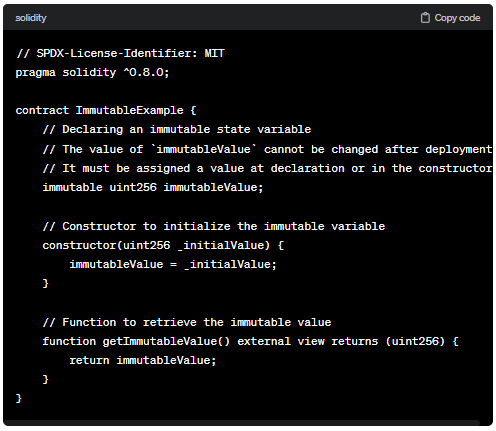
**Immutability:**

In Solidity, the **immutable** keyword is used to declare state variables whose values cannot be changed after they are initialized. These variables are similar to constants but are initialized at deployment time rather than at compile time.

Once assigned a value in the constructor or at declaration (for state variables), the value of an **immutable** variable cannot be modified throughout the lifetime of the contract. This immutability guarantees that the value remains constant and cannot be altered by any means, providing security and predictability to the contract's behavior.



In case of constant state variable -> It is inline (means we need to initialize at the time of declaration).

To over come this problem where we can make our state variable and initialize its value at the time of deployment.

In case of immutability we can initialize inline and also through constructor.

Here the cost ->

1) Simple (23534)

2) Immutable (21526)

3) Constant (21472)

**Code:**

//SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.8.0;

contract Immutable{

    address public immutable test;

    address public constant test2 = address(1);

    address public test3 = address(1);

    constructor(address \_test){

        test = \_test;

    }

    function I() public view returns(address){

        return test;

    }

    function C() public pure returns(address){

        return test2;

    }

    function S() public view returns(address){

        return test3;

    }

}

**Output:**

