

SPPU-BE-COMP-CONTENT - KSKA Git

BT

CLASSMATE

Date :

Page :

ASSIGNMENT - 4

Q1

- A structure in Solidity is a user defined data type that groups together multiple variables of different types under one name.
- It is useful when we want to represent a real world entity (Student, employee, Product)

Example:

```
contract StudentData {
```

```
    struct Student {
```

```
        uint id;
```

```
        string name;
```

```
        uint age;
```

```
    }
```

```
    Student public s1 = Student(1, "Alice", 20);
```

```
}
```

Q2

- Array in Solidity is a data structure that stores a sequence of elements of the same type.
- Arrays can be fixed size or dynamic.
- Useful to store items like "Student ID's", balances, or addresses.

SPPU-BE-COMP-CONTENT - KSKA Git

Example:

```
pragma solidity ^0.8.0;
contract studentArray {
    uint[] public marks;
    function addMark (uint _mark) public {
        marks.push(_mark);
    }
    function getMark (uint index) public view returns(uint) {
        return marks[index];
    }
}
```

Q3

- Fallback function is special unnamed function in solidity
- It is executed when:
 1. A contract receives ether without data.
 2. A function call does not match any defined function.
- Must be marked payable if it is to receive Ether
- Use cases:
 - collect Ether sent to contract
 - Handle incorrect function's calls gracefully.

```
pragma solidity ^0.8.0;
contract FallbackExample {
    event Received (address sender, uint value);
    fallback () external payable {
        emit Received (msg.sender, msg.value);
    }
}
```