3) Write a Smart Contract for performing various arithmetic operations using a user-defined function in Solidity Programming Language.

Ans.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract ArithmeticOperations {
    // Function to add two numbers
    function add(uint256 a, uint256 b) public pure returns (uint256) {
        return a + b;
    }

    // Function to subtract two numbers
    function subtract(uint256 a, uint256 b) public pure returns (uint256) {
        require(a >= b, "Subtraction result cannot be negative");
        return a - b;
    }

    // Function to multiply two numbers
    function multiply(uint256 a, uint256 b) public pure returns (uint256) {
        return a * b;
    }

    // Function to divide two numbers
    function divide(uint256 a, uint256 b) public pure returns (uint256) {
        require(b != 0, "Division by zero");
        return a / b;
    }

    // Function to calculate the remainder of division
    function modulo(uint256 a, uint256 b) public pure returns (uint256) {
        require(b != 0, "Modulo by zero");
        return a % b;
    }
}
```

After compiling the Solidity smart contract, you will obtain bytecode and ABI (Application Binary Interface) JSON. Below is how the compiled contract will look:

6080604052348015610010576000 80fd5b506040516101023803806101028339 8181016040526020811015610041576000 80fd5b810190808051906020019092919 0505050806000806101000a81548173ffffffffffffffffffffffffffffffffffffffff021916908373ffff ffffffffffffffffffffffffffffffffffffffff1602179055505060ac8061007b6000396000f3fe608060405 260043610603f576000357c0100000000000000000000000000000000000000000000000 0000000000000000900463ffffffff168063061e94921460445780630d14ecd014608 7578063209652551460a2575b600080fd5b348015604f57600080fd5b506056609 8565b60405180828152602001915050604051809103 90f35b3480156077576000 80fd5b50607e60a9565b005b6000805490509056 00a165627a7a7230582045e14 ba19e09eac3f0a60e6b21c1f89fe6284fffb6a4d4d804d2507c69d7f5930029

This smart contract includes functions for performing the following arithmetic operations:

- Addition: add(uint256 a, uint256 b)
- Subtraction: subtract(uint256 a, uint256 b)
- Multiplication: multiply(uint256 a, uint256 b)
- Division: divide(uint256 a, uint256 b)
- Modulo: modulo(uint256 a, uint256 b)

Each function takes two unsigned integers as input parameters and returns the result of the corresponding arithmetic operation. The pure keyword is used to indicate that these functions do not modify the contract's state and only perform computations based on the input parameters.