**Blockchain Technology**

**Basic Solidity Smart Contracts – 1**

**Aim:** Basic Solidity Smart contracts-1

1. Find the given number is negative or positive.
2. Find the given is even or odd.
3. Find the factorial of given number using “for” loop.
4. Find the factorial of given number using “while” loop.
5. Find the given number is prime or not.

**Description**:

1. Solidity is an object-oriented programming language for implementing smart contracts on various blockchain platforms, most notably, Ethereum.
2. Remix ide is used to write and deploy the smart contract.

**Code:**

1. Find the given number is negative or positive.

*//SPDX-License-Identifier: UNLICENSED*

pragma solidity ^0.8.0;

contract PositiveNegative{

    int private num;

    function setNumber(int \_num) public{

        num = \_num;

    }

    function PosNeg() public view returns (string memory res){

        if (num < 0){

            res = "This number is negative";

        }

        else{

            res = "This number is positive";

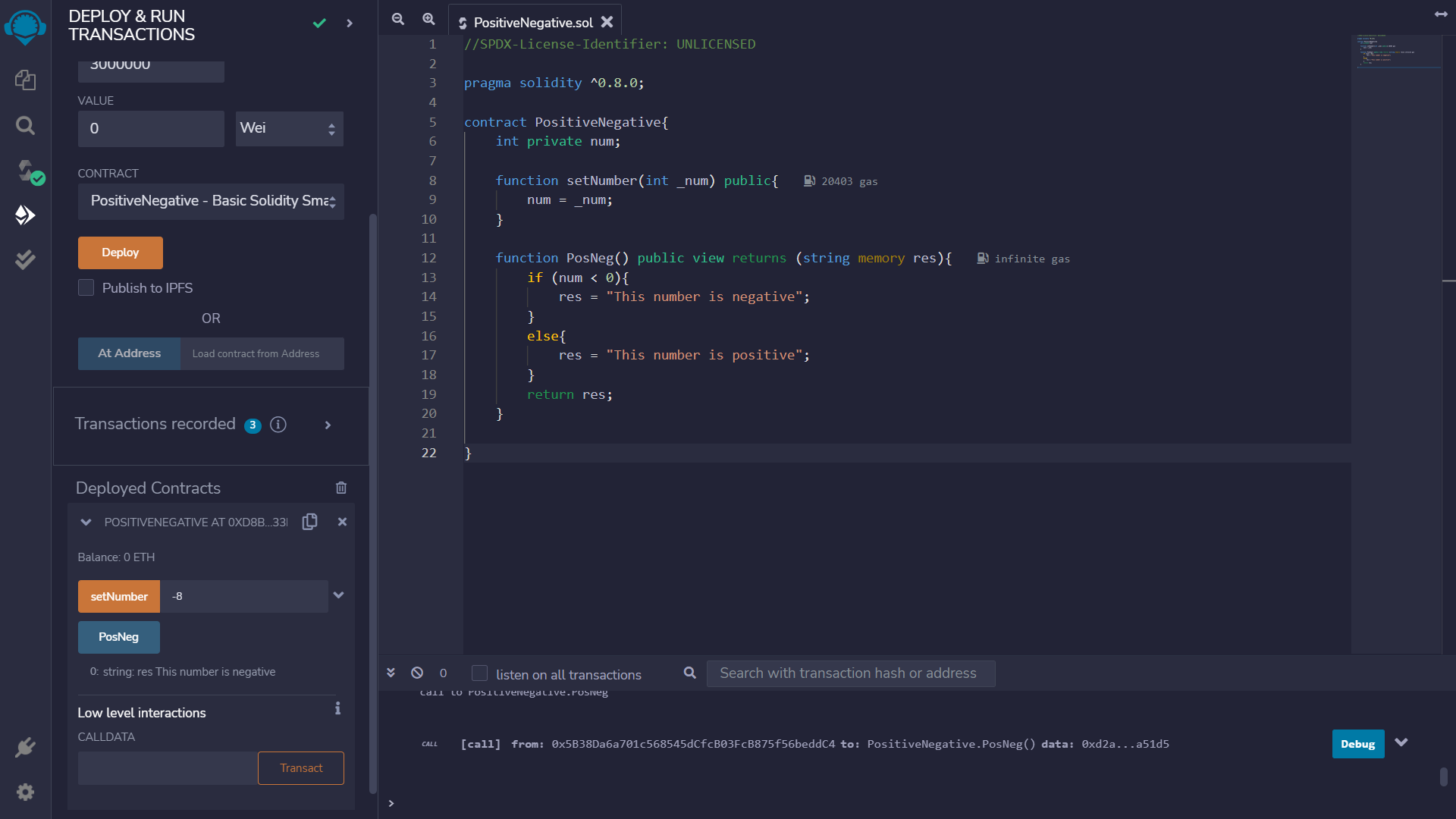
        }

        return res;

    }

}

Output:



1. Find the given number is odd or even.

*//SPDX-License-Identifier: UNLICENSED*

pragma solidity ^0.8.0;

contract EvenOdd{

    int private num;

    function setNumber(int \_num) public{

        num = \_num;

    }

    function Even\_Odd() public view returns (string memory res){

        if (num%2 == 0){

            res = "This number is even";

        }

        else{

            res = "This number is odd";

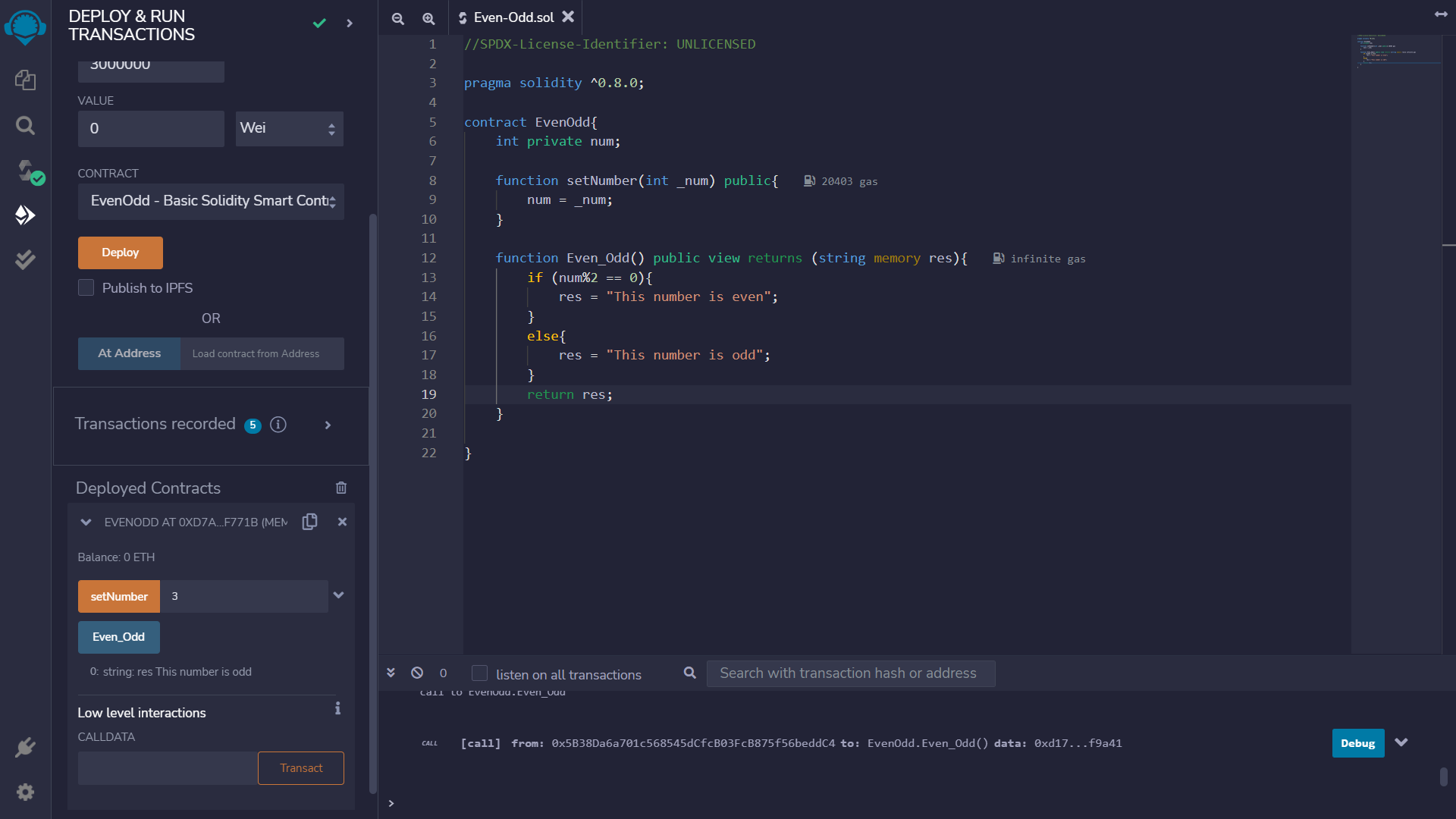
        }

        return res;

    }

}

Output:



1. Find the factorial of given number using “for” loop.

*//SPDX-License-Identifier: UNLICENSED*

pragma solidity ^0.8.0;

contract FactorialUsingForLoop{

*/\*int private num;*

*function setNumber(int \_num) public{*

*num = \_num;*

*}\*/*

    function factorial(int num) pure public returns (int res){

        if (num == 0){

            res = 1;

        }

        else{

            int facto = 1;

            for(int i = 1; i <= num; i++){

                facto = facto\*i;

            }

            res = facto;

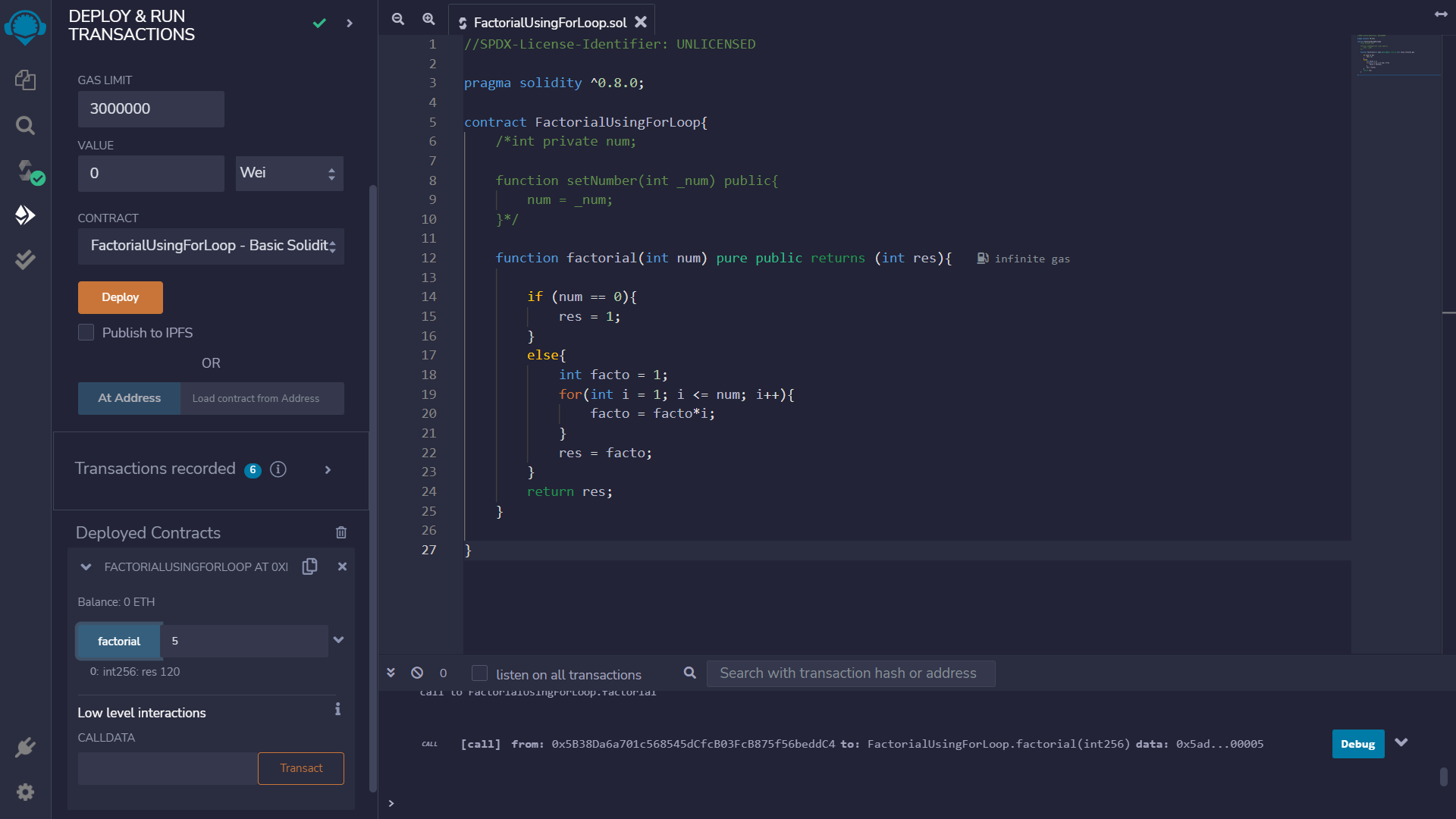
        }

        return res;

    }

}

Output:



1. Find the factorial of given number using “while” loop.

*// SPDX-License-Identifier: MIT*

pragma solidity ^0.8.0;

contract  fac {

    function factorial(uint num) pure public returns(uint){

        uint fact = 1;

        uint i =1;

        while(i <= num){

            fact = fact \* i;

            i++;

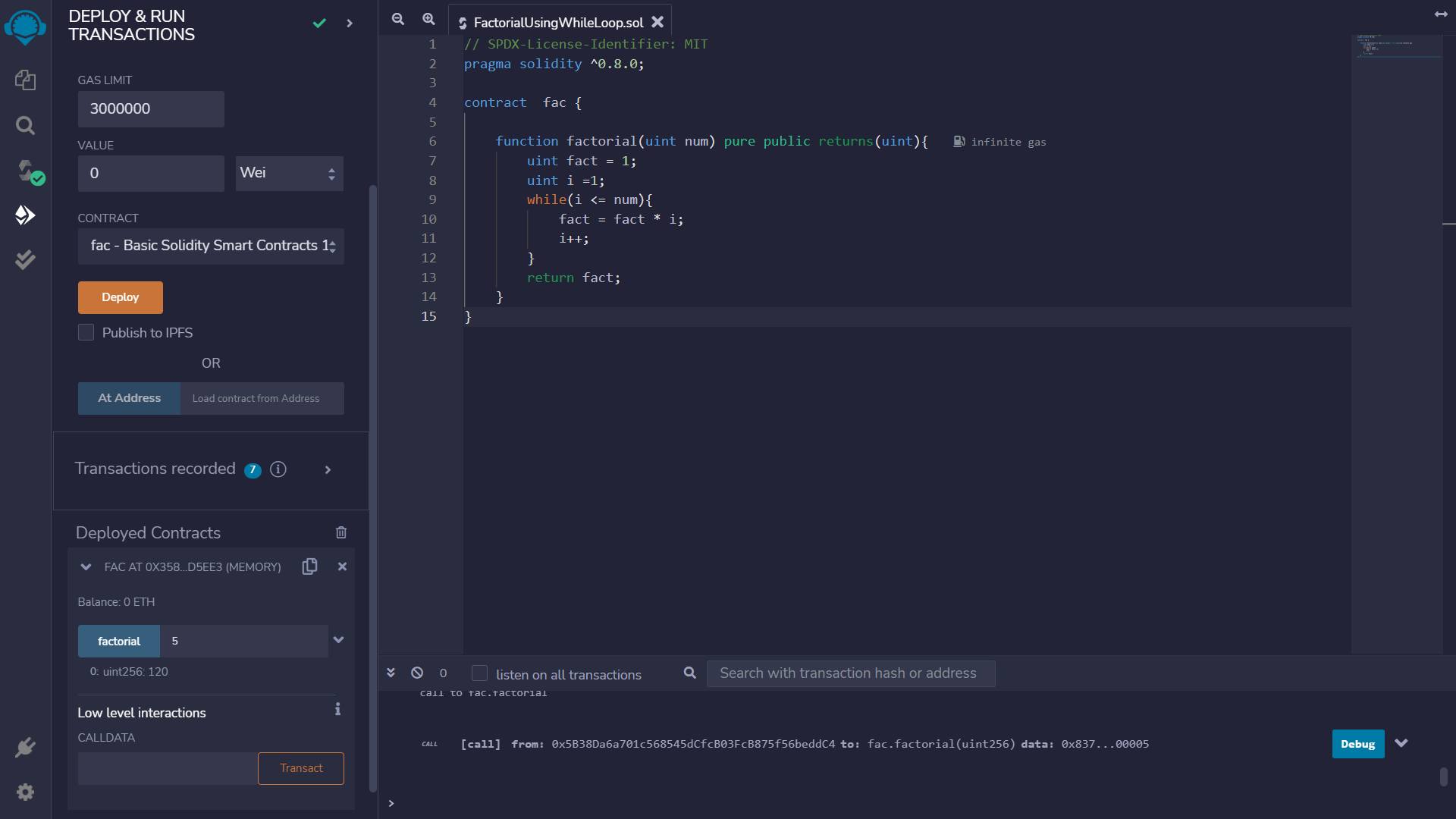
        }

        return fact;

    }

}

Output:



1. Find the given number is prime or not.

*// SPDX-License-Identifier: MIT*

pragma solidity ^0.8.0;

contract PrimeNumberChecker {

    function isPrime(uint256 n) public pure returns (string memory) {

        for (uint256 i = 2; i < n; i++) {

            if (n % i == 0) {

                return "Not a prime";

            }

        }

        return "prime";

    }

}

Output:

