**Blockchain Study Notes Day 6:**

**Module 2 - Solidity Basics**  
**Chapter 2 - Variables and Constants in Solidity**

**Introduction to Variables and Constants**

In Solidity, variables are used to store data, while constants store unchangeable values. Understanding the types and usage of these elements is crucial for efficient smart contract development.

**1. Types of Variables in Solidity**

Solidity supports three main types of variables:

**1.1. State Variables**

* Stored permanently on the blockchain.
* Declared outside functions.
* **Example**:

uint public myStateVariable = 10;

**1.2. Local Variables**

* Declared inside functions and exist only during function execution.
* Not stored on the blockchain.
* **Example**:

function myFunction() public pure returns (uint) {

uint localVariable = 20;

return localVariable;

}

**1.3. Global Variables**

* Special variables provided by Solidity to access blockchain information.
* Examples include msg.sender (address of the caller) and block.timestamp (current block timestamp).
* **Example**:

address public sender = msg.sender;

uint public timestamp = block.timestamp;

**2. Constants**

* Constants are immutable variables whose value is set at the time of declaration and cannot be changed.
* Declared using the constant keyword to optimize gas usage.
* **Example**:

uint public constant MY\_CONSTANT = 100;

**Benefits of Constants**:

* **Gas Efficiency**: Constants consume less gas during execution.
* **Code Clarity**: Improve code readability by indicating that the value is fixed.

**3. Variable Data Types**

Solidity supports various data types, such as:

* **Integer (uint, int)**:
  + uint: Unsigned integer (non-negative).
  + int: Signed integer (can be negative).

uint public myUint = 42;

int public myInt = -42;

* **Boolean (bool)**:
  + Stores true or false.

bool public isTrue = true;

* **Address**:
  + Stores Ethereum addresses.

address public myAddress = 0x123...;

* **String**:
  + Stores text.

string public myString = "Hello, Blockchain!";

**4. Example Program Using Variables and Constants**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract VariablesAndConstants {

// State variable

uint public myStateVariable = 50;

// Constant

uint public constant MY\_CONSTANT = 100;

// Function demonstrating local and global variables

function demoVariables() public view returns (address, uint) {

uint localVariable = 10; // Local variable

return (msg.sender, localVariable + MY\_CONSTANT);

}

}

**Explanation**:

1. myStateVariable: State variable stored on the blockchain.
2. MY\_CONSTANT: A constant value for efficient gas usage.
3. localVariable: Local variable used within the function.
4. msg.sender: Global variable providing the caller’s address.

**Home Task**

1. **Modify the Example Program**:
   * Add a state variable bool isContractActive and a function to toggle its value.
2. **Experiment with Global Variables**:
   * Use block.number and block.difficulty in a function to return current block details.
3. **Write a Contract with Multiple Constants**:
   * Define constants for mathematical operations like PI = 3.14.

**Conclusion**

Understanding variables and constants is essential for efficient smart contract design in Solidity. By leveraging state, local, and global variables, along with constants, developers can create contracts that are both functional and optimized for gas usage.

Day 6 Notes

***Prepared by Munawar Johar***