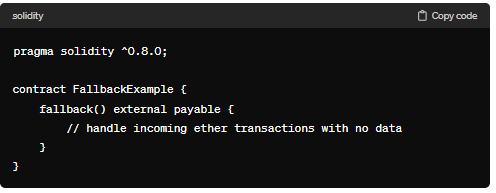
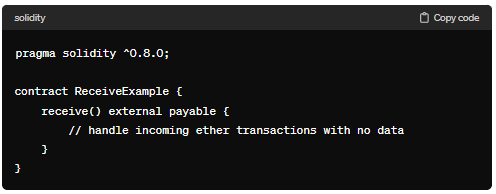
**Fallback and Receive:**

In Solidity, "fallback" and "receive" are special functions related to the handling of ether transactions in smart contracts.

**Fallback Function**: In earlier versions of Solidity, a function marked as **fallback** is executed when a contract receives a message that does not match any of its function signatures, or when a contract receives ether without any accompanying data. It's essentially a catch-all function.



**Receive Function**: Starting from Solidity version 0.6.0, a new feature called the "receive" function was introduced. This function is similar to the fallback function, but it is only triggered when the contract receives ether without any data. If a contract has a receive function, it cannot have a fallback function.



It's important to note that in the latest versions of Solidity (from 0.8.0), using the **receive** and **fallback** functions directly is discouraged, and instead, you should use explicit function definitions. For example, you might want to use **receive()** for receiving ether and have separate functions for other functionality.

Here whatever the task performed by the “receiver” function can be handled by the “fallback” function also. But in reverse whatever the function handled by the “fallback” function cannot be handled by by “Receive” function.

**Fallback:**

1. It is executed when a non-existent function is called on the contract.
2. It is required to be marked external.
3. It has no name.
4. It has no arguments
5. It cannot return anything
6. It can be defined one per contract.
7. If not marked payable, it will throw exception if contract receives ether.
8. It’s main use is to directly send the ETH to contract.

**Code:**

//SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.8.0;

contract fall\_receive{

    event test(string \_msg,address \_sender, uint \_content, bytes \_information);

   fallback() external payable {

        emit test("fallback",msg.sender, msg.value,msg.data);

    }

    receive() external payable{

        emit test("receiver", msg.sender,msg.value, "");

    }

    function checkBal() public view returns(uint) {

        return address(this).balance;

    }

}

**Output:**

