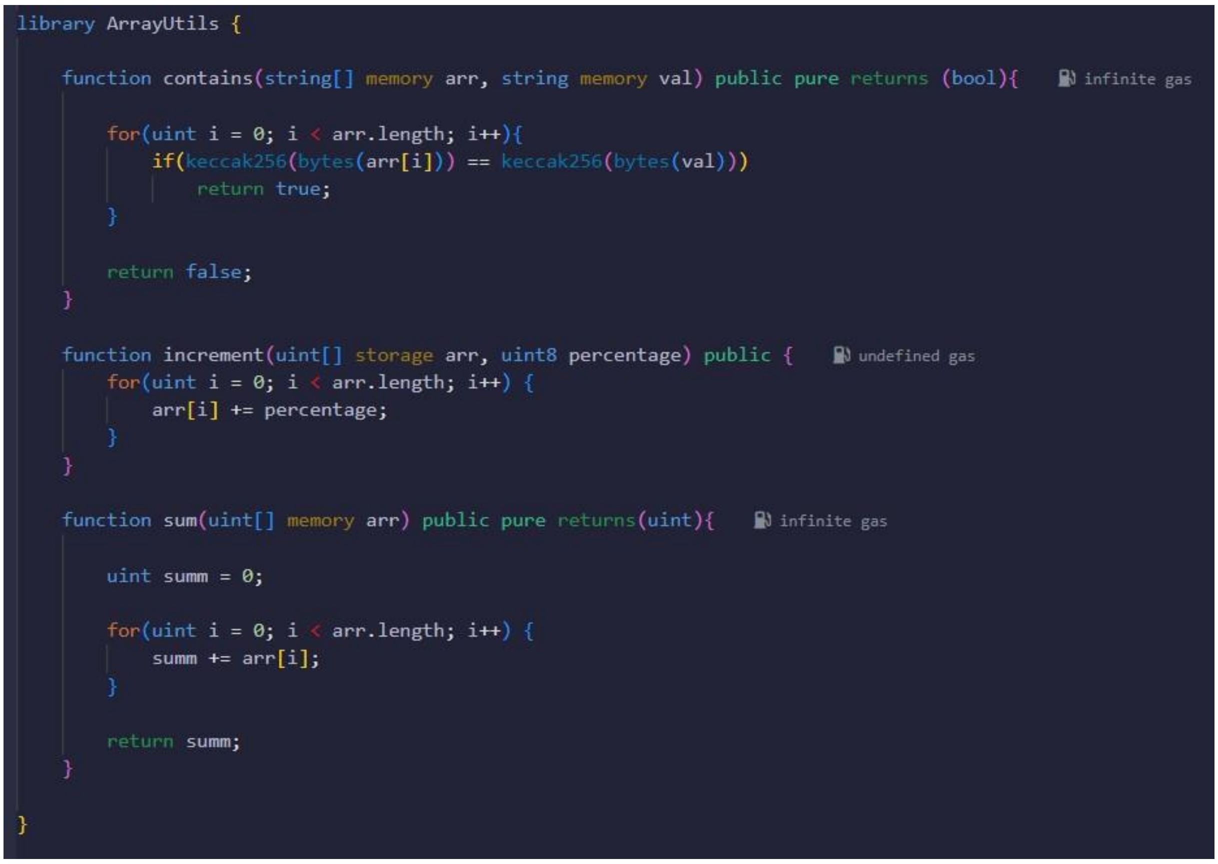
Introduction to Blockchain and Smart Contracts

Assignment 5 - Arda Harman (Group 22)

1.1 - Implementation of a library for handling arrays



```
library ArrayUtils {

    function contains(string[] memory arr, string memory val) public pure returns (bool){    infinite gas

        for(uint i = 0; i < arr.length; i++){
            if(keccak256(bytes(arr[i])) == keccak256(bytes(val)))
                return true;
        }

        return false;
    }

    function increment(uint[] storage arr, uint8 percentage) public {    undefined gas
        for(uint i = 0; i < arr.length; i++) {
            arr[i] += percentage;
        }
    }

    function sum(uint[] memory arr) public pure returns(uint){    infinite gas

        uint summ = 0;

        for(uint i = 0; i < arr.length; i++) {
            summ += arr[i];
        }

        return summ;
    }

}
```

- Contains
  ➔ This function checks whether the given string exists at the given array or not

  At contains, I used memory for variables arr and for val since they are temporary…

  - They will be only used when function is executed

  Since the function returns boolean, (whether it contains the given string or not) I used return bool

  I used keccak256(bytes(…)) to compare the the equality of 2 strings

- Increment
  - ➔ This funtction just increments every element of the array by the given percentage

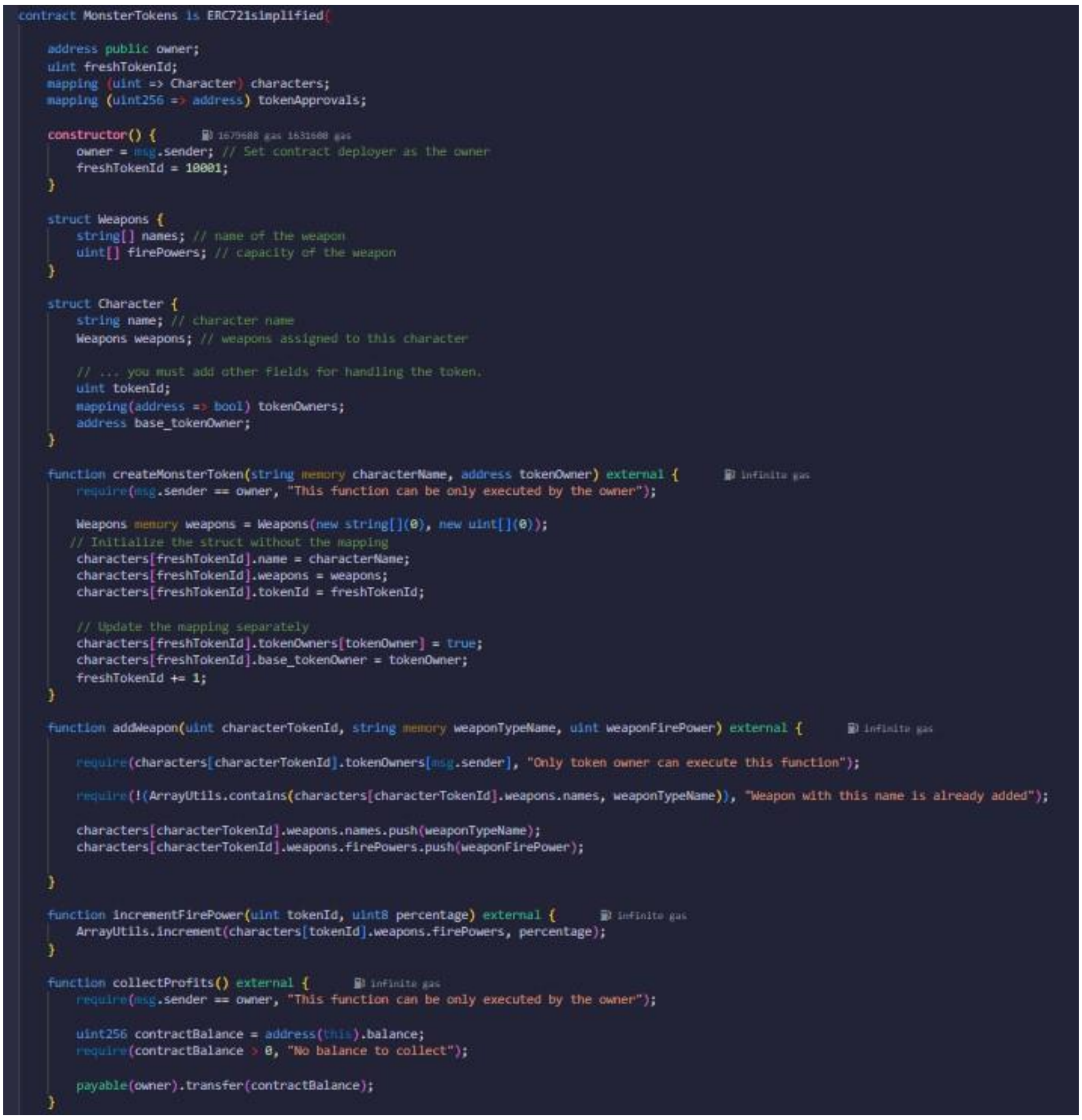
  Here, I used “storage” for the array. That is because the changes at the array will be permanent

    - We are not returning a temporary array and assigning it

- Sum
  - ➔ This function is for summing up all the elements of an array and returning the summation

  I used “memory” since the array I use in this function (array which is passed as an argument) is temporary

1.2 - Implementation of MonsterTokens contract



```
contract MonsterTokens is ERC721simplified{

    address public owner;
    uint freshTokenId;
    mapping (uint => Character) characters;
    mapping (uint256 => address) tokenApprovals;

    constructor() {      1679688 gas 1631688 gas
        owner = msg.sender; // Set contract deployer as the owner
        freshTokenId = 10001;
    }

    struct Weapons {
        string[] names; // name of the weapon
        uint[] firePowers; // capacity of the weapon
    }

    struct Character {
        string name; // character name
        Weapons weapons; // weapons assigned to this character

        // ... you must add other fields for handling the token.
        uint tokenId;
        mapping(address => bool) tokenOwners;
        address base_tokenOwner;
    }

    function createMonsterToken(string memory characterName, address tokenOwner) external {      infinite gas
        require(msg.sender == owner, "This function can be only executed by the owner");

        Weapons memory weapons = Weapons(new string[](0), new uint[](0));
       // Initialize the struct without the mapping
        characters[freshTokenId].name = characterName;
        characters[freshTokenId].weapons = weapons;
        characters[freshTokenId].tokenId = freshTokenId;

        // Update the mapping separately
        characters[freshTokenId].tokenOwners[tokenOwner] = true;
        characters[freshTokenId].base_tokenOwner = tokenOwner;
        freshTokenId += 1;
    }

    function addWeapon(uint characterTokenId, string memory weaponTypeName, uint weaponFirePower) external {      infinite gas

        require(characters[characterTokenId].tokenOwners[msg.sender], "Only token owner can execute this function");

        require(!(ArrayUtils.contains(characters[characterTokenId].weapons.names, weaponTypeName)), "Weapon with this name is already added");

        characters[characterTokenId].weapons.names.push(weaponTypeName);
        characters[characterTokenId].weapons.firePowers.push(weaponFirePower);

    }

    function incrementFirePower(uint tokenId, uint8 percentage) external {      infinite gas
        ArrayUtils.increment(characters[tokenId].weapons.firePowers, percentage);
    }

    function collectProfits() external {      infinite gas
        require(msg.sender == owner, "This function can be only executed by the owner");

        uint256 contractBalance = address(this).balance;
        require(contractBalance > 0, "No balance to collect");

        payable(owner).transfer(contractBalance);
    }
```

- createMonsterToken function
  ➔ It creates a Character and adds it to the characters array

For “characterName” I had used memory because it is just used while the execution of the createMonsterToken function and then it is not used… so that is the reason for the usage of memory

It only works if the function is called by the owner

- addWeapon function
  ➔ Adds a weapon to the character

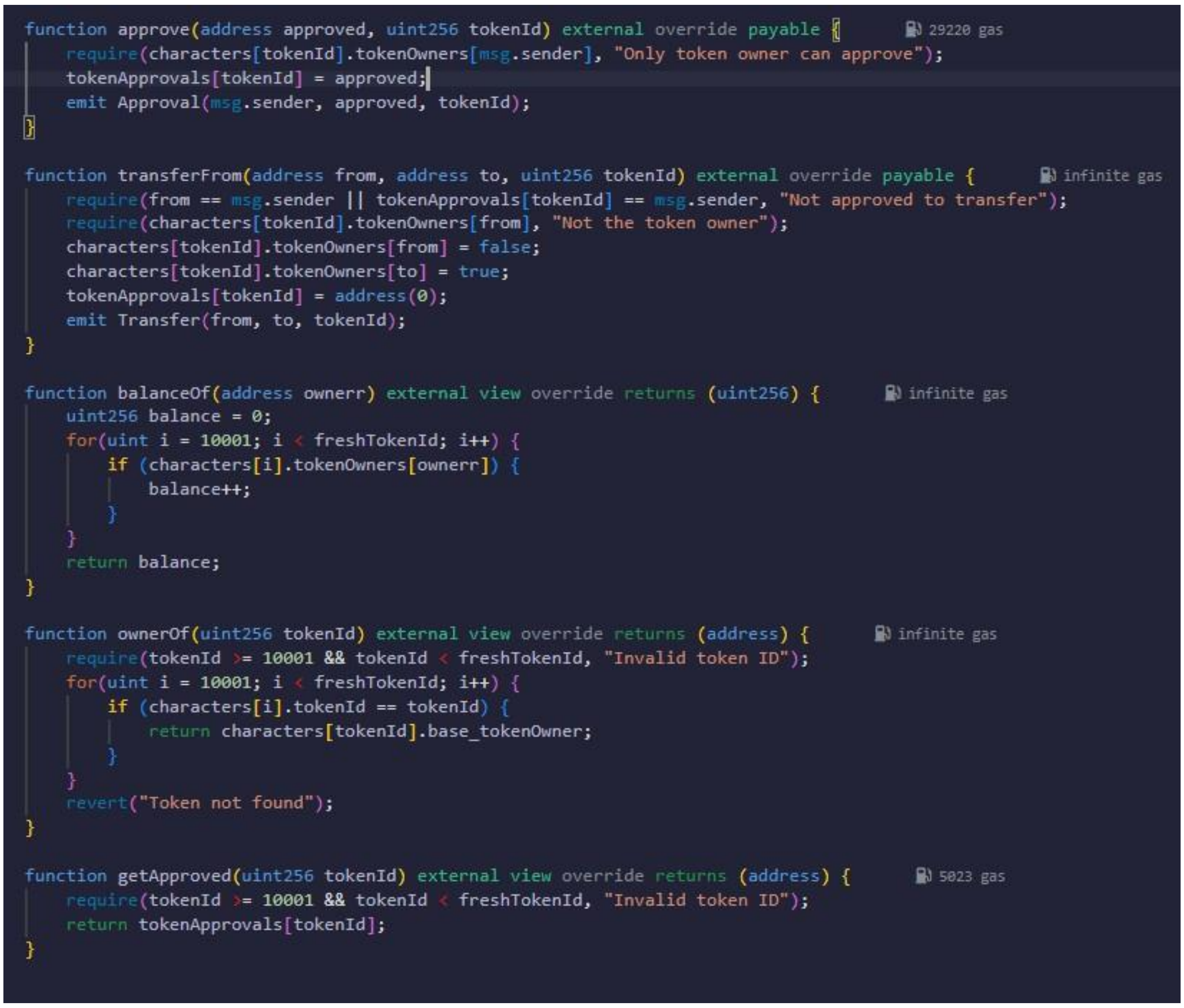
If there doesn’t exist a weapon already and this function is executed by the owner, then

it gets executed. Also, memory is used here again since weaponType is only used during the execution of the function

- incrementFirePower function
  ➔ Increments the power of all of the weapons of a characters

- collectProfits function
  ➔ This function is for collecting the recieved payments

1.3 - Implementation of ERC721simplified interface



```
function approve(address approved, uint256 tokenId) external override payable {      29220 gas
    require(characters[tokenId].tokenOwners[msg.sender], "Only token owner can approve");
    tokenApprovals[tokenId] = approved;
    emit Approval(msg.sender, approved, tokenId);
}

function transferFrom(address from, address to, uint256 tokenId) external override payable {      infinite gas
    require(from == msg.sender || tokenApprovals[tokenId] == msg.sender, "Not approved to transfer");
    require(characters[tokenId].tokenOwners[from], "Not the token owner");
    characters[tokenId].tokenOwners[from] = false;
    characters[tokenId].tokenOwners[to] = true;
    tokenApprovals[tokenId] = address(0);
    emit Transfer(from, to, tokenId);
}

function balanceOf(address ownerr) external view override returns (uint256) {      infinite gas
    uint256 balance = 0;
    for(uint i = 10001; i < freshTokenId; i++) {
        if (characters[i].tokenOwners[ownerr]) {
            balance++;
        }
    }
    return balance;
}

function ownerOf(uint256 tokenId) external view override returns (address) {      infinite gas
    require(tokenId >= 10001 && tokenId < freshTokenId, "Invalid token ID");
    for(uint i = 10001; i < freshTokenId; i++) {
        if (characters[i].tokenId == tokenId) {
            return characters[tokenId].base_tokenOwner;
        }
    }
    revert("Token not found");
}

function getApproved(uint256 tokenId) external view override returns (address) {      5023 gas
    require(tokenId >= 10001 && tokenId < freshTokenId, "Invalid token ID");
    return tokenApprovals[tokenId];
}
```

- approve
  ➔ This function is for approving an address to operate on a token

- transferForm
  ➔ This function transfers the ownership of a token

- balanceOf
  ➔ Returns the number of tokens owned by the address

- ownerOf
  ➔ This function returns the owner of the token with the given id

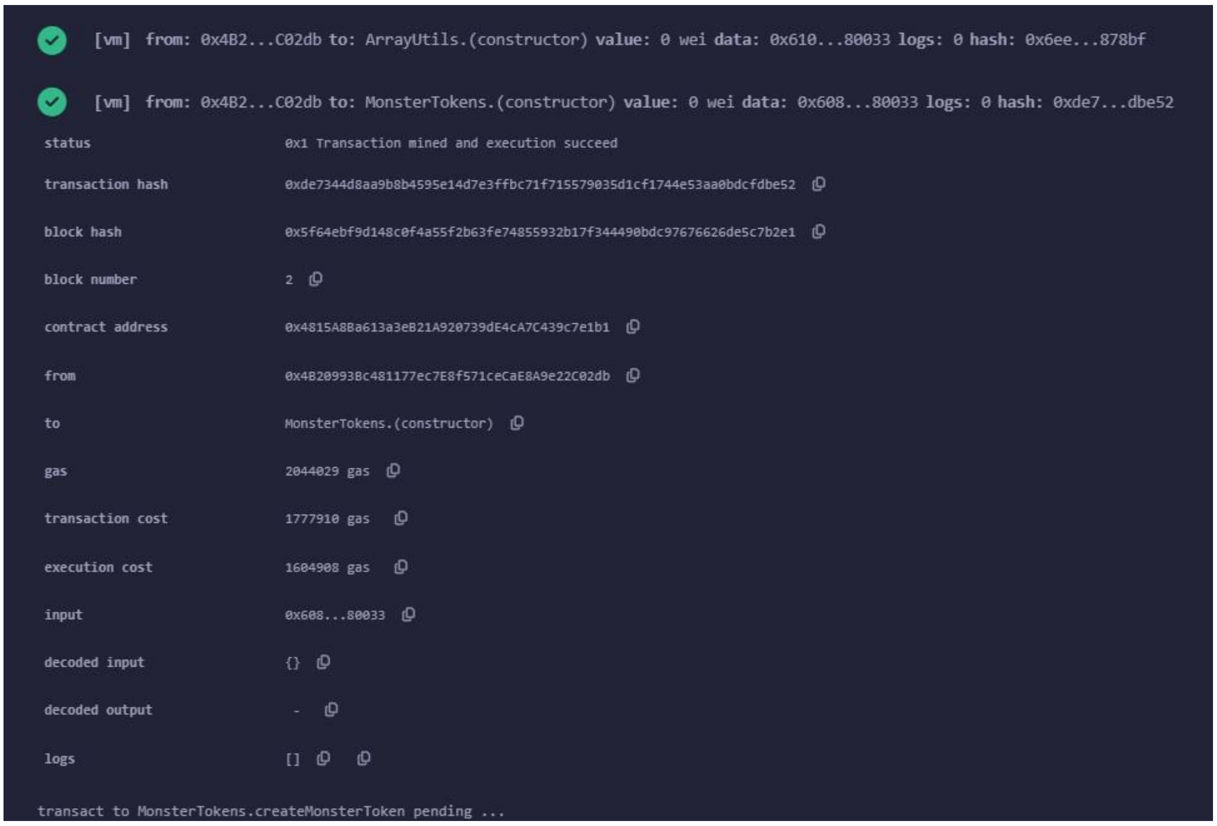
- getApproved
  ➔ This function must return the approved address

2. Testing the contract

1) Assigning roles

Role	Address
GameMaster	0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
TokenOwner1	0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB
TokenOwner2	0x617F2E2fD72FD9D5503197092aC168c91465E7f2

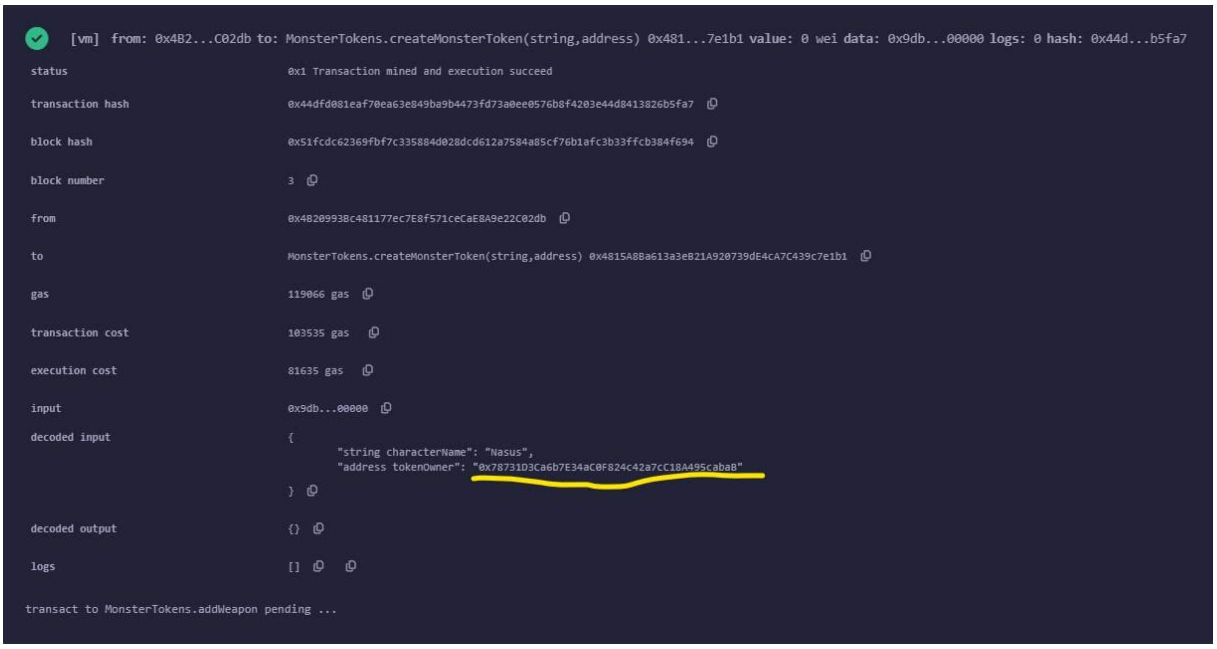
2) Deploying MonsterTokens with address GameMaster



```
[vm] from: 0x4B2...C02db to: ArrayUtils.(constructor) value: 0 wei data: 0x610...80033 logs: 0 hash: 0x6ee...878bf

[vm] from: 0x4B2...C02db to: MonsterTokens.(constructor) value: 0 wei data: 0x608...80033 logs: 0 hash: 0xde7...dbe52
status                 0x1 Transaction mined and execution succeed
transaction hash       0xde7344d8aa9b8b4595e14d7e3ffbc71f715579035d1cf1744e53aa0bdcfdbe52
block hash             0x5f64ebf9d148c0f4a55f2b63fe74855932b17f344490bdc97676626de5c7b2e1
block number           2
contract address       0x4815A8Ba613a3eB21A920739dE4cA7C439c7e1b1
from                   0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
to                     MonsterTokens.(constructor)
gas                    2044029 gas
transaction cost       1777910 gas
execution cost         1604908 gas
input                  0x608...80033
decoded input          {}
decoded output          -
logs                   []

transact to MonsterTokens.createMonsterToken pending ...
```

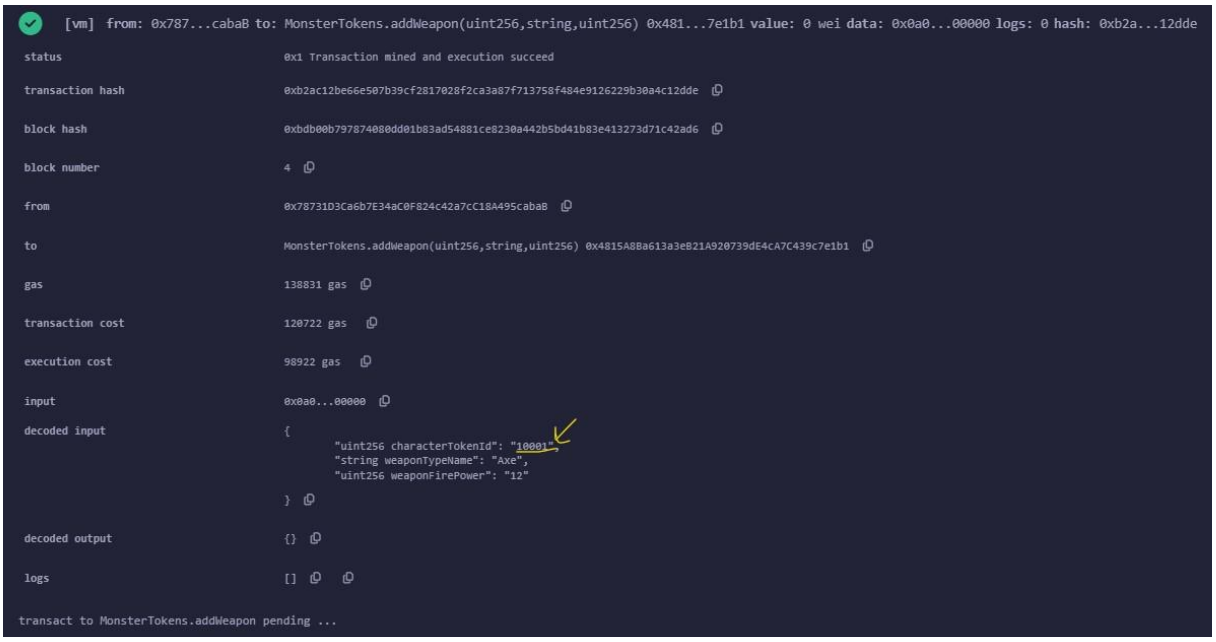
3) Create two tokens, each one owned by one TokenOwner. Add two weapons to each token with different data.

- tokenOwner1 (I named first token as “Nasus” and underlined the corresponding adderess with yellow)

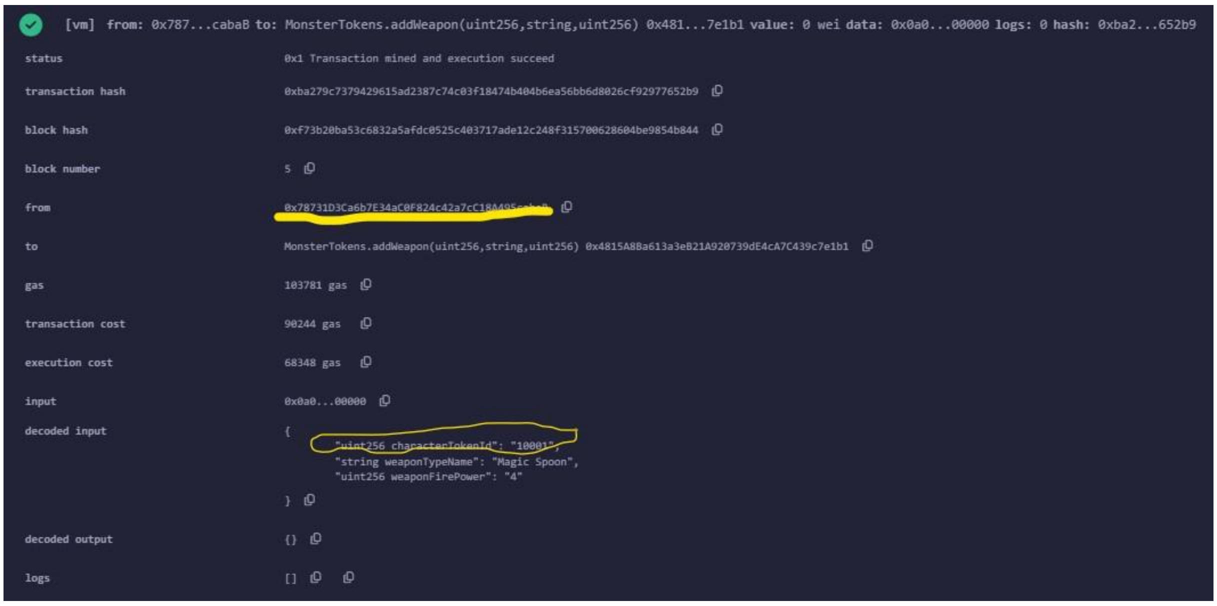


```
[vm]  from: 0x4B2...C02db to: MonsterTokens.createMonsterToken(string,address) 0x481...7e1b1 value: 0 wei data: 0x9db...00000 logs: 0 hash: 0x44d...b5fa7
status                   0x1 Transaction mined and execution succeed
transaction hash         0x44dfd081eaf70ea63e849ba9b4473fd73a0ee0576b8f4203e44d8413826b5fa7
block hash               0x51fcdc62369fbf7c335884d028dcd612a7584a85cf76b1afc3b33ffcb384f694
block number             3
from                     0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
to                       MonsterTokens.createMonsterToken(string,address) 0x4815A8Ba613a3eB21A920739dE4cA7C439c7e1b1
gas                      119066 gas
transaction cost         103535 gas
execution cost           81635 gas
input                    0x9db...00000
decoded input            {
                                 "string characterName": "Nasus",
                                 "address tokenOwner": "0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB"
                         }
decoded output           {}
logs                     []

transact to MonsterTokens.addWeapon pending ...
```

- First weapon of the tokenOwner1 (named axe)
    - By the way, it can be seen that axe really belongs to the token with the id 10001

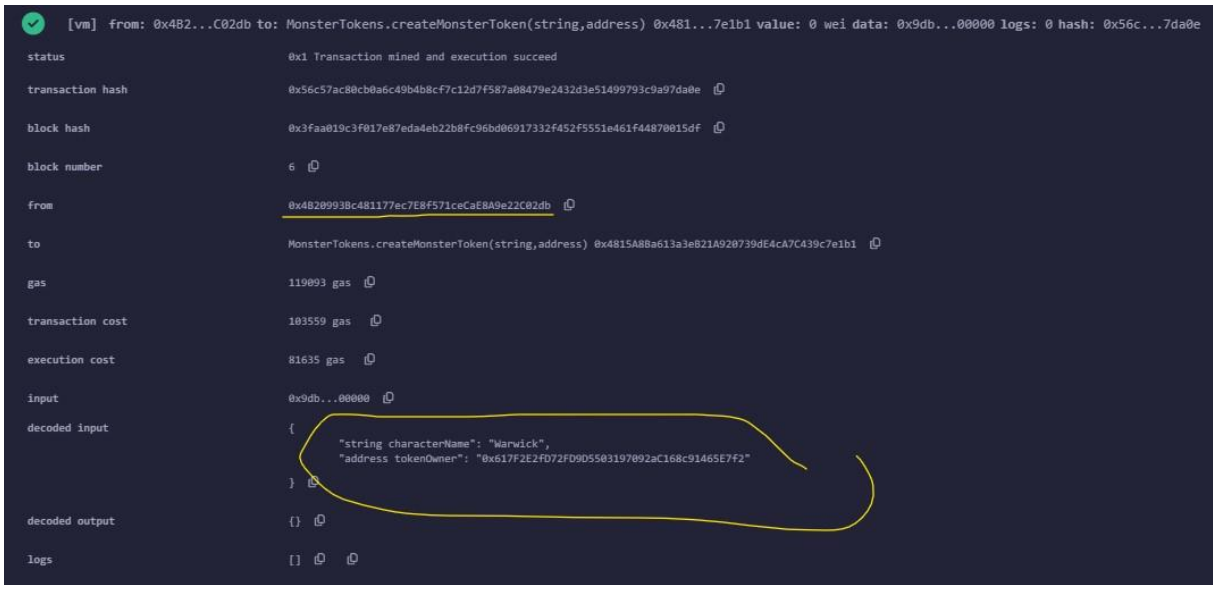


```
[vm]  from: 0x787...cabaB to: MonsterTokens.addWeapon(uint256,string,uint256) 0x481...7e1b1 value: 0 wei data: 0x0a0...00000 logs: 0 hash: 0xb2a...12dde
status                   0x1 Transaction mined and execution succeed
transaction hash         0xb2ac12be66e507b39cf2817028f2ca3a87f713758f484e9126229b30a4c12dde
block hash               0xbdb00b797874080dd01b83ad54881ce8230a442b5bd41b83e413273d71c42ad6
block number             4
from                     0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB
to                       MonsterTokens.addWeapon(uint256,string,uint256) 0x4815A8Ba613a3eB21A920739dE4cA7C439c7e1b1
gas                      138831 gas
transaction cost         120722 gas
execution cost           98922 gas
input                    0x0a0...00000
decoded input            {
                                 "uint256 characterTokenId": "10001",
                                 "string weaponTypeName": "Axe",
                                 "uint256 weaponFirePower": "12"
                         }
decoded output           {}
logs                     []

transact to MonsterTokens.addWeapon pending ...
```



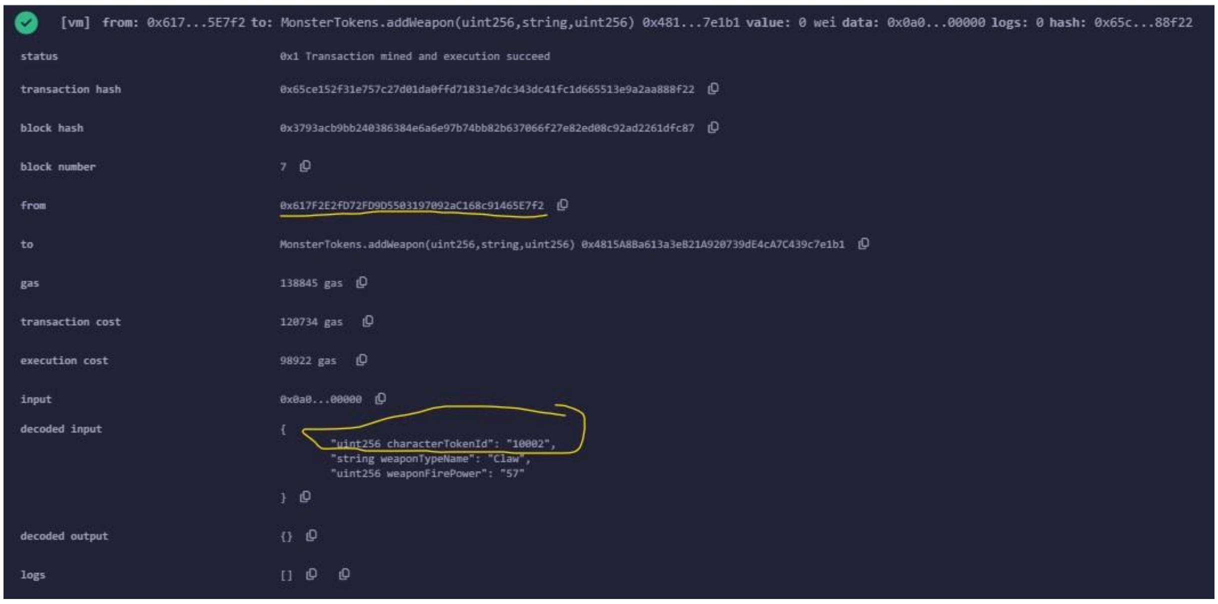
```
[vm] from: 0x787...cabaB to: MonsterTokens.addWeapon(uint256,string,uint256) 0x481...7e1b1 value: 0 wei data: 0x0a0...00000 logs: 0 hash: 0xba2...652b9
status                  0x1 Transaction mined and execution succeed
transaction hash        0xba279c7379429615ad2387c74c03f18474b404b6ea56bb6d8026cf92977652b9
block hash              0xf73b20ba53c6832a5afdc0525c403717ade12c248f315700628604be9854b844
block number            5
from                    0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB
to                      MonsterTokens.addWeapon(uint256,string,uint256) 0x4815A88a613a3eB21A920739dE4cA7C439c7e1b1
gas                     103781 gas
transaction cost        90244 gas
execution cost          68348 gas
input                   0x0a0...00000
decoded input           {
                            "uint256 characterTokenId": "10001",
                            "string weaponTypeName": "Magic Spoon",
                            "uint256 weaponFirePower": "4"
                        }
decoded output          {}
logs                    []
```

- tokenOwner2 (I named first token as “Warwick” and underlined the corresponding adderess with yellow)



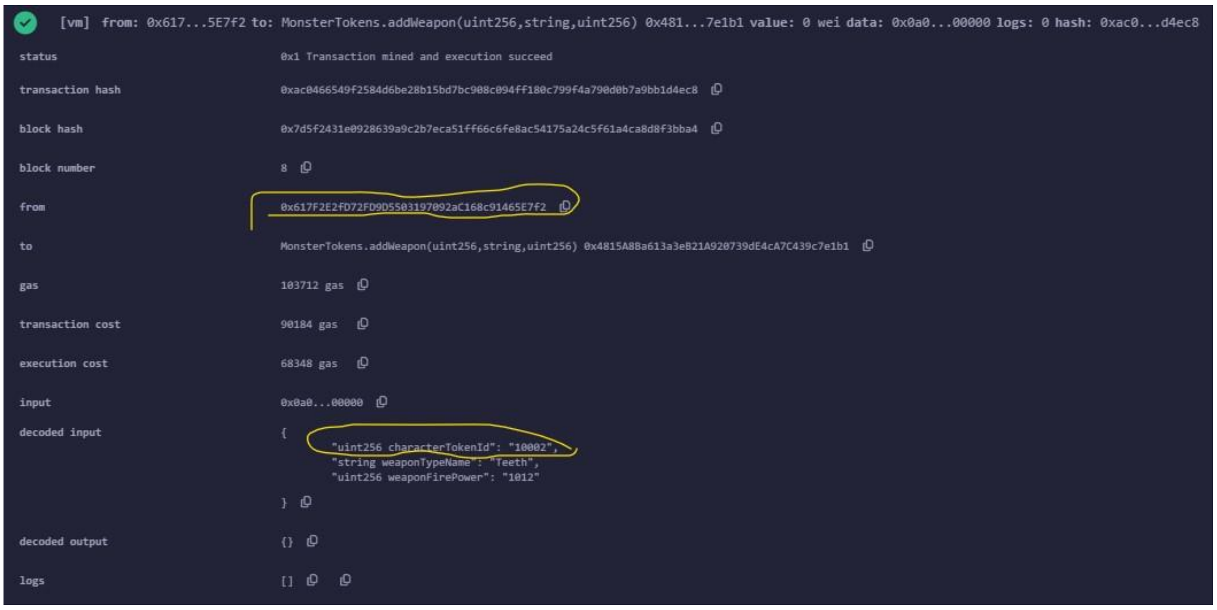
```
[vm] from: 0x4B2...C02db to: MonsterTokens.createMonsterToken(string,address) 0x481...7e1b1 value: 0 wei data: 0x9db...00000 logs: 0 hash: 0x56c...7da0e
status                  0x1 Transaction mined and execution succeed
transaction hash        0x56c57ac80cb0a6c49b4b8cf7c12d7f587a08479e2432d3e51499793c9a97da0e
block hash              0x3faa019c3f017e87eda4eb22b8fc96bd06917332f452f5551e461f44870015df
block number            6
from                    0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
to                      MonsterTokens.createMonsterToken(string,address) 0x4815A88a613a3eB21A920739dE4cA7C439c7e1b1
gas                     119093 gas
transaction cost        103559 gas
execution cost          81635 gas
input                   0x9db...00000
decoded input           {
                            "string characterName": "Warwick",
                            "address tokenOwner": "0x617F2E2fD72FD9D5503197092aC168c91465E7f2"
                        }
decoded output          {}
logs                    []
```

- First weapon of the tokenOwner2 (named “Claw”) (it really belongd to the token with the id 10002)



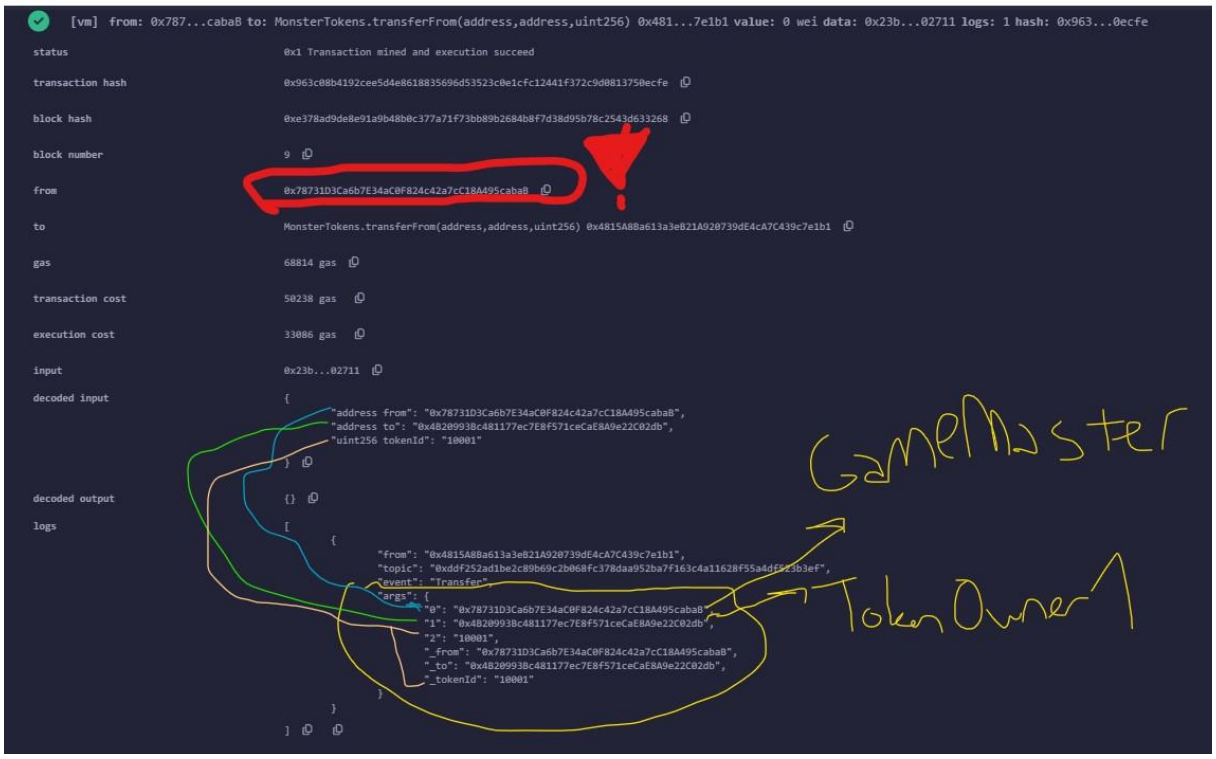
```
[vm] from: 0x617...5E7f2 to: MonsterTokens.addWeapon(uint256,string,uint256) 0x481...7e1b1 value: 0 wei data: 0x0a0...00000 logs: 0 hash: 0x65c...88f22
status              0x1 Transaction mined and execution succeed
transaction hash    0x65ce152f31e757c27d01da0ffd71831e7dc343dc41fc1d665513e9a2aa888f22
block hash          0x3793acb9bb240386384e6a6e97b74bb82b637066f27e82ed08c92ad2261dfc87
block number        7
from                0x617F2E2fD72FD9D5503197092aC168c91465E7f2
to                  MonsterTokens.addWeapon(uint256,string,uint256) 0x4815A8Ba613a3eB21A920739dE4cA7C439c7e1b1
gas                 138845 gas
transaction cost    120734 gas
execution cost      98922 gas
input               0x0a0...00000
decoded input       {
                        "uint256 characterTokenId": "10002",
                        "string weaponTypeName": "Claw",
                        "uint256 weaponFirePower": "57"
                    }
decoded output      {}
logs                []
```

- Second weapon of the tokenOwner2 (named “Teeth”)



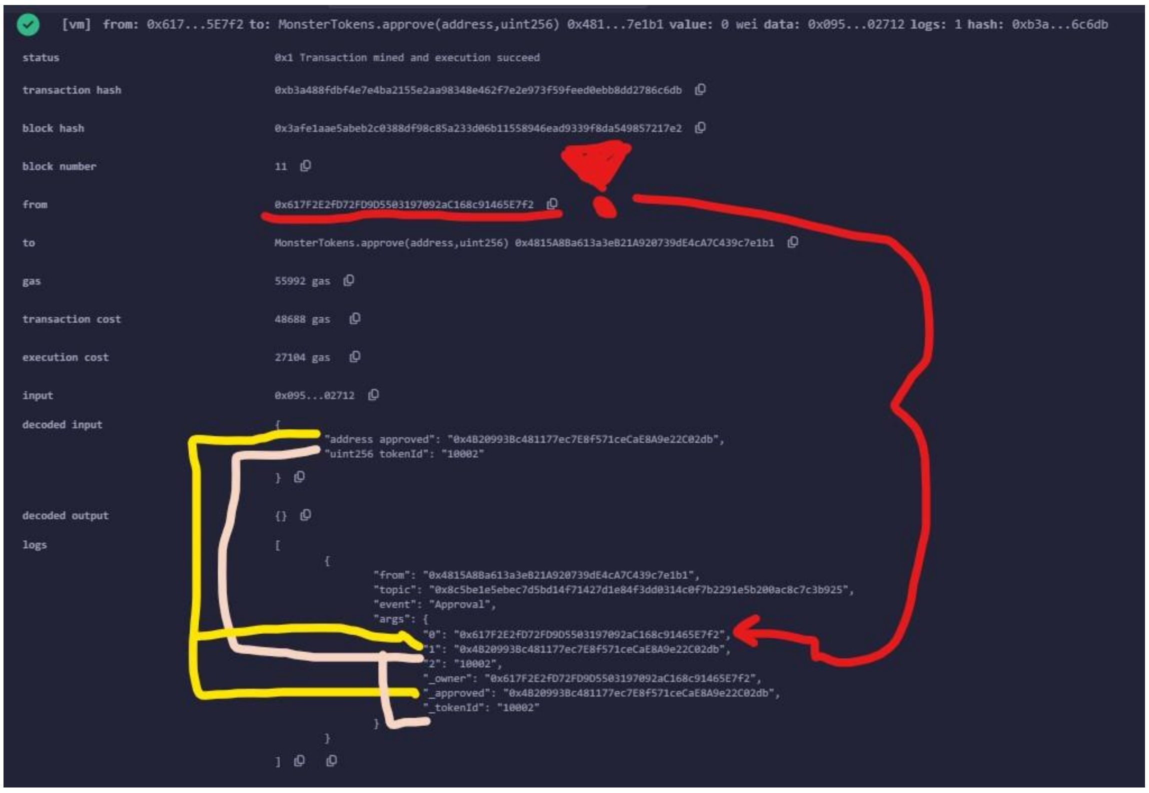
```
[vm] from: 0x617...5E7f2 to: MonsterTokens.addWeapon(uint256,string,uint256) 0x481...7e1b1 value: 0 wei data: 0x0a0...00000 logs: 0 hash: 0xac0...d4ec8
status              0x1 Transaction mined and execution succeed
transaction hash    0xac0466549f2584d6be28b15bd7bc908c094ff180c799f4a790d0b7a9bb1d4ec8
block hash          0x7d5f2431e0928639a9c2b7eca51ff66c6fe8ac54175a24c5f61a4ca8d8f3bba4
block number        8
from                0x617F2E2fD72FD9D5503197092aC168c91465E7f2
to                  MonsterTokens.addWeapon(uint256,string,uint256) 0x4815A8Ba613a3eB21A920739dE4cA7C439c7e1b1
gas                 103712 gas
transaction cost    90184 gas
execution cost      68348 gas
input               0x0a0...00000
decoded input       {
                        "uint256 characterTokenId": "10002",
                        "string weaponTypeName": "Teeth",
                        "uint256 weaponFirePower": "1012"
                    }
decoded output      {}
logs                []
```

4) Perform a transfer of one of the tokens to GameMaster (I transferred first token)



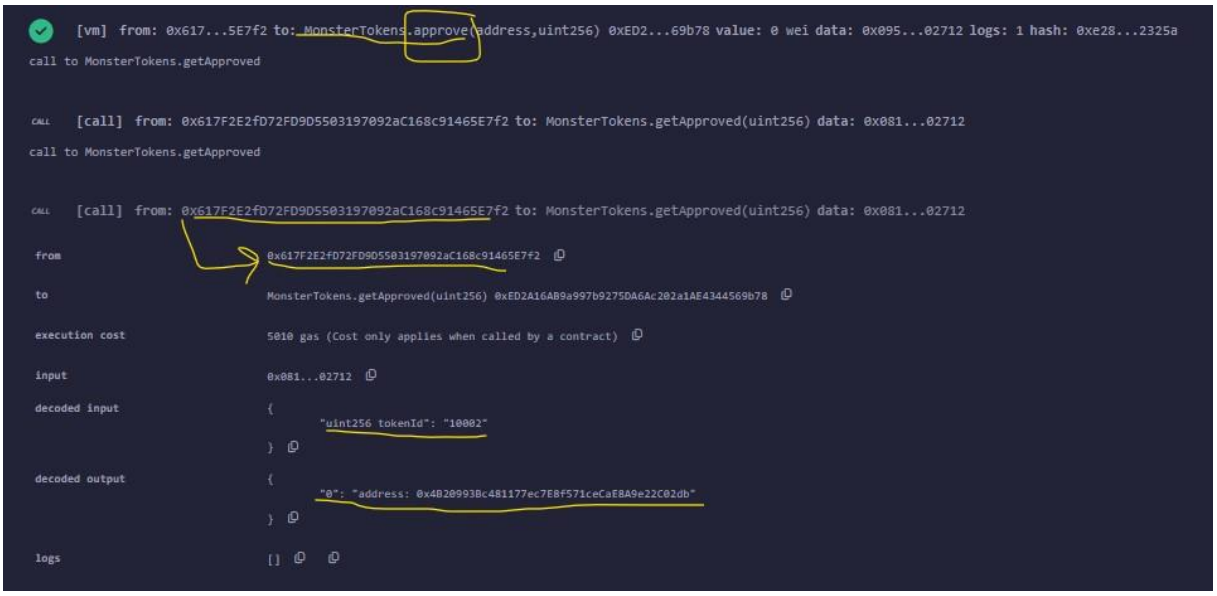
```
[vm] from: 0x787...cabaB to: MonsterTokens.transferFrom(address,address,uint256) 0x481...7e1b1 value: 0 wei data: 0x23b...02711 logs: 1 hash: 0x963...0ecfe
status                 0x1 Transaction mined and execution succeed
transaction hash       0x963c08b4192cee5d4e8618835696d53523c0e1cfc12441f372c9d0813750ecfe
block hash             0xe378ad9de8e91a9b48b0c377a71f73bb89b2684b8f7d38d95b78c2543d633268
block number           9
from                   0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB
to                     MonsterTokens.transferFrom(address,address,uint256) 0x4815A88a613a3eB21A920739dE4cA7C439c7e1b1
gas                    68814 gas
transaction cost       50238 gas
execution cost         33086 gas
input                  0x23b...02711
decoded input          {
                           "address from": "0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB",
                           "address to": "0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db",
                           "uint256 tokenId": "10001"
                       }
decoded output         {}
logs                   [
                           {
                               "from": "0x4815A88a613a3eB21A920739dE4cA7C439c7e1b1",
                               "topic": "0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef",
                               "event": "Transfer",
                               "args": {
                                   "0": "0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB",
                                   "1": "0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db",
                                   "2": "10001",
                                   "_from": "0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB",
                                   "_to": "0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db",
                                   "_tokenId": "10001"
                               }
                           }
                       ]
```

5) Approve GameMaster to operate the other token. (I used 2nd token to approve)



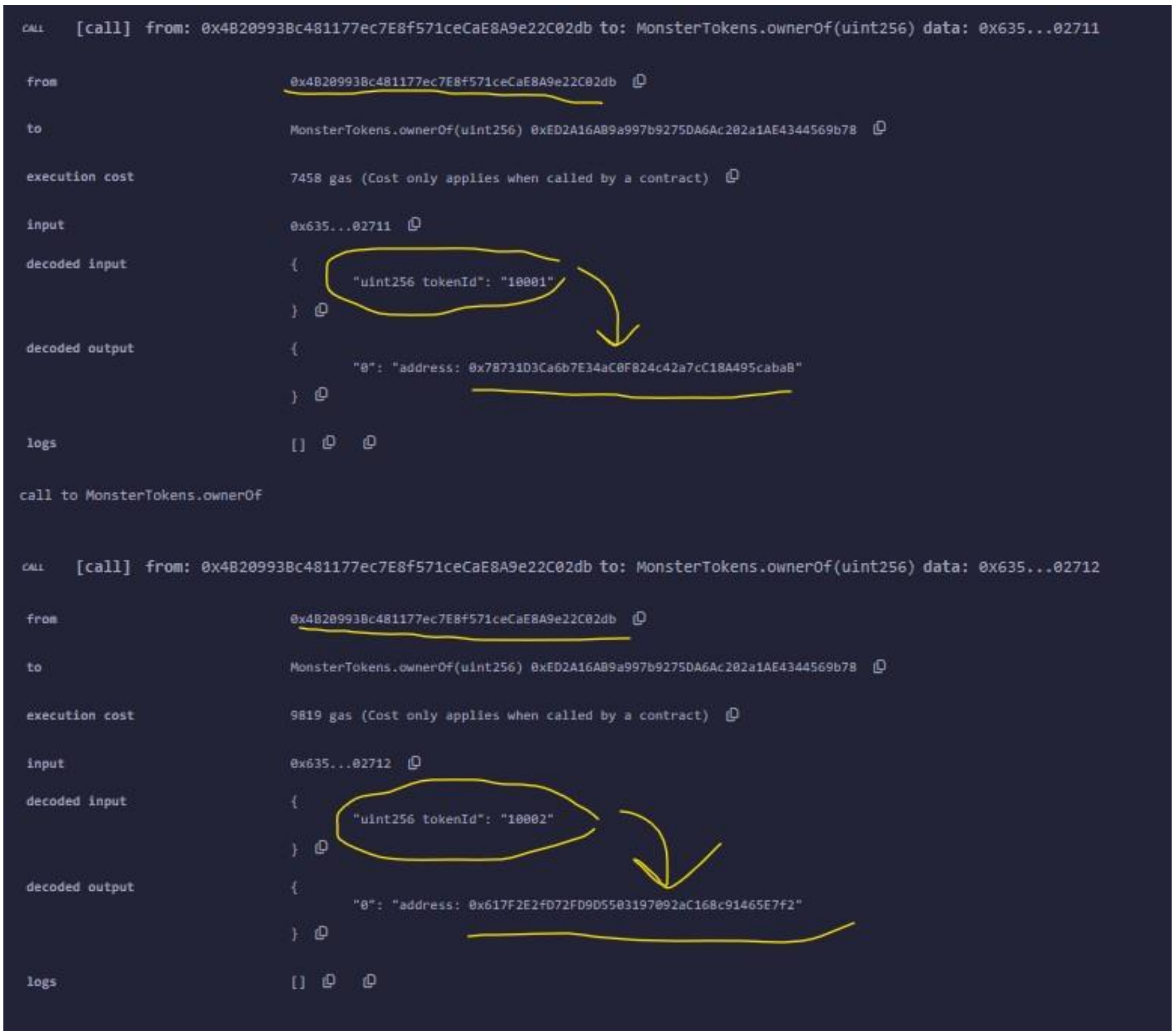
```
[vm] from: 0x617...5E7f2 to: MonsterTokens.approve(address,uint256) 0x481...7e1b1 value: 0 wei data: 0x095...02712 logs: 1 hash: 0xb3a...6c6db
status                 0x1 Transaction mined and execution succeed
transaction hash       0xb3a488fdbf4e7e4ba2155e2aa98348e462f7e2e973f59feed0ebb8dd2786c6db
block hash             0x3afe1aae5abeb2c0388df98c85a233d06b11558946ead9339f8da549857217e2
block number           11
from                   0x617F2E2fD72FD9D5503197092aC168c91465E7f2
to                     MonsterTokens.approve(address,uint256) 0x4815A88a613a3eB21A920739dE4cA7C439c7e1b1
gas                    55992 gas
transaction cost       48688 gas
execution cost         27104 gas
input                  0x095...02712
decoded input          {
                           "address approved": "0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db",
                           "uint256 tokenId": "10002"
                       }
decoded output         {}
logs                   [
                           {
                               "from": "0x4815A88a613a3eB21A920739dE4cA7C439c7e1b1",
                               "topic": "0x8c5be1e5ebec7d5bd14f71427d1e84f3dd0314c0f7b2291e5b200ac8c7c3b925",
                               "event": "Approval",
                               "args": {
                                   "0": "0x617F2E2fD72FD9D5503197092aC168c91465E7f2",
                                   "1": "0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db",
                                   "2": "10002",
                                   "_owner": "0x617F2E2fD72FD9D5503197092aC168c91465E7f2",
                                   "_approved": "0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db",
                                   "_tokenId": "10002"
                               }
                           }
                       ]
```

- getApproved



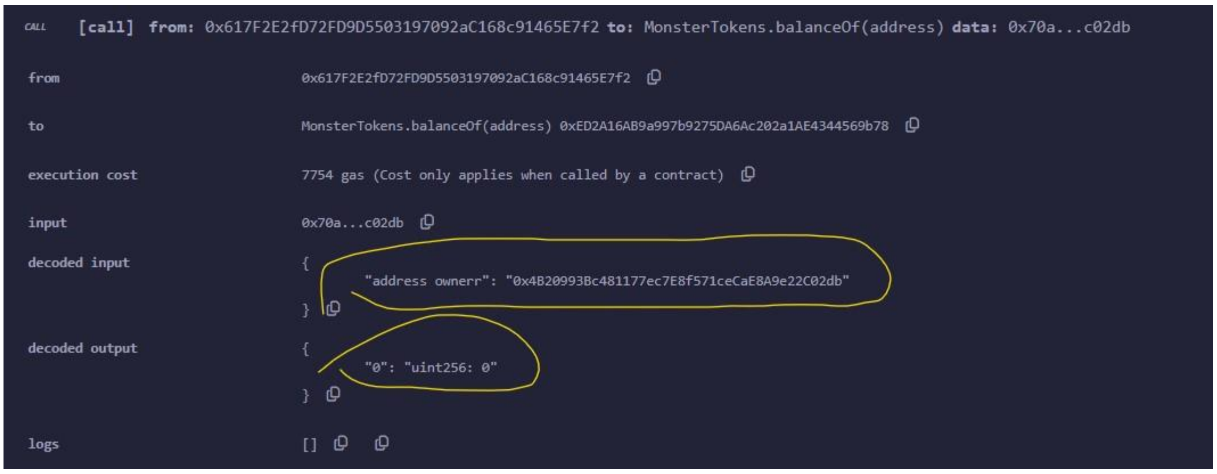
```
[vm] from: 0x617...5E7f2 to: MonsterTokens.approve(address,uint256) 0xED2...69b78 value: 0 wei data: 0x095...02712 logs: 1 hash: 0xe28...2325a
call to MonsterTokens.getApproved

CALL  [call] from: 0x617F2E2fD72FD9D5503197092aC168c91465E7f2 to: MonsterTokens.getApproved(uint256) data: 0x081...02712
call to MonsterTokens.getApproved

CALL  [call] from: 0x617F2E2fD72FD9D5503197092aC168c91465E7f2 to: MonsterTokens.getApproved(uint256) data: 0x081...02712
 from               0x617F2E2fD72FD9D5503197092aC168c91465E7f2
 to                 MonsterTokens.getApproved(uint256) 0xED2A16AB9a997b9275DA6Ac202a1AE4344569b78
 execution cost     5010 gas (Cost only applies when called by a contract)
 input              0x081...02712
 decoded input      {
                           "uint256 tokenId": "10002"
                    }
 decoded output     {
                           "0": "address: 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db"
                    }
 logs               []
```

- ownerOf



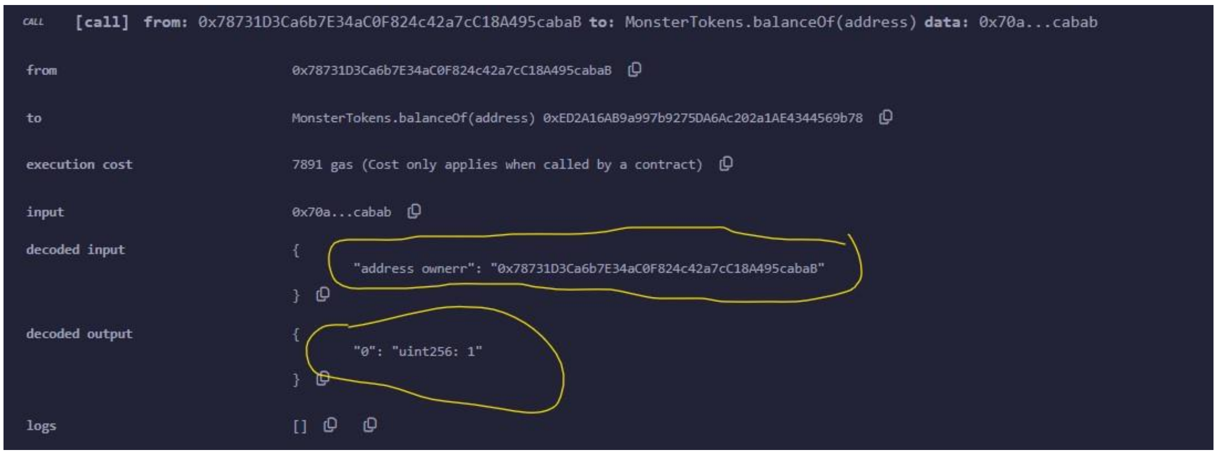
```
CALL  [call] from: 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db to: MonsterTokens.ownerOf(uint256) data: 0x635...02711
 from               0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
 to                 MonsterTokens.ownerOf(uint256) 0xED2A16AB9a997b9275DA6Ac202a1AE4344569b78
 execution cost     7458 gas (Cost only applies when called by a contract)
 input              0x635...02711
 decoded input      {
                           "uint256 tokenId": "10001"
                    }
 decoded output     {
                           "0": "address: 0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB"
                    }
 logs               []
call to MonsterTokens.ownerOf

CALL  [call] from: 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db to: MonsterTokens.ownerOf(uint256) data: 0x635...02712
 from               0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
 to                 MonsterTokens.ownerOf(uint256) 0xED2A16AB9a997b9275DA6Ac202a1AE4344569b78
 execution cost     9819 gas (Cost only applies when called by a contract)
 input              0x635...02712
 decoded input      {
                           "uint256 tokenId": "10002"
                    }
 decoded output     {
                           "0": "address: 0x617F2E2fD72FD9D5503197092aC168c91465E7f2"
                    }
 logs               []
```

- Balance off
    - GameMaster



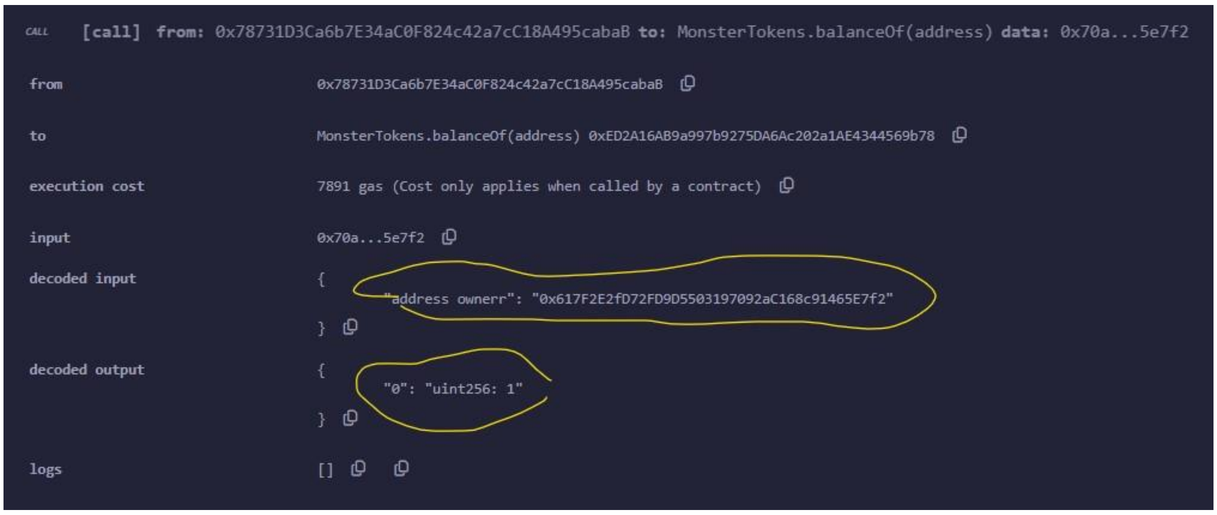
```
CALL   [call] from: 0x617F2E2fD72FD9D5503197092aC168c91465E7f2 to: MonsterTokens.balanceOf(address) data: 0x70a...c02db

from              0x617F2E2fD72FD9D5503197092aC168c91465E7f2
to                MonsterTokens.balanceOf(address) 0xED2A16AB9a997b9275DA6Ac202a1AE4344569b78
execution cost    7754 gas (Cost only applies when called by a contract)
input             0x70a...c02db
decoded input     {
                      "address ownerr": "0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db"
                  }
decoded output    {
                      "0": "uint256: 0"
                  }
logs              []
```

- Token1



```
CALL   [call] from: 0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB to: MonsterTokens.balanceOf(address) data: 0x70a...cabab

from              0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB
to                MonsterTokens.balanceOf(address) 0xED2A16AB9a997b9275DA6Ac202a1AE4344569b78
execution cost    7891 gas (Cost only applies when called by a contract)
input             0x70a...cabab
decoded input     {
                      "address ownerr": "0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB"
                  }
decoded output    {
                      "0": "uint256: 1"
                  }
logs              []
```

- Token2



```
CALL   [call] from: 0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB to: MonsterTokens.balanceOf(address) data: 0x70a...5e7f2

from              0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB
to                MonsterTokens.balanceOf(address) 0xED2A16AB9a997b9275DA6Ac202a1AE4344569b78
execution cost    7891 gas (Cost only applies when called by a contract)
input             0x70a...5e7f2
decoded input     {
                      "address ownerr": "0x617F2E2fD72FD9D5503197092aC168c91465E7f2"
                  }
decoded output    {
                      "0": "uint256: 1"
                  }
logs              []
```