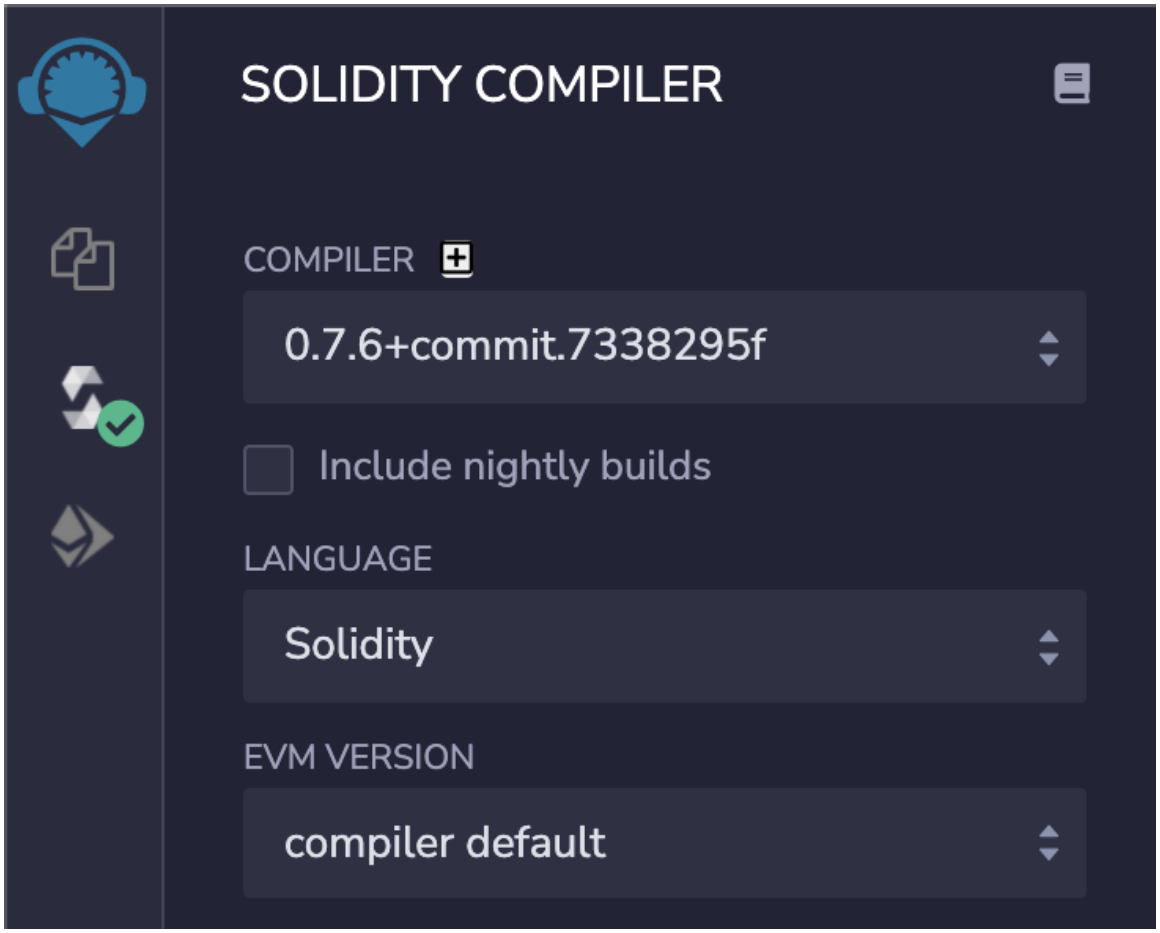
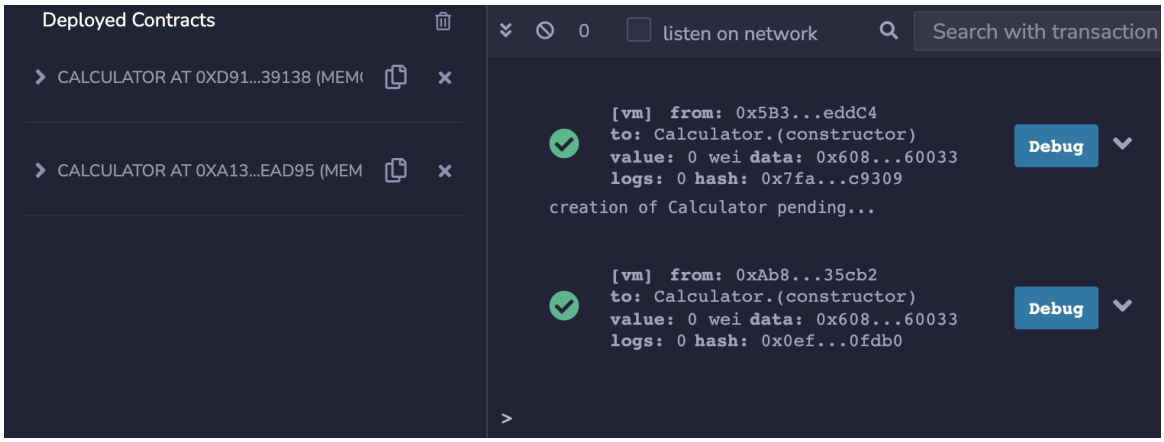
Nicolas Neven
2/6/2022
ITP 356

Homework 2

Remix

1. 

   I changed line 1 to pragma solidity ^0.7.0.

2. 

   I selected a second account to deploy from right below environment selection.

3. 

I expanded the dropdown arrow within the Remix console output for the add() call.

MetaMask

1. 0xBB379331De54A7c0a4b2bfF5A54A14cdba7E9E6d
2. https://kovan.etherscan.io/tx/0xb87822cc233439cd79540e284a35c84cae7b5295d84c2aa8c6298b9903665f94
3. https://kovan.etherscan.io/tx/0x5203df222e4791ee78bc4d849bc1d174b951bd113848a41997b60f80c17062ab
4. 0xD49Fb7b033b54993a70b6F8f133F94eb7A62Ed87
5. https://kovan.etherscan.io/tx/0x79677df0c9adff2728ae879267cf54e13575a727c0727437899144b345e183b7

General Solidity

1. Because it specifies the compiler version to be used for the current Solidity file. Solidity is continuously being developed and upgraded so it is necessary to make sure your code will compile and run with the proper EVM compiler version.
2. It is important to specify having an open-source license (or not) because it tells other users who can view your code if they are allowed to use parts of your code or if they are not allowed giving you legal rights if they attempt to steal your intellectual property.

Variable Types

1.
    a. address user
    b. uint256 balance
    c. bytes32 transcriptHash
    d. int256 temperature
    e. uint256 speedLimit
    f. bool lightSwitch
    g. bytes32 className
    h. address[] degens

2.
    a. 2^2 - 1
    b. 2^4 - 1
    c. 2^128 - 1
    d. (2^4 / 2) - 1
    e. (2^256 / 2) - 1
3. An array is accessed by putting the index of the element you want to access. A mapping is accessed by putting the key of the value you want to access.

Mappings

1. It will return 0. Every possible key is initialized to 0 in a map.
2. Keep track of all keys used in a map in a second array. The size of this array will be how many keys exist within a mapping.
3. Only if you use 2 maps. You cannot create a bidirectional map using only 1 map.
4.
    a. mapping(address => uint) balances
    b. mapping(bytes32 => uint) assets
    c. mapping(address => bool) votes
    d. mapping(address => address) addresses
    e. mapping(uint => bytes32) nfts
    f. mapping(bytes32 => address) contacts
5.
    a. mapping(uint => mapping(uint => bytes32)) transcriptRegistry
    b. mapping(address => mapping(bytes32 => uint)) votingRegistry
    Use 0 to represent no vote, 1 voted no, 2 voted yes.

Functions

1. When we want to create a section of code with a specific purpose that we can reuse multiple times either by users or by the contract itself.
2. An argument (data) that is passed into the function to be used by the function.
3.
    a. public
    b. external
    c. private or internal
4. A function signature defines the name of the function, input parameters, modifiers, and return values. Yes, you can have two functions with the same name if they have different signatures.
5. Then you can add a return value to the function.
6. Function modifiers are used to check conditions. They are used when you need to check the same conditions multiple times.

7. When an assert() is false it uses up all the remaining gas while require() refunds all the remaining gas. Both reverts all changes made (when false).
8. _; means you've reached the end of this modifier without reverts!
9.
    a. pure
    b. view
    c. neither
    d. pure