Apply Fundamentals of Blockchain

Lecturer 5

By: Japhet Moise H.

Set up solidity environment

- Installing Code editor (remix, visual studio code)
- Installing node.js and npm (Node Package Manager) for package management.
- Use this link: https://nodejs.org/en/download/package-manager/
- Installing Solidity compiler (solc) and Ethereum development tools (e.g.,Truffle, Hardhat).
- Use this link: https://docs.soliditylang.org/en/latest/installing-solidity.html

Solidity program syntax

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.8.0;
contract HelloGeeks {
uint a;
function set(uint x) public {
a = x;
}
function get() public view returns (uint) {
return a;
}
}
```

- **Pragma**
- The first line is a pragma directive which tells that the source code is written for Solidity version
- **Contract**
- A Solidity contract is a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain.
- The line uint storedData declares a state variable called storedData of type uint and the functions set and get can be used to modify or retrieve the value of the variable(uint=undefined intiger)

- **Importing Files**

import "filename";

- **Variables**

Variables in Solidity are used to store data that can be used later in the program. There are different types of variables like uint (unsigned integer), string, bool (boolean), etc.

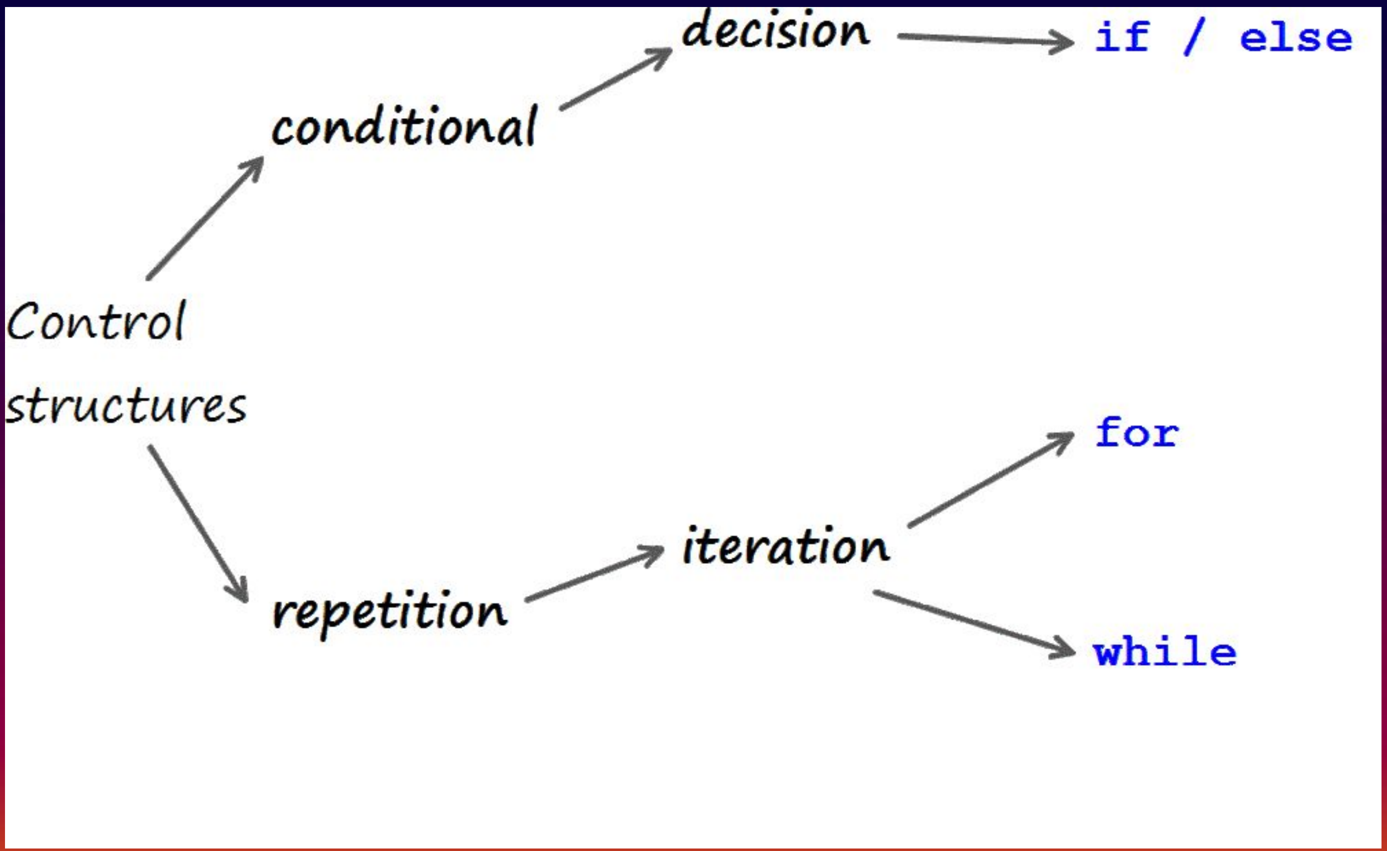
- Example uint =23;

Functions

- Functions in Solidity are similar to functions in other programming languages. They are used to perform specific tasks or operations.

```
function set(uint x) public {
a = x; }
function get() public view returns (uint) {
return a;}
```

Control Structures



- Control structures are used to control the flow of the program. Solidity supports various control structures such as if/else statements, loops.

Keyword Explanation

- **abstract :** Indicates that a contract or function is incomplete and must be implemented by a child contract.
- **Address:** A 20-byte Ethereum address.
- **Bool:** A boolean value (true or false).
- **Block:** Exits a loop or switch statement.
- **Bytes:** A dynamic byte array.
- **bytes32 :** A 32-byte array.
- **Constant:** Indicates that a function does not modify the contract state.
- **Contract:** Defines a smart contract.

- enum: A user-defined type that can only have a certain set of values.
- **Event:** A way to log an occurrence in the contract.
- **External:**Indicates that a function can only be called from outside the contract.
- **Function:** Defines a function.
- **if :** A conditional statement.

Thank you!!!