

Exercise 5 - Polymorphism in solidity

Implementation of polymorphism and its types in solidity using Remix IDE

Name : Naman

Reg No: 20BKT0046

Q1) Implementation of function polymorphism

Code:

```
//SPDX-License-Identifier: UNLICENSED
```

```
pragma solidity >=0.5.0 < 0.9.0;
```

```
contract methodOverloading {
```

```
// Function to get value of the string variable
```

```
    function getValue(string memory _strin) public pure returns(string memory)
```

```
    {
```

```
        return _strin;
```

```
    }
```

```
// function to get value of the unsigned integer variable
```

```
    function getValue(uint _num) public pure returns(uint)
```

```
    {
```

```
        return _num;
```

```
    }
```

```
}
```

```

1 //SPDX-License-Identifier: UNLICENSED
2 pragma solidity >=0.5.0 < 0.9.0;
3
4 contract methodOverloading {
5     // Function to get value of the string variable
6     function getValue(string memory _strin) public pure returns(string memory)
7     {
8         return _strin;
9     }
10    // function to get value of the unsigned integer variable
11    function getValue(uint _num) public pure returns(uint)
12    {
13        return _num;
14    }
15 }

```

Transactions recorded: 1

Run transactions using the latest compilation result

Save Run

Deployed Contracts

METHODOVERLOADING AT 0X9D: [icon] x

Balance: 0 ETH

getValue

_num: 5

Calldata Parameters call

getValue

_strin: NAMAN

Calldata Parameters call

Transaction details:

- execution cost: 23039 gas (Cost only applies when called by a contract)
- input: 0x960...00000
- decoded input: {"string_strin": "NAMAN"}
- decoded output: {"0": "string: NAMAN"}
- logs: []

```

2 pragma solidity >=0.5.0 < 0.9.0;
3
4 contract methodOverloading {
5     // Function to get value of the string variable
6     function getValue(string memory _strin) public pure returns(string memory)
7     {
8         return _strin;
9     }
10    // function to get value of the unsigned integer variable
11    function getValue(uint _num) public pure returns(uint)
12    {
13        return _num;
14    }
15 }

```

Transactions recorded: 2

Run transactions using the latest compilation result

Save Run

Deployed Contracts

METHODOVERLOADING AT 0X9D: [icon] x

Balance: 0 ETH

getValue

_num: 5

Calldata Parameters call

getValue

0: uint256: 5

_strin: NAMAN

Calldata Parameters call

Transaction details:

- CALL: [call] from: 0x5B38Da6a701c568545dCfC803FcB875f56beddC4 to: methodOverloading.getValue(uint256) data: 0x0ff...00005
- from: 0x5B38Da6a701c568545dCfC803FcB875f56beddC4
- to: methodOverloading.getValue(uint256) 0x907F74d0C41E726EC95884E8e97Fa6129e3b5E99
- execution cost: 21784 gas (Cost only applies when called by a contract)
- input: 0x0ff...00005
- decoded input: {"uint256 _num": "5"}
- decoded output: {"0": "uint: 5"}

Q2) Implementation of contract polymorphism

Code:

```
//SPDX-License-Identifier: UNLICENSED
```

```
pragma solidity >=0.5.0 < 0.9.0;
```

```
// Contract definition
```

```
contract parent
```

```
{
```

```
    // Internal state variable
```

```
    uint internal sum;
```

```

// Function to set the value of internal state variable sum
function setValue(uint _num1, uint _num2) public
{
    sum = _num1 + _num2;
}

// Function to return a value 10
function getValue() public view returns(uint)
{
    return sum;
}
}

// Defining child contract
contract child is parent
{
    // Function getValue overloaded to return internal state variable sum defined in the
    // parent contract
    function getValue() public view returns(uint)
    {
        return sum;
    }
}

// Defining calling contract
contract ContractPolymorphism
{
    // Creating object
    parent pc = new child();

    // Function to set values of 2 unsigned integers
    function getInput(uint _num1, uint _num2) public

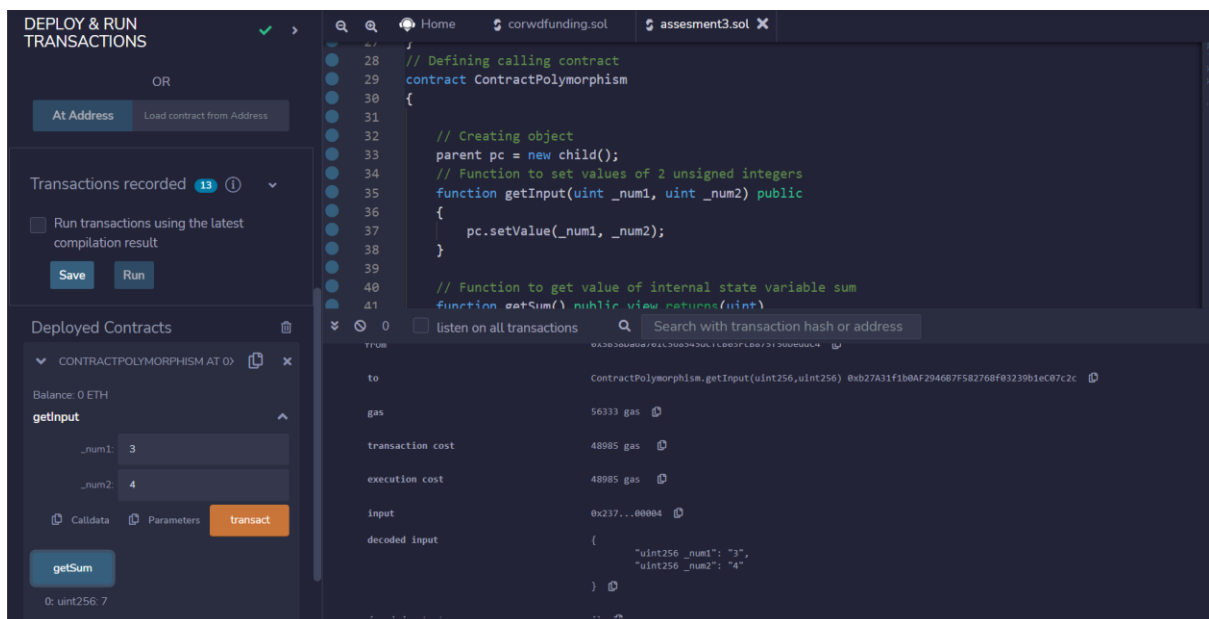
```

```

{
    pc.setValue(_num1, _num2);
}

// Function to get value of internal state variable sum
function getSum() public view returns(uint)
{
    return pc.getValue();
}
}

```



3) Implementation of abstract contract in solidity

Code:

```
// SPDX-License-Identifier: GPL-3.0
```

```
pragma solidity >=0.7.0 <0.9.0;
```

```
abstract contract AbstractHelloWorld
```

```
{
```

```

//function declaration without definition in abstract contract
function GetValue() virtual public view returns (uint);
function SetValue(uint _value) virtual public;
function AddNumber(uint _value) virtual public returns(uint)
{
    return _value;
}
}
contract HelloWorld is AbstractHelloWorld
{
    uint private simpleInteger;
    //function definition
    function GetValue() override public view returns (uint)
    {
        return simpleInteger;
    }
    function SetValue(uint _value) override public
    {
        simpleInteger = _value;
    }
    function AddNumber(uint _value) override public view returns (uint)
    {
        return (simpleInteger + _value);
    }
}
contract Client
{
    AbstractHelloWorld myObj= new HelloWorld();

```

```

function GetIntegerValue() public returns (uint)
{
    myObj.SetValue(100);
    return myObj.AddNumber(200) + 10;
}
}

```

Output:

Client:

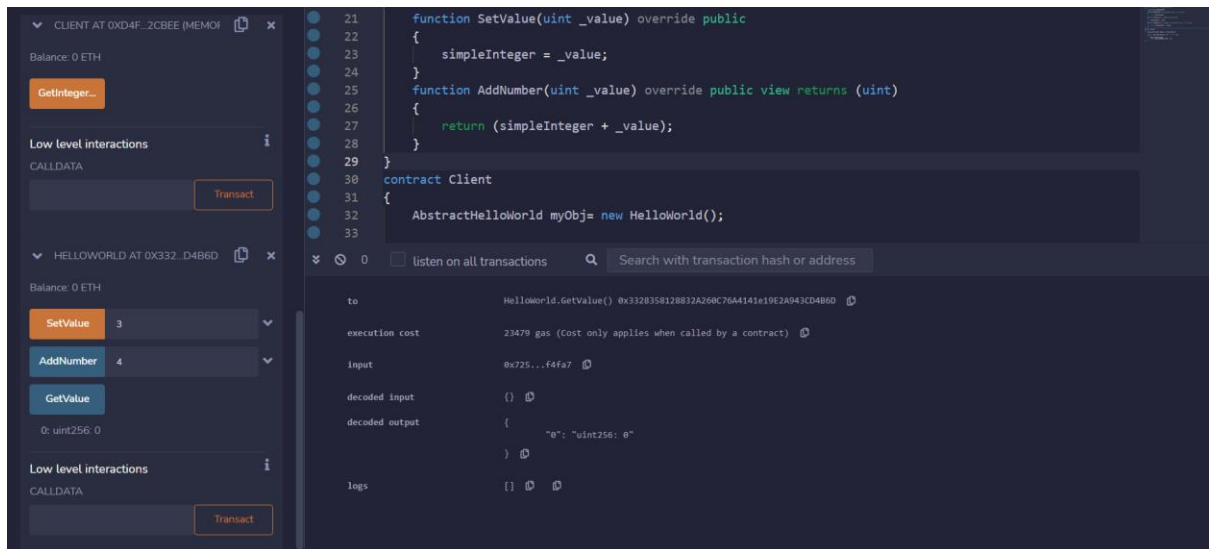
The screenshot displays the Remix IDE interface with the following components:

- Left Panel:**
 - Transactions recorded:** 18
 - Run transactions using the latest compilation result:** (checked)
 - Save** and **Run** buttons.
 - Deployed Contracts:**
 - CLIENT AT 0XD4F...2CBEE (MEMO):**
 - Balance: 0 ETH
 - GetInteger...** button
 - Low level interactions:**
 - CALLDATA:** (input field)
 - Transact** button
- Center Panel:** Solidity code editor showing the `Client` contract:


```

contract Client
{
    AbstractHelloWorld myObj= new HelloWorld();
}
      
```
- Right Panel:** Transaction execution details for `0x1f6...fe241`:
 - execution cost:** 31260 gas
 - input:** `0x1f6...fe241`
 - decoded input:** `()`
 - decoded output:** `{ "0": "uint256: 310" }`
 - logs:** `[]`
 - val:** `0 wei`

HelloWorld



Q4) Implementation of the interface in solidity

Code:

```
// SPDX-License-Identifier: GPL-3.0
```

```
pragma solidity >=0.7.0 <0.9.0;
```

```
interface IHelloWorld
```

```
{
```

```
    function GetValue() external view returns (uint);
```

```
    function SetValue(uint _value) external;
```

```
}
```

```
contract HelloWorld is IHelloWorld
```

```
{
```

```
    uint private simpleInteger;
```

```
    function GetValue() public view returns (uint)
```

```
    {
```

```
        return simpleInteger;
```

```
    }
```

```
    function SetValue(uint _value) public
```

```
    {
```

```
        simpleInteger = _value;
```

```

    }
}

contract Client
{
    function GetSetIntegerValue() public returns (uint)
    {
        IHelloWorld myObj = new HelloWorld();
        myObj.SetValue(100);
        return myObj.GetValue() + 10;
    }
}

```

Output:

The screenshot displays a web-based Ethereum development environment. On the left, the 'TRANSACTIONS' panel shows 'Transactions recorded: 20'. Below it, the 'Deployed Contracts' section lists a contract named 'HELLOWORLD AT 0X5E1...4EFF5'. The contract's balance is 0 ETH. The 'Set Value' field is set to 4, and the 'Add Number' field is set to 8. The 'Get Value' button is highlighted. The 'Low level interactions' section shows the 'CALLDATA' field with a 'Transact' button.

The main editor displays the Solidity code for the 'HelloWorld' contract:

```

contract HelloWorld is AbstractHelloWorld
{
    uint private simpleInteger;
    //function definition
    function GetValue() override public view returns (uint)
    {
        return simpleInteger;
    }
    function SetValue(uint _value) override public
    {
        simpleInteger = _value;
    }
}

```

The bottom panel shows the execution details of a transaction. The transaction is a 'call' from address '0x58380a6a701c568545dcfc803fc8875f56bedd0c4' to the contract address '0x5e17b14ad06c386305a32928f985b298ba34eff5'. The execution cost is 23479 gas. The input is '0x725...fafa7'. The decoded output is a JSON object: `{ "0": "uint256: 0" }`. The logs section is empty.