

SOLIDITY PROGRAMMING

Karachi Institute of Technology and Entrepreneurship (KITE)

Session 2: Solidity (Functions and Contracts)

Date: 6th November 2021

Instructor: Syed Faisal ur Rahman

Readings

<https://www.tutorialspoint.com/solidity/index.htm>

<https://www.dappuniversity.com/articles/solidity-tutorial>

<https://101blockchains.com/solidity-tutorial/>

<https://www.geeksforgeeks.org/introduction-to-solidity/?ref=leftbar-rightbar>

<https://betterprogramming.pub/learn-solidity-functions-ddd8ea24c00d>

<https://www.bitdegree.org/learn/solidity-types>

https://www.tutorialspoint.com/solidity/solidity_mappings.htm

<https://medium.com/coinmonks/what-the-hack-is-memory-and-storage-in-solidity-6b9e62577305>

<https://www.ops.gov.ie/app/uploads/2021/01/Blockchain-Develop-Deploy-and-Test-Your-First-Smart-Contract.pdf>

https://ethereumbuilders.gitbooks.io/guide/content/en/solidity_tutorials.html

DEVELOPMENT ENVIRONMENT

<https://remix.ethereum.org/>

READING: VIEW AND PURE FUNCTION

<https://www.geeksforgeeks.org/solidity-view-and-pure-functions/?ref=leftbar-rightbar>

TASK 1: PURE FUNCTION

Ref: <https://www.geeksforgeeks.org/solidity-view-and-pure-functions/?ref=leftbar-rightbar>

```
// Solidity program to demonstrate pure functions  
pragma solidity ^0.5.0;
```

```
// Defining a contract
```

```

contract Test {

    // Defining pure function to calculate product and sum of numbers
    function getResult(
    ) public pure returns(
        uint product, uint sum){
        uint num1 = 2;
        uint num2 = 4;
        product = num1 * num2;
        sum = num1 + num2;
    }
}

```

PURE FUNCTION: INCORRECT USE

Ref: <https://www.geeksforgeeks.org/solidity-view-and-pure-functions/?ref=leftbar-rightbar>

```

pragma solidity ^0.5.0;

// Defining a contract

contract Test {

    //state variable

    uint test=1;

    // Defining pure function to calculate product and sum of numbers

    function getResult(

    ) public pure returns(

        uint product, uint sum){

        uint num1 = 2;

        uint num2 = 4;

        product = num1 * num2;

        sum = num1 + num2 +test ;

    }

}

```

TASK 2: VIEW FUNCTION

If we want to access values from the environment or state variables then view function is needed.

Ref: <https://www.geeksforgeeks.org/solidity-view-and-pure-functions/?ref=leftbar-rightbar>

```

pragma solidity ^0.5.0;

// Defining a contract
contract Test {

    uint test =1;

    // Defining view function to calculate product and sum of numbers
    function getResult(

    ) public view returns(

    uint product, uint sum){

        uint num1 = 2;

        uint num2 = 4;

        product = num1 * num2;

        sum = num1 + num2 + test ;

    }

}

```

READING: PAYABLE FUNCTION

<https://rangesh.medium.com/6-payable-functions-in-solidity-smartcontract-ethereum-d2535e346dc1>

TASK 3: PAYABLE FUNCTION

//ref: <https://rangesh.medium.com/6-payable-functions-in-solidity-smartcontract-ethereum-d2535e346dc1>

```

pragma solidity ^0.4.4;

contract Sample {

    uint amount =1;

    function payme() payable{

        amount += msg.value;

    }

}

```

TASK 4: SOLIDITY CONTRACT TO FIND OWNER ADDRESS AND BALANCE

//ref: <https://www.geeksforgeeks.org/creating-a-smart-contract-that-returns-address-and-balance-of-owner-using-solidity/>

// Solidity program to retrieve address and balance of owner

pragma solidity ^0.6.8;

// Creating a contract

contract MyContract

{

 // Private state variable

 address private owner;

 // Defining a constructor

 constructor() public{

 owner=msg.sender;

 }

 // Function to get address of owner

 function getOwner(

) public view returns (address) {

 return owner;

 }

 // Function to return current balance of owner

 function getBalance(

) public view returns(uint256){

 return owner.balance;

 }

}

https://www.tutorialspoint.com/solidity/solidity_contracts.htm

READING: FUNCTION OVERLOADING AND OVERRIDING

https://www.tutorialspoint.com/solidity/solidity_function_overloading.htm

<https://medium.com/upstate-interactive/solidity-override-vs-virtual-functions-c0a5dfb83aaf>

READING: INHERITANCE, ABSTRACT, INTERFACE

https://www.tutorialspoint.com/solidity/solidity_inheritance.htm

<https://solidity-by-example.org/super/>

<https://solidity-by-example.org/inheritance/>

<https://solidity-by-example.org/shadowing-inherited-state-variables/>

<https://www.bitdegree.org/learn/solidity-inheritance>

<https://www.geeksforgeeks.org/solidity-inheritance/?ref=leftbar-rightbar>

https://www.tutorialspoint.com/solidity/solidity_abstract_contracts.htm

<https://www.geeksforgeeks.org/solidity-abstract-contract/?ref=lbp>

<https://www.geeksforgeeks.org/solidity-basics-of-interface/?ref=leftbar-rightbar>

https://www.tutorialspoint.com/solidity/solidity_interfaces.htm

<https://solidity-by-example.org/interface/>

TASK 5: INHERITANCE AND FUNCTION OVERRIDING

//ref: <https://solidity-by-example.org/inheritance/>

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.3;

/* Graph of inheritance

A

/\

B C

```
/\//
```

```
F D,E
```

```
*/
```

```
contract A {
```

```
    function foo() public pure virtual returns (string memory) {
```

```
        return "A";
```

```
    }
```

```
}
```

```
// Contracts inherit other contracts by using the keyword 'is'.
```

```
contract B is A {
```

```
    // Override A.foo()
```

```
    function foo() public pure virtual override returns (string memory) {
```

```
        return "B";
```

```
    }
```

```
}
```

```
contract C is A {
```

```
    // Override A.foo()
```

```
    function foo() public pure virtual override returns (string memory) {
```

```
        return "C";
```

```
    }
```

```
}
```

```
// Contracts can inherit from multiple parent contracts.
```

```
// When a function is called that is defined multiple times in
```

```
// different contracts, parent contracts are searched from
```

```
// right to left, and in depth-first manner.
```

```

contract D is B, C {

    // D.foo() returns "C"

    // since C is the right most parent contract with function foo()

    function foo() public pure override(B, C) returns (string memory) {

        return super.foo();

    }

}

```

```

contract E is C, B {

    // E.foo() returns "B"

    // since B is the right most parent contract with function foo()

    function foo() public pure override(C, B) returns (string memory) {

        return super.foo();

    }

}

```

// Inheritance must be ordered from “most base-like” to “most derived”.

// Swapping the order of A and B will throw a compilation error.

```

contract F is A, B {

    function foo() public pure override(A, B) returns (string memory) {

        return super.foo();

    }

}

```

TASK 6: SHADOWING INHERITED STATE VARIABLES

// ref: <https://solidity-by-example.org/shadowing-inherited-state-variables/>

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.3;

```

contract A {

    string public name = "Contract A";


    function getName() public view returns (string memory) {

        return name;

    }

}

// Shadowing is disallowed in Solidity 0.6
// This will not compile
// contract B is A {
//     string public name = "Contract B";
// }

contract C is A {

    // This is the correct way to override inherited state variables.

    constructor() {

        name = "Contract C";

    }

    // C.getName returns "Contract C"

}

```

TASK 7: INTERFACE

//ref: https://www.tutorialspoint.com/solidity/solidity_interfaces.htm

```
pragma solidity ^0.5.0;
```

```

interface Calculator {

    function getResult() external view returns(uint);

}

contract Test is Calculator {

```



```
constructor() public {}  
  
function getResult() external view returns(uint){  
  
    uint a = 1;  
  
    uint b = 2;  
  
    uint result = a + b;  
  
    return result;  
  
}  
}
```

READING: METAMASK, RINKEBY TESTNET AND CONTRACT DEPLOYMENT

<https://metamask.io/>

<https://remix-ide.readthedocs.io/en/latest/run.html>

<https://rinkeby.etherscan.io/>

<https://medium.com/compound-finance/the-beginners-guide-to-using-an-ethereum-test-network-95bbbc85fc1d>

TASK 8: DEPLOY A CONTRACT ON RINKEBY TESTNET

- 1- Create an account using Metamask using its Google chrome extension: <https://metamask.io/>
- 2- Login to your Metamask account and select Rinkeby testnet.
- 3- Open remix in the same browser which has your Metamask account logged in.
- 4- Write a sample contract.
- 5- Compile it as usual.
- 6- In the “Deploy & run” part, instead of using Javascript VM, use: Injected Web3.
- 7- This will deploy your contract on Rinkeby testnet.

TASK 9: EXPLORE THE RINKEBY TESTNET:

Got to: <https://rinkeby.etherscan.io/>

Search: 0x10b9E144DE89B118a27fdf16577888CdC59f6ad5

This is just a test contract which I deployed.

This will also show my public address: 0x76C62EB54bEc8132Cff0bCBBB3aA1022f33Fb9AC

