**1: Regular Functions**

✅ Code

Create a file named: Addition.sol

Paste this:

solidity

Copy

Edit

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.17;

contract Addition {

int public input1;

int public input2;

function setInputs(int \_input1, int \_input2) public {

input1 = \_input1;

input2 = \_input2;

}

function additions() public view returns(int) {

return input1 + input2;

}

function subtract() public view returns(int) {

return input1 - input2;

}

}

✅ Compile

In the Solidity Compiler tab, select 0.8.17.

Click Compile Addition.sol

✅ Deploy

In Deploy & Run Transactions:

Environment: Remix VM (Prague)

Contract: Addition

Click Deploy.

✅ Test

In Deployed Contracts:

Call setInputs(5,3)

Call additions() → should return 8.

Call subtract() → should return 2.

🎉 Done with Regular Functions!

**✨ Step 2: View Functions**

✅ Code

Create: ViewDemo.sol

Paste:

solidity

Copy

Edit

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract view\_demo {

uint256 num1 = 2;

uint256 num2 = 4;

function getResult() public view returns (uint256 product, uint256 sum) {

product = num1 \* num2;

sum = num1 + num2;

}

}

✅ Compile

Compile ViewDemo.sol.

✅ Deploy

Deploy the view\_demo contract.

✅ Test

Click getResult():

Copy

Edit

product: 8

sum: 6

**✨ Step 3: View & Pure Functions**

✅ Code

Create: ViewAndPure.sol

Paste:

solidity

Copy

Edit

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.3;

contract ViewAndPure {

uint public x = 1;

function addToX(uint y) public view returns (uint) {

return x + y;

}

function add(uint i, uint j) public pure returns (uint) {

return i + j;

}

}

✅ Compile

Compile ViewAndPure.sol.

✅ Deploy

Deploy ViewAndPure.

✅ Test

addToX(5) → returns 6.

add(2,3) → returns 5.

**✨ Step 4: Fallback Function**

✅ Code

Create: FallbackFn.sol

Paste

solidity

Copy

Edit

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.17;

contract fallbackfn {

event Log(string func, address sender, uint value, bytes data);

fallback() external payable {

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

}

receive() external payable {

emit Log("receive", msg.sender, msg.value, "");

}

}

✅ Compile

Compile FallbackFn.sol.

✅ Deploy

Deploy fallbackfn.

✅ Test Fallback and Receive

In Deployed Contracts, copy the contract address.

To trigger receive():

In Remix, expand Deployed Contracts.

In VALUE field at the top, enter e.g., 1 ether.

Click Transact (this sends ether with empty calldata)

To trigger fallback():

Switch to Low-level interactions (in Remix console).

Paste the contract address and send a transaction with data.

Watch Logs emitted in the console.