**3B Part 1: Function Overloading**

**📄 Create a file:**

**CopyEdit**

**FunctionOverloading.sol**

**Paste this code:**

**solidity**

**CopyEdit**

**// SPDX-License-Identifier: MIT**

**pragma solidity ^0.8.17;**

**contract FunctionOverloading {**

**function sum(uint a) public pure returns (uint) {**

**return a + 10;**

**}**

**function sum(uint a, uint b) public pure returns (uint) {**

**return a + b;**

**}**

**function sum(uint a, uint b, uint c) public pure returns (uint) {**

**return a + b + c;**

**}**

**function exampleUsage() public pure returns (uint, uint, uint) {**

**uint result1 = sum(5); // Calls the first**

**uint result2 = sum(5, 10); // Calls the second**

**uint result3 = sum(5, 10, 15); // Calls the third**

**return (result1, result2, result3);**

**}**

**}**

**🧪 Deploy and Test:**

1. **Deploy FunctionOverloading**
2. **Click exampleUsage()**

**✅ Output should be:**

* **15, 15, 30**

**✅ 3B Part 2: Mathematical Functions**

**📄 Create a file:**

**CopyEdit**

**MathOperations.sol**

**Paste this:**

**solidity**

**CopyEdit**

**// SPDX-License-Identifier: MIT**

**pragma solidity ^0.8.17;**

**contract MathOperations {**

**function computeMod() public pure returns (uint addModResult, uint mulModResult) {**

**uint x = 3;**

**uint y = 2;**

**uint k = 6;**

**addModResult = addmod(x, y, k); // (3 + 2) % 6 = 5**

**mulModResult = mulmod(x, y, k); // (3 \* 2) % 6 = 0**

**}**

**}**

**🧪 Deploy and Test:**

1. **Deploy MathOperations**
2. **Call computeMod()**

**✅ Output:**

* **addModResult = 5**
* **mulModResult = 0**

**✅ 3B Part 3: Cryptographic Functions**

**📄 Create a file:**

**CopyEdit**

**CryptoDemo.sol**

**Paste this:**

**solidity**

**CopyEdit**

**// SPDX-License-Identifier: MIT**

**pragma solidity ^0.8.17;**

**contract CryptoDemo {**

**function callKeccak256() public pure returns(bytes32 result) {**

**return keccak256(abi.encodePacked("BLOCKCHAIN"));**

**}**

**function callSha256() public pure returns(bytes32 result) {**

**return sha256(abi.encodePacked("BLOCKCHAIN"));**

**}**

**function callRipemd160() public pure returns(bytes20 result) {**

**return ripemd160(abi.encodePacked("BLOCKCHAIN"));**

**}**

**}**

**🧪 Deploy and Test:**

1. **Deploy CryptoDemo**
2. **Click each of:**
   * **callKeccak256()**
   * **callSha256()**
   * **callRipemd160()**

**✅ You’ll get different hash values for the same input string using different algorithms.**