Assignment no.3

Prepare your build system and Building Bitcoin Core.

a. Write Hello World smart contract in a higher programming language (Solidity).

b. Solidity example using arrays and functions

a)

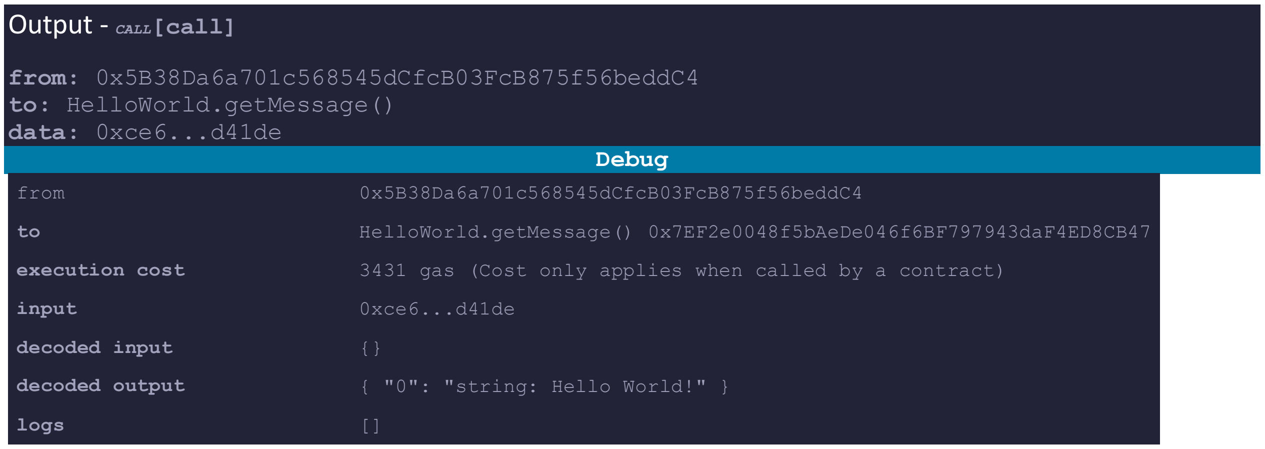
```
pragma solidity ^0.8.0;

contract HelloWorld {
    string public message;

    constructor() {
        message = "Hello World!";
    }

    function getMessage() public view returns (string memory) {
        return message;
    }

    function setMessage(string memory newMessage) public {
        message = newMessage;
    }
}
```



```
Output - CALL [call]

from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to: HelloWorld.getMessage()
data: 0xce6...d41de
                              Debug
from                      0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                        HelloWorld.getMessage() 0x7EF2e0048f5bAeDe046f6BF797943daF4ED8CB47
execution cost            3431 gas (Cost only applies when called by a contract)
input                     0xce6...d41de
decoded input             {}
decoded output            { "0": "string: Hello World!" }
logs                      []
```

```
b) pragma solidity ^0.8.0;


contract ArrayExample {
    uint[] public numbers;

    function addNumber(uint _number) public {
        numbers.push(_number);
    }

    function getNumber(uint _index) public view returns (uint) {
        require(_index < numbers.length, "Invalid index");
        return numbers[_index];
    }

    function getSum() public view returns (uint) {
        uint sum = 0;
        for (uint i = 0; i < numbers.length; i++) {
            sum += numbers[i];
        }
        return sum;
    }
}
```

OUTPUT:

```
[vm]
from: 0x5B3...eddC4
to: ArrayExample.addNumber(uint256) 0xf8e...9fBe8
value: 0 wei
data: 0xfce...00001
logs: 0
hash: 0x332...b291b
                                  Debug
```

status	true Transaction mined and execution succeed
transaction hash	0x33200943ede27eab3d98648b52f7d57103f0bfb73d2177d68a943256d88b291b
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to	ArrayExample.addNumber(uint256) 0xf8e81D47203A594245E36C48e151709F0C19fBe8
gas	56209 gas

transaction cost	48877 gas
execution cost	27673 gas
input	0xfce...00001
decoded input	{ "uint256 _number": "1" }
decoded output	{}
logs	[]
val	0 wei

```
transact to ArrayExample.addNumber pending ...
[vm]
from: 0x5B3...eddC4
to: ArrayExample.addNumber(uint256) 0xf8e...9fBe8
value: 0 wei
data: 0xfce...00002
logs: 0
hash: 0xc00...a6a04
                                   Debug
```

status	true Transaction mined and execution succeed
transaction hash	0xc008a7189ccf863f524d6248af052eb5e21f98799e5191580edc43ed3eca6a04
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to	ArrayExample.addNumber(uint256) 0xf8e81D47203A594245E36C48e151709F0C19fBe8
gas	56209 gas
transaction cost	48877 gas
execution cost	27673 gas
input	0xfce...00002
decoded input	{ "uint256 _number": "2" }
decoded output	{}
logs	[]
val	0 wei

```
transact to ArrayExample.addNumber pending ...
[vm]
from: 0x5B3...eddC4
to: ArrayExample.addNumber(uint256) 0xf8e...9fBe8
value: 0 wei
data: 0xfce...00003
logs: 0
hash: 0x58f...cacfd
                                   Debug
```

status	true Transaction mined and execution succeed
transaction hash	0x58f77b3c9ff10088a84b4c51d76d0628e38026c6aeb772bc55ccffc732bcacfd
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to	ArrayExample.addNumber(uint256) 0xf8e81D47203A594245E36C48e151709F0C19fBe8
gas	56209 gas
transaction cost	48877 gas
execution cost	27673 gas
input	0xfce...00003
decoded input	{ "uint256 _number": "3" }
decoded output	{}
logs	[]
val	0 wei

```
call to ArrayExample.getSum
```

```
CALL[call]
from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to: ArrayExample.getSum()
data: 0x569...c5f6d
```

Debug

from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to	ArrayExample.getSum() 0xf8e81D47203A594245E36C48e151709F0C19fBe8
execution cost	16268 gas (Cost only applies when called by a contract)
input	0x569...c5f6d
decoded input	{}
decoded output	{ "0": "uint256: 9" }
logs	[]