**[Exercise 5 - Polymorphism in solidity](https://moovit.vit.ac.in/course/view.php?id=12088" \l "section-6)**

**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

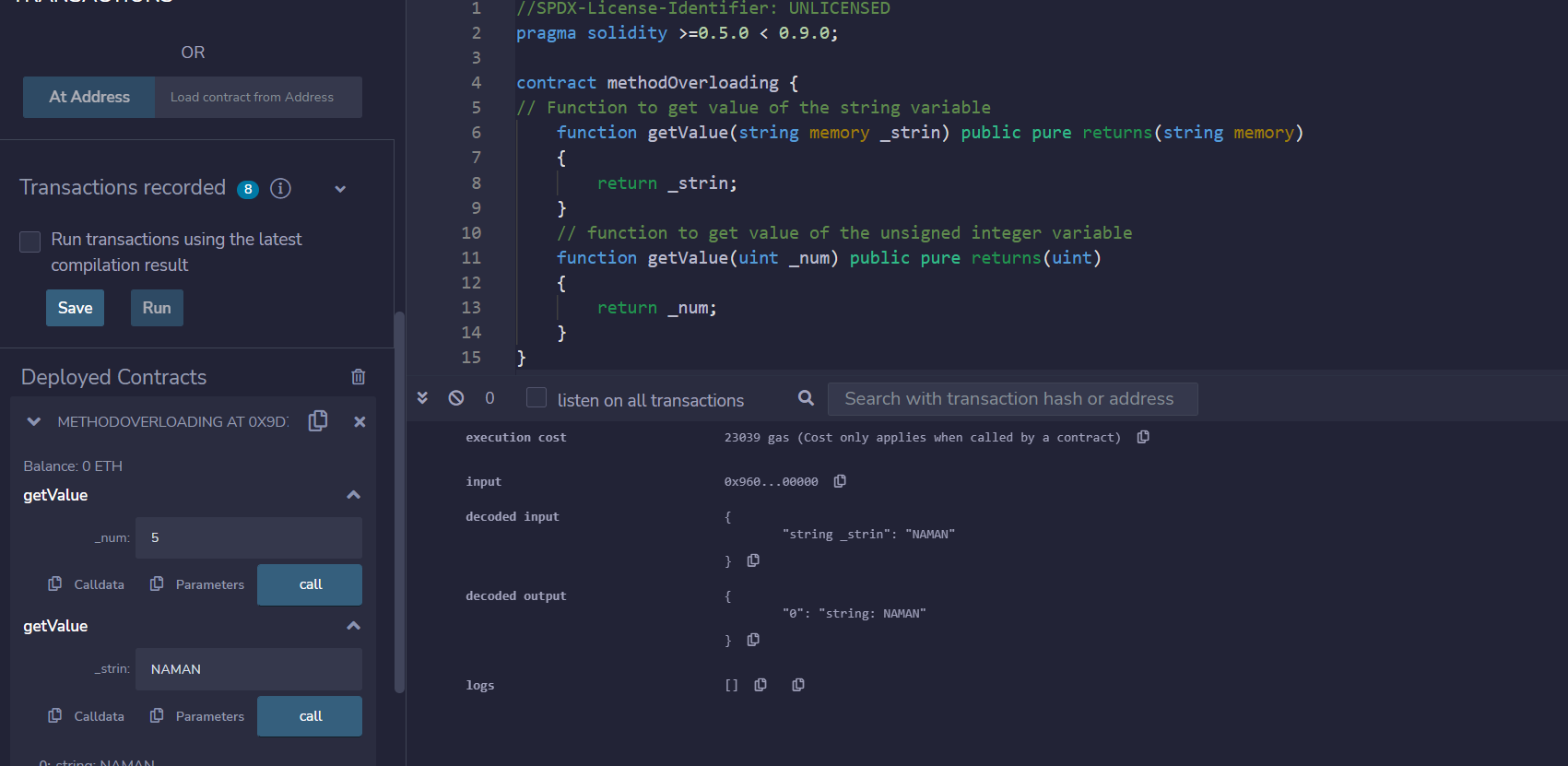
    function getValue(uint \_num) public pure returns(uint)

    {

        return \_num;

    }

}



A screenshot of a computer

Description automatically generated with medium confidence

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

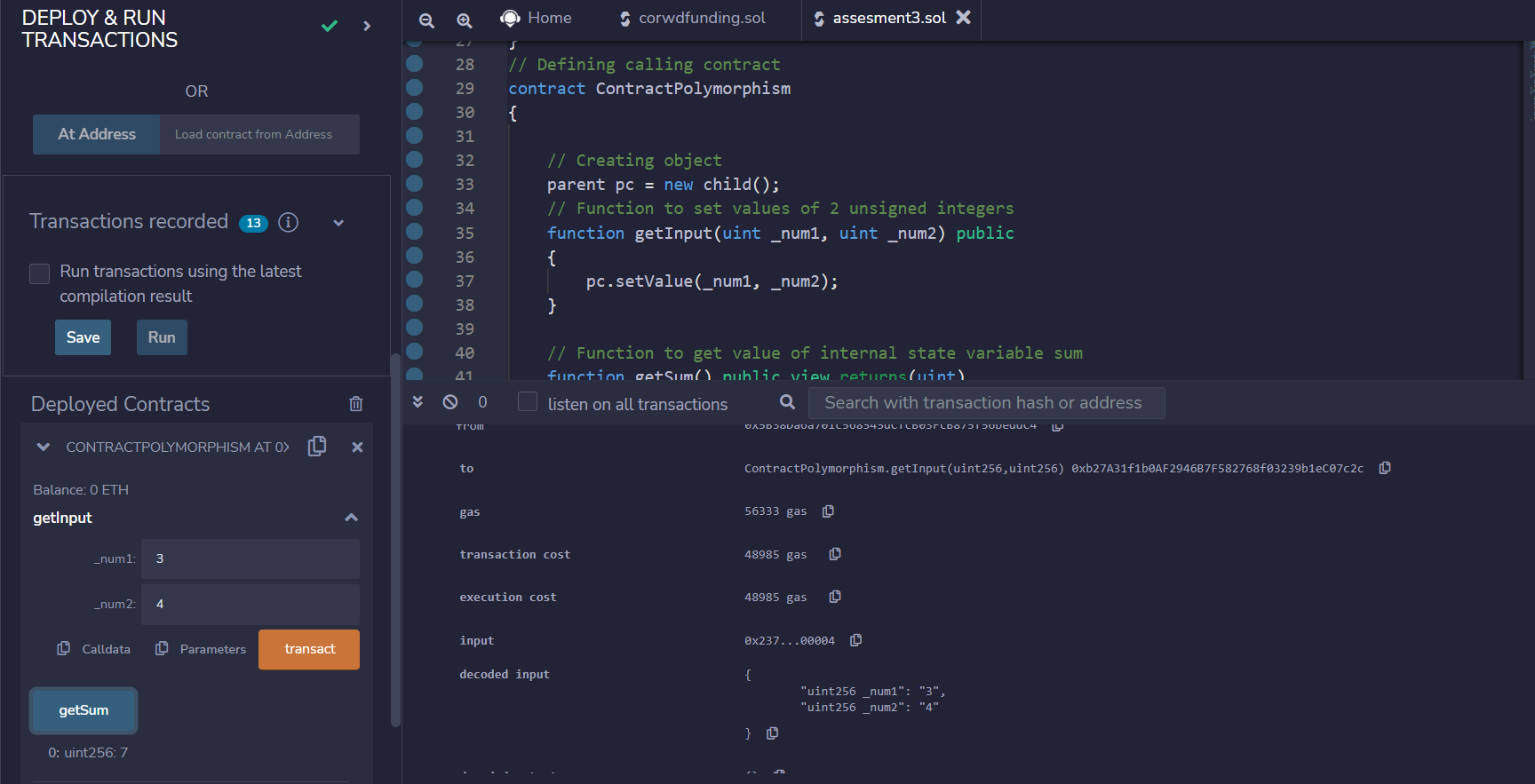
    function getSum() public view returns(uint)

    {

        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);

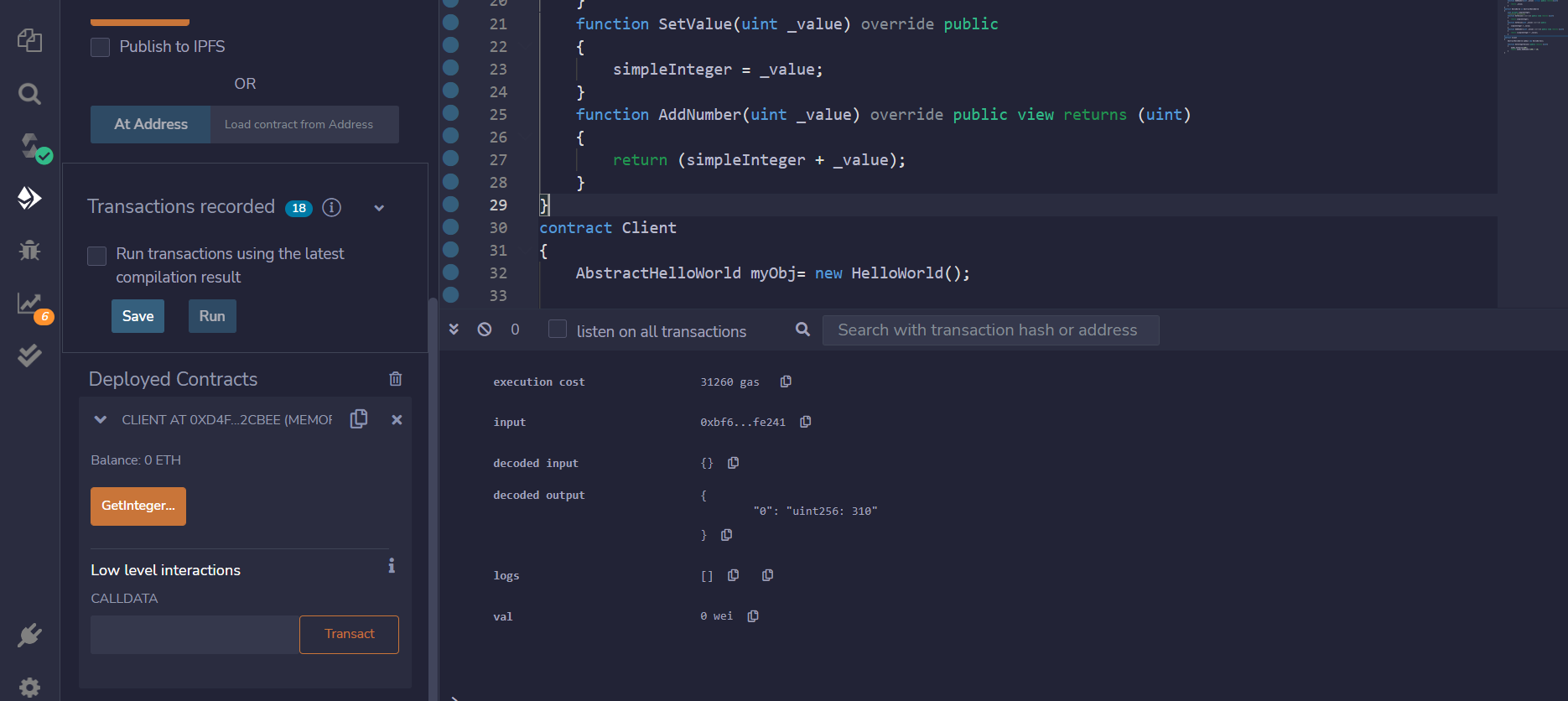
        return myObj.AddNumber(200) + 10;

    }

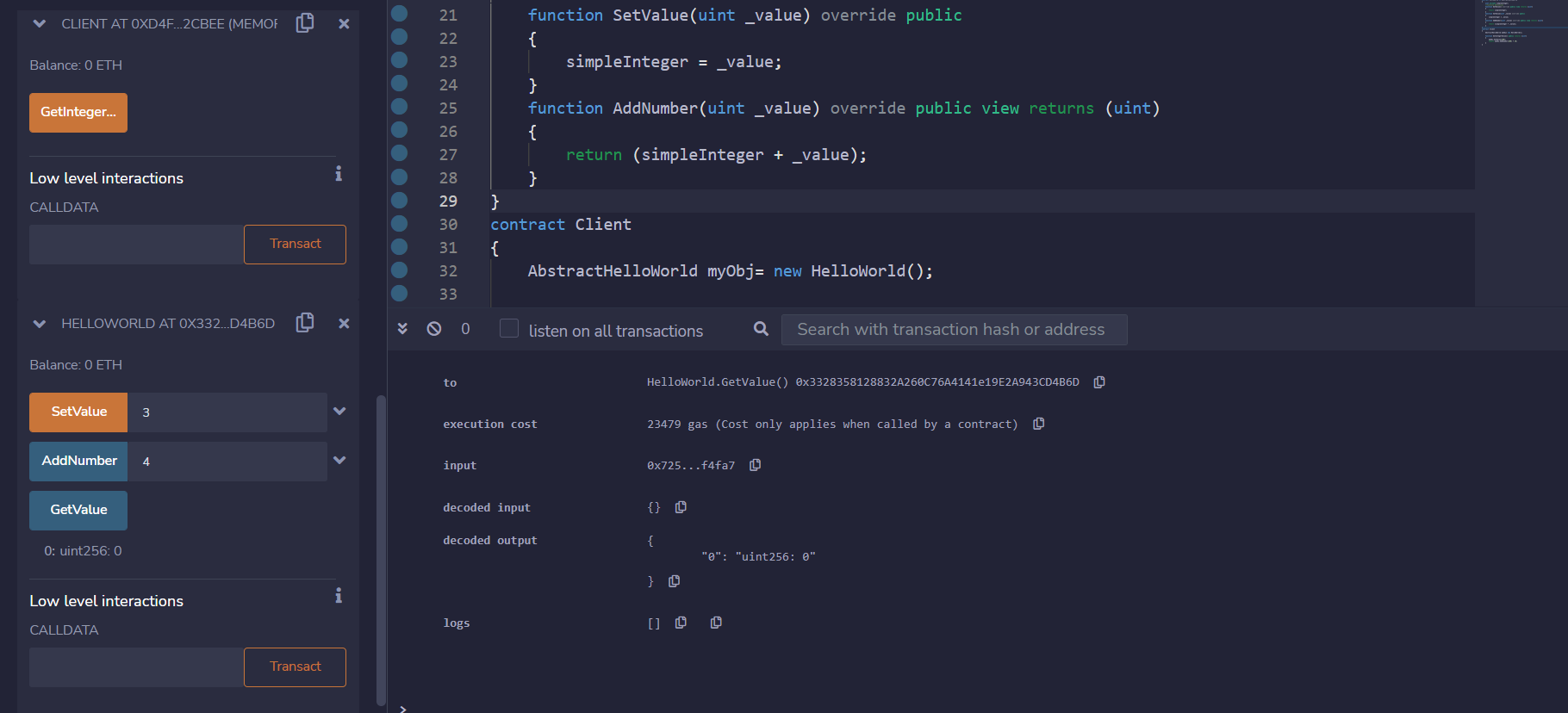
}

Output:

Client:



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:

