Solidity
Data structures

Intro

A data structure is a way to store and organise data for effective usage and optimisation in any programming language.

Other complex data structures in Solidity includes:

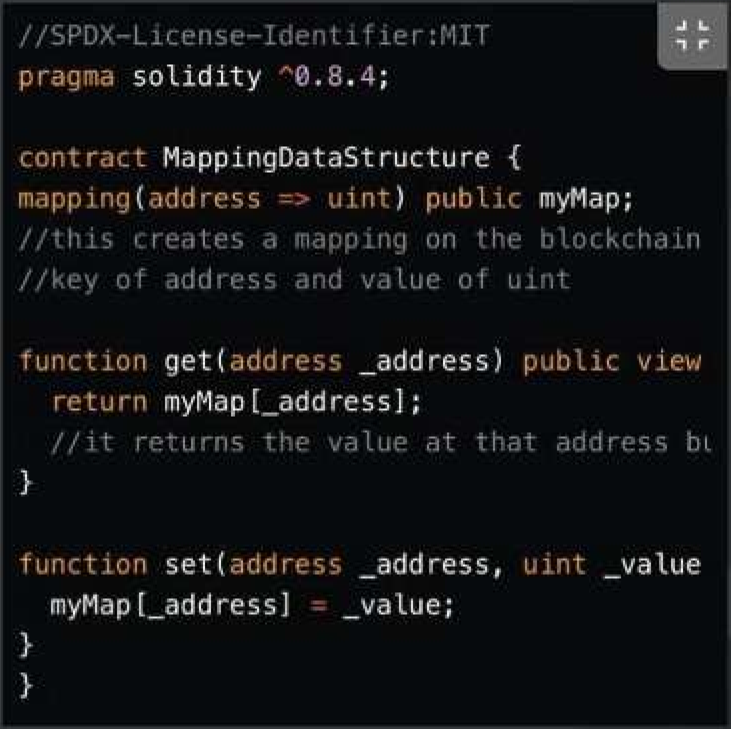
- Arrays
- Mapping
- Struct
- Enum

The primitive data structures in solidity are:

- uint
- int
- address

Mapping

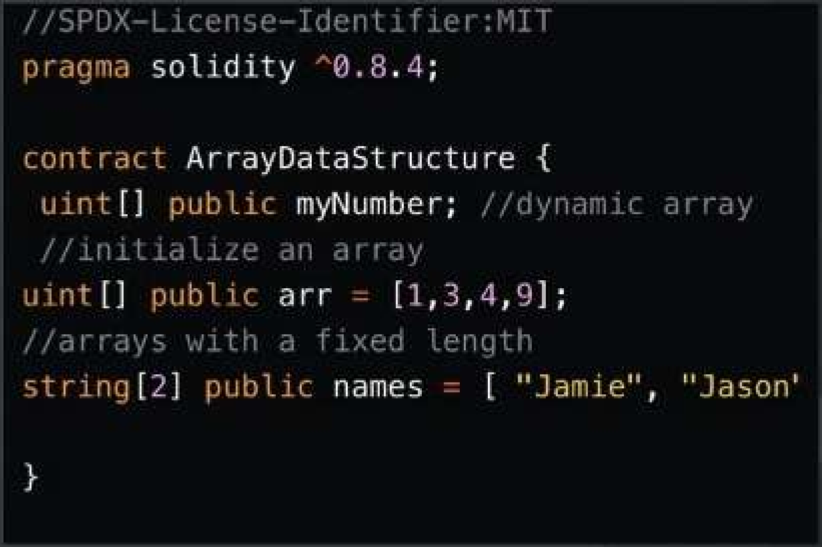
A map is Solidity is a data structure with a key:value pair store of data.



```
//SPDX-License-Identifier:MIT
pragma solidity ^0.8.4;

contract MappingDataStructure {
mapping(address => uint) public myMap;
//this creates a mapping on the blockchain
//key of address and value of uint

function get(address _address) public view
  return myMap[_address];
  //it returns the value at that address bu
}

function set(address _address, uint _value
  myMap[_address] = _value;
}
}
```

Arrays

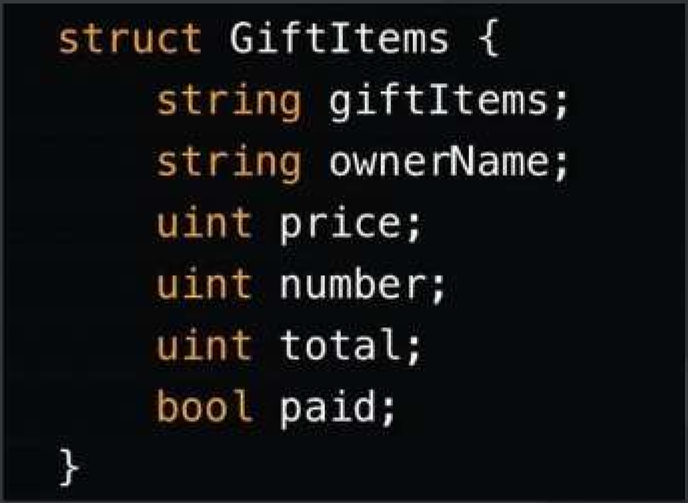
Arrays in Solidity can be of dynamic length where we can add infinite amount of items in our shopping list and we also have arrays of fixed length.



```
//SPDX-License-Identifier:MIT
pragma solidity ^0.8.4;

contract ArrayDataStructure {
 uint[] public myNumber; //dynamic array
 //initialize an array
uint[] public arr = [1,3,4,9];
//arrays with a fixed length
string[2] public names = [ "Jamie", "Jason'

}
```

Structs

Structs are your own defined types. A structs can be defined outside a contract and then imported into the contract. They can contain any value type.



```
struct GiftItems {
    string giftItems;
    string ownerName;
    uint price;
    uint number;
    uint total;
    bool paid;
}
```