int4 = -256
46
47      self.var_uint1 = 8
48      self.var_uint2 = 64
49      self.var_uint3 = 128
50      self.var_uint4 = 256
51
52      self.my_grandma_wallet = msg.sender
53
54      self.var_byte1 = convert(b"Hello Vyper!", bytes32)
55      self.var_byte2 = convert(b"Hello Vyper!", bytes18)
56      self.var_bytes = b"Test Bytes data"
57      self.author = "Nguyen Trung Duy"
58
59      self.direction = Direction.NORTH
60

```

- Kết quả biên dịch



### iii. Mutability

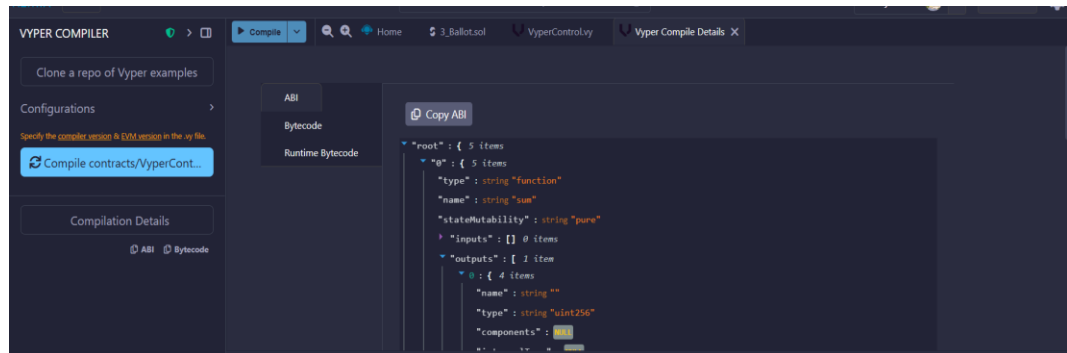
- Kết quả biên dịch:



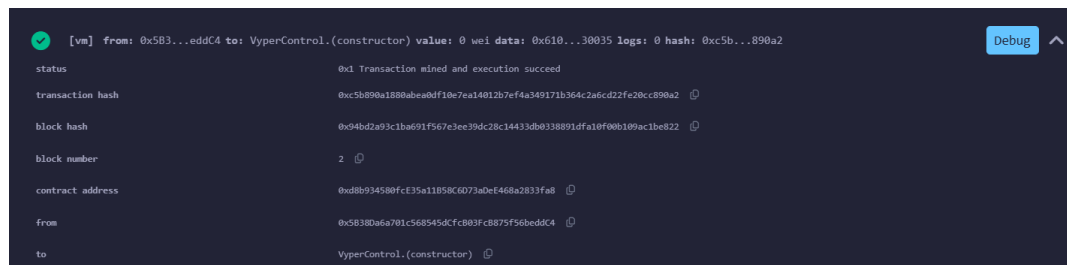
#### iv. Cấu trúc điều khiển

Yêu cầu: Biên dịch, sau đó triển khai đoạn hợp đồng thông minh trên nền tảng Ethereum để xem kết quả như thế nào, và ghi nhận vào báo cáo?

- Kết quả biên dịch :

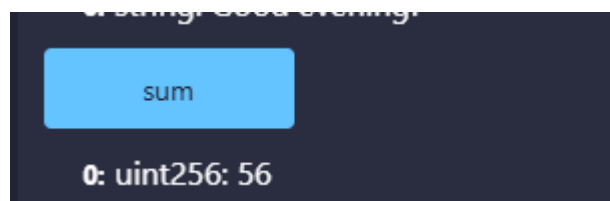


- Kết quả triển khai



- Sau khi triển khai lên Ethereum, các hàm hoạt động đúng như logic điều khiển được lập trình:

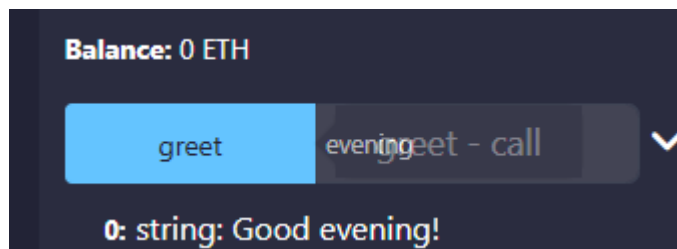
- Hàm sum(): trả về 56.



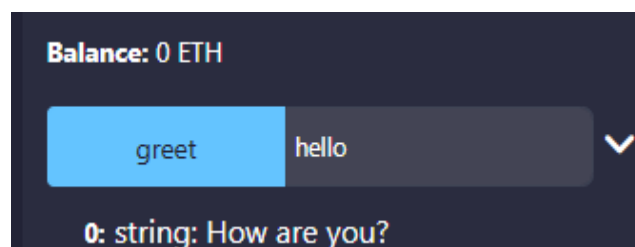
- Hàm greet(time):
  - Kết quả: Trả về lời chào phù hợp với cấu trúc if/elif/else.
  - Hoạt động:
    - Nếu nhập "morning", trả về "Good morning!".



- Nếu nhập "evening", trả về "Good evening!".



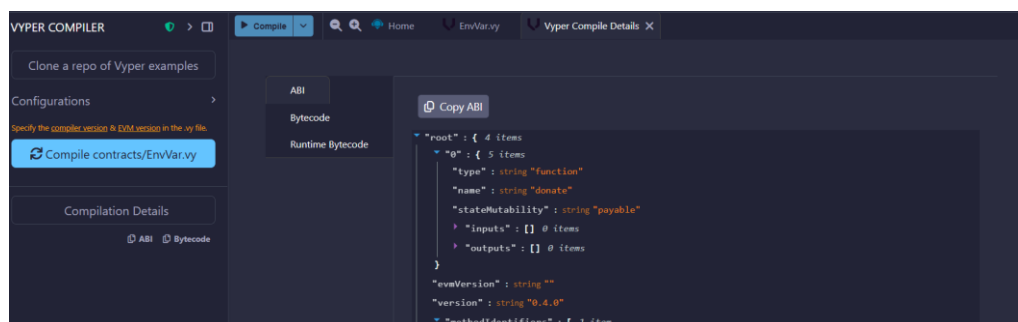
- Nếu nhập chuỗi khác, trả về "How are you?". Ví dụ: nhập "hello":



## v. Biến môi trường

Yêu cầu: Biên dịch, sau đó triển khai đoạn hợp đồng thông minh trên nền tảng Ethereum để xem kết quả như thế nào, và ghi nhận vào báo cáo?

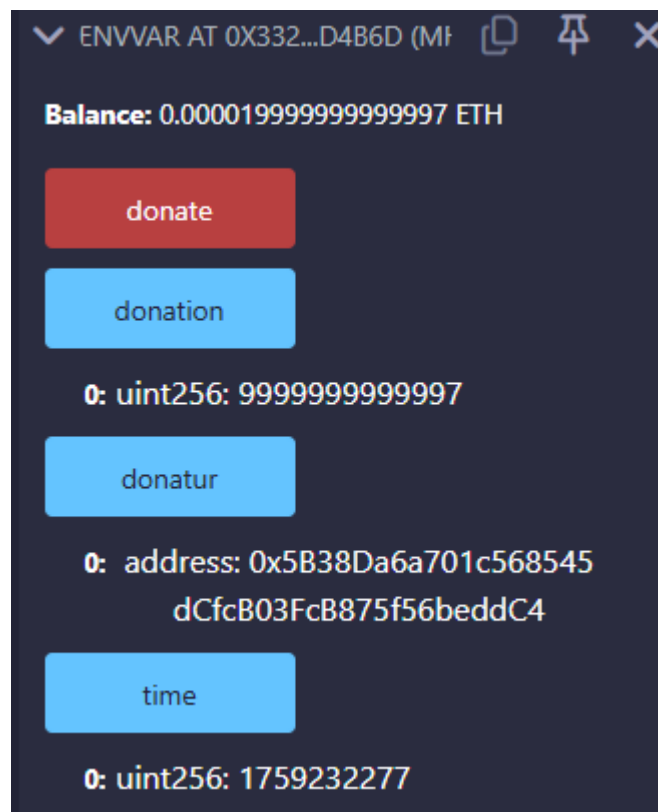
- Kết quả biên dịch



- Kết quả triển khai

```
[vm] from: 0x583...eddC4 to: EnvVar.(constructor) value: 0 wei data: 0x610...30034 logs: 0 hash: 0x04f...ae5d7
status      0x1 Transaction mined and execution succeed
transaction hash  0x04fdb225945291fd1da8b477f79074eaddc413893e20c94accd6a34c128ae5d7
block hash      0xb208640ec6812d7dd829c1f8b80e42dda8a759b7c994976bac3e846cfe3ca64fb
block number     1
contract address 0xd9145CCE52D386f254917e481eB44e9943F39138
from             0x5838Da6a701c568545dCfcB03FcB875f56beddC4
to              EnvVar.(constructor)
```

- **Kết luận:** Hợp đồng đã hoạt động đúng theo thiết kế, chứng minh khả năng truy cập và lưu trữ các biến môi trường quan trọng sau: **msg.sender** (người gửi), **msg.value** (giá trị ETH) và **block.timestamp** (thời gian block)

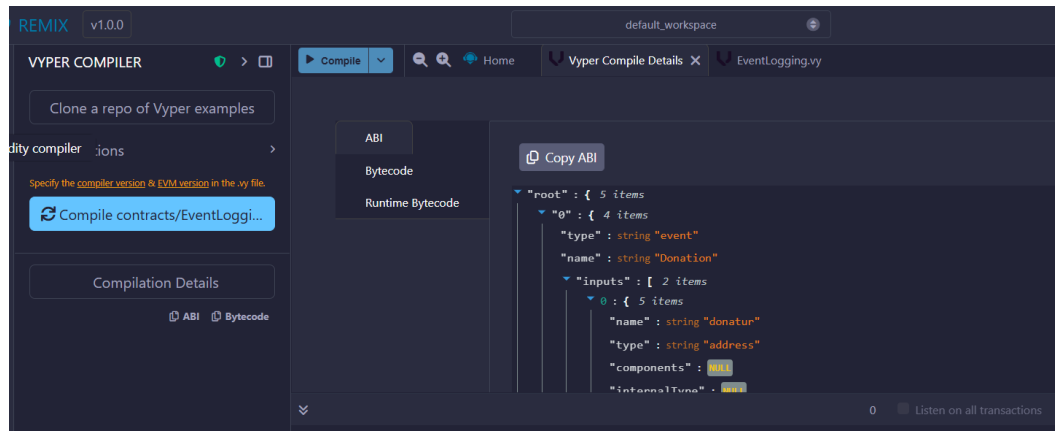


## vi. Event logging

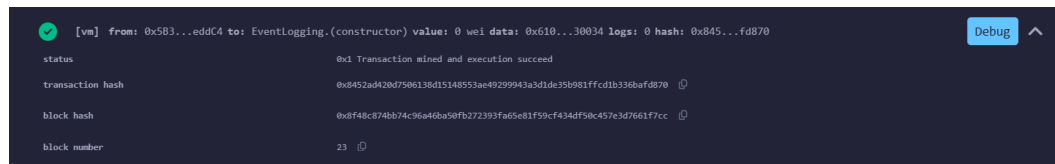
Yêu cầu: Biên dịch, sau đó triển khai đoạn hợp đồng thông minh trên nền tảng Ethereum để xem kết quả như thế nào, và ghi nhận vào báo cáo?

- Kết quả biên dịch



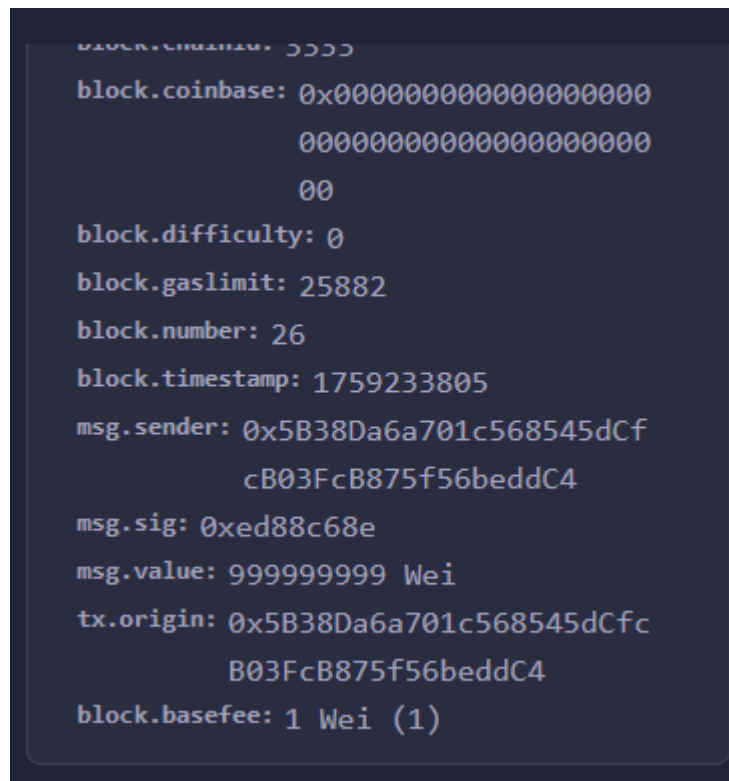


## - Kết quả triển khai



## - Kết luận:

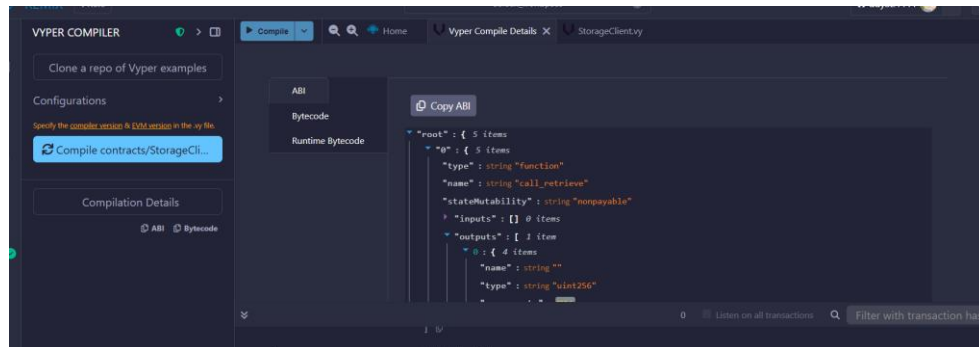
Hàm donate() đã thực hiện đúng chức năng của mình là phát ra một sự kiện Donation chứa thông tin chi tiết về người quyên góp (msg.sender) và số lượng tiền (msg.value).



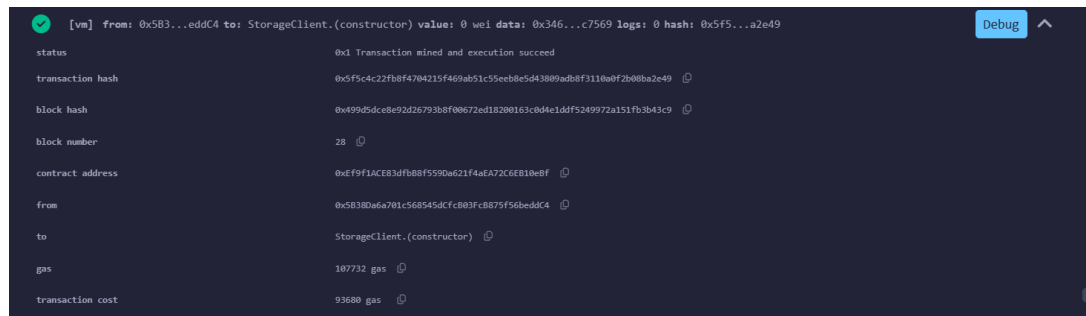
## vii. Interface

Yêu cầu: Biên dịch, sau đó triển khai đoạn hợp đồng thông minh trên nền tảng Ethereum để xem kết quả như thế nào, và ghi nhận vào báo cáo?

- Kết quả biên dịch



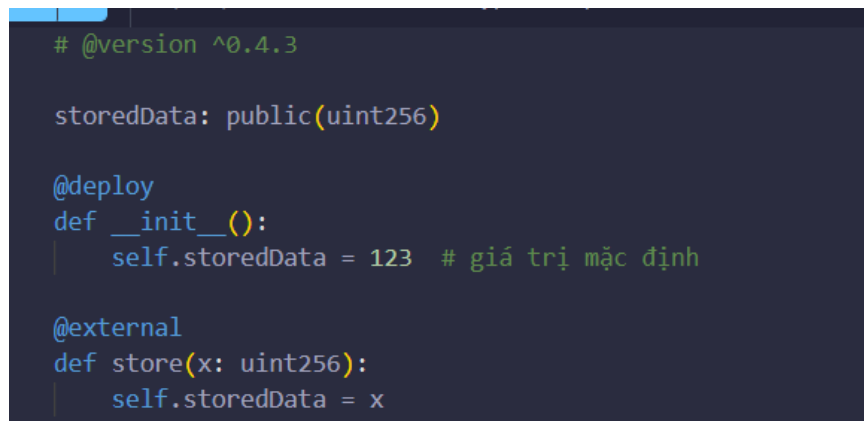
- Kết quả triển khai






- Kết luận:

Sau khi triển khai hợp đồng Storage.vy lưu trữ một giá trị số nguyên và triển khai hợp đồng StorageClient.vy với địa chỉ của Storage, việc gọi hàm call\_retrieve() từ StorageClient đã trả về chính xác giá trị mà hàm retrieve() trong Storage.vy lưu trữ.

- Hàm retrieve trong Storage.vy lưu 123



- Khi call\_retrieve() trong StorageClient

▼ STORAGECLIENT AT 0XEF9...10)   

**Balance:** 0 ETH

call\_retrieve

0: uint256: 123