Group C

Assignment No: 4

Title of the Assignment: Write a program in solidity to create Student data. Use the following constructs:

- Structures
- Arrays
- Fallback

Deploy this as smart contract on Ethereum and Observe the transaction fee and Gas values.

Objective of the Assignment: Students should be able to learn about Solidity. Its datatypes and implementations.

Prerequisite:

1. Basic Programming Logic
2. Basic knowledge of Solidity

Contents for Theory:

1. Solidity - Arrays

2. Solidity - Structures

3. Solidity – Fallback

4. Implementation

1. **Solidity – Arrays :**

Arrays are data structures that store the fixed collection of elements of the same data types in which each and every element has a specific location called index. Instead of creating numerous individual variables of the same type, we just declare one array of the required size and store the elements in the array and can be accessed using the index. In Solidity, an array can be of fixed size or dynamic size. Arrays have a continuous memory location, where the lowest index corresponds to the first element while the highest represents the last.

Creating an Array

To declare an array in Solidity, the data type of the elements and the number of elements should be specified. The size of the array must be a positive integer and data type should be a valid Solidity type

Syntax:

<data type> <array name>[size] = <initialization>

Fixed-size Arrays :

The size of the array should be predefined. The total number of elements should not exceed the size of the array. If the size of the array is not specified then the array of enough size is created which is enough to hold the initialization.

Dynamic Array:

The size of the array is not predefined when it is declared. As the elements are added the size of array changes and at the runtime, the size of the array will be determined.

Array Operations :

- **Accessing Array Elements:**

The elements of the array are accessed by using the index. If you want to access ith element then you have to access (i-1)th index.

- **Length of Array:**

  Length of the array is used to check the number of elements present in an array. The size of the memory array is fixed when they are declared, while in case the dynamic array is defined at runtime so for manipulation length is required.

- **Push:**

  Push is used when a new element is to be added in a dynamic array. The new element is always added at the last position of the array.

- **Pop:**

  Pop is used when the last element of the array is to be removed in any dynamic array.

- **Solidity – Structures:**

  Structs in Solidity allows you to create more complicated data types that have multiple properties. You can define your own type by creating a **struct**.
  They are useful for grouping together related data.

  Structs can be declared outside of a contract and imported in another contract. Generally, it is used to represent a record. To define a structure *struct* keyword is used, which creates a new data type.

  Syntax:

```
struct <structure_name> {
   <data type> variable_1;
   <data type> variable_2;
}
```

For accessing any element of the structure, 'dot operator' is used, which separates the struct variable and the element we wish to access. To define the variable of structure data type structure name is used.

2. Solidity – Fallback :

The solidity fallback function is executed if none of the other functions match the function identifier or no data was provided with the function call. Only one unnamed function can be assigned to a contract and it is executed whenever the contract receives plain Ether without any data. To receive Ether and add it to the total balance of the contract,

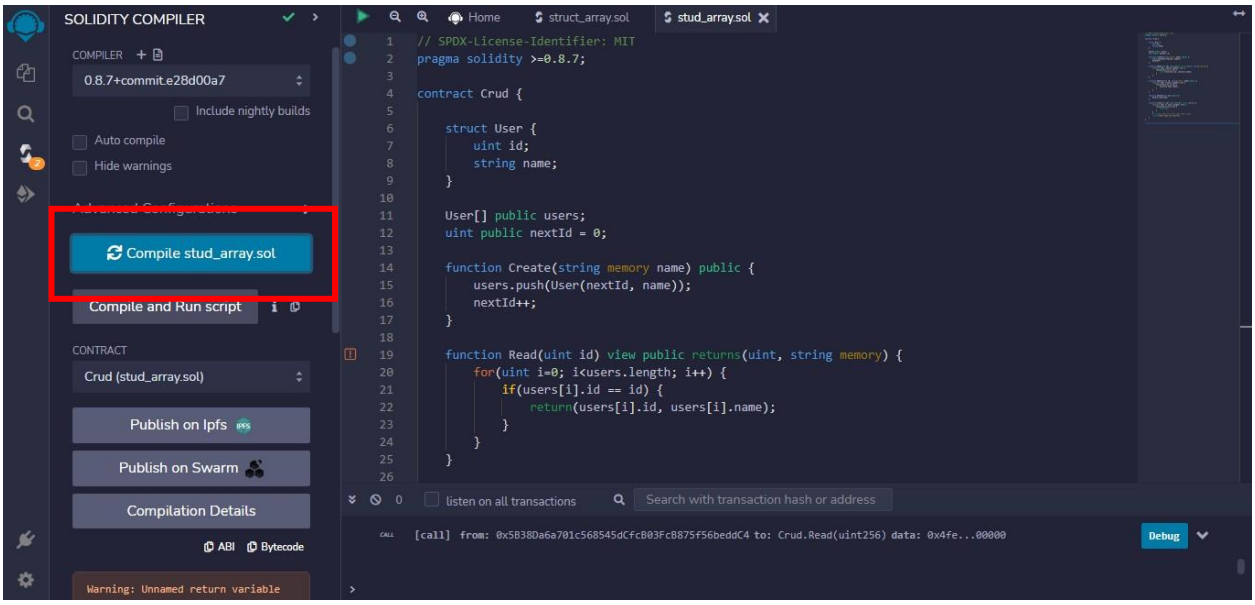
the fallback function must be marked payable. If no such function exists, the contract cannot receive Ether through regular transactions and will throw an exception**.**

Properties of a fallback function:

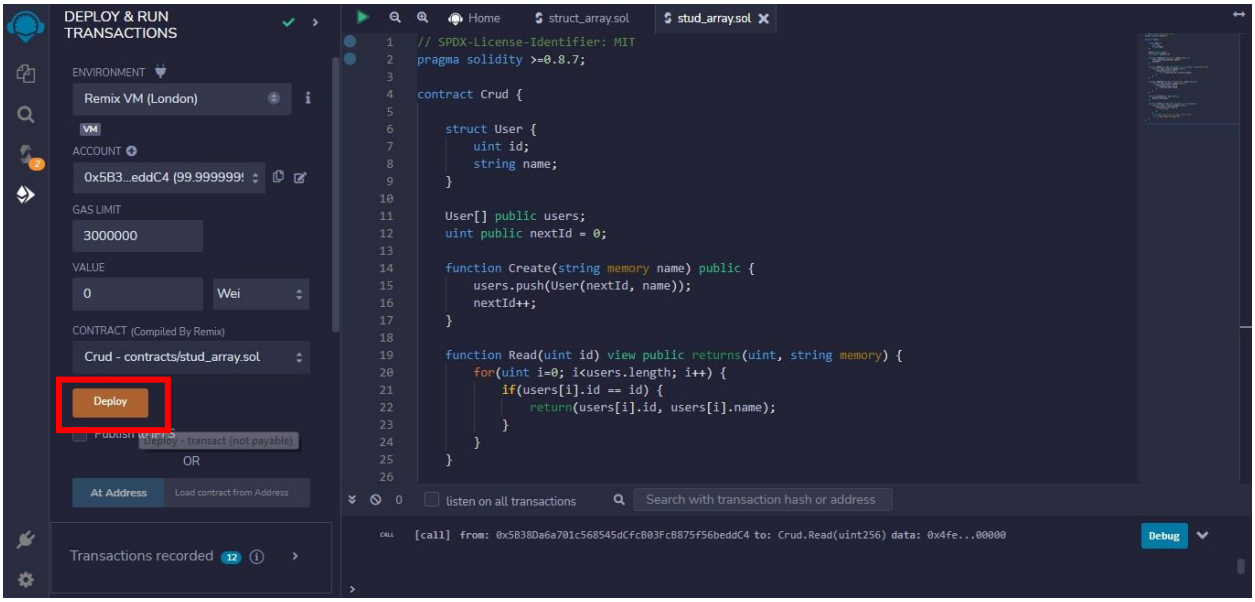
1. Has no name or arguments.
2. If it is not marked **payable**, the contract will throw an exception if it receives plain ether without data.
3. Cannot return anything.
4. Can be defined once per contract.
5. It is also executed if the caller meant to call a function that is not available
6. It is mandatory to mark it external.
7. It is limited to 2300 gas when called by another function. It is so for as to make this function call as cheap as possible.

Output –

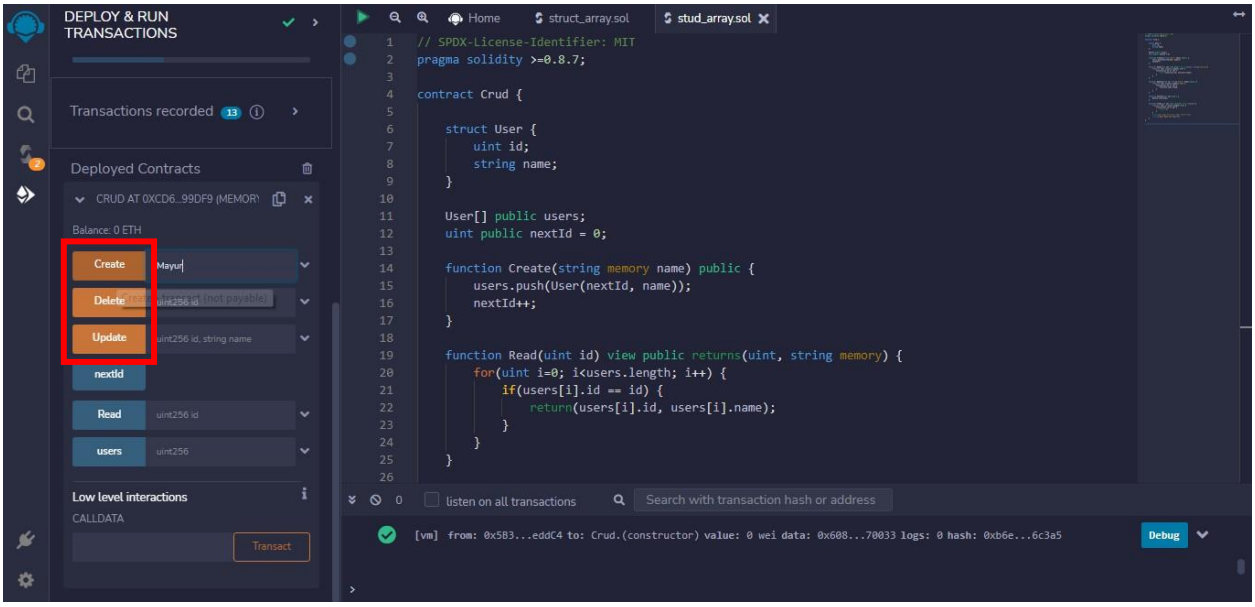
Step 1 – Compile the program by clicking on compile button.



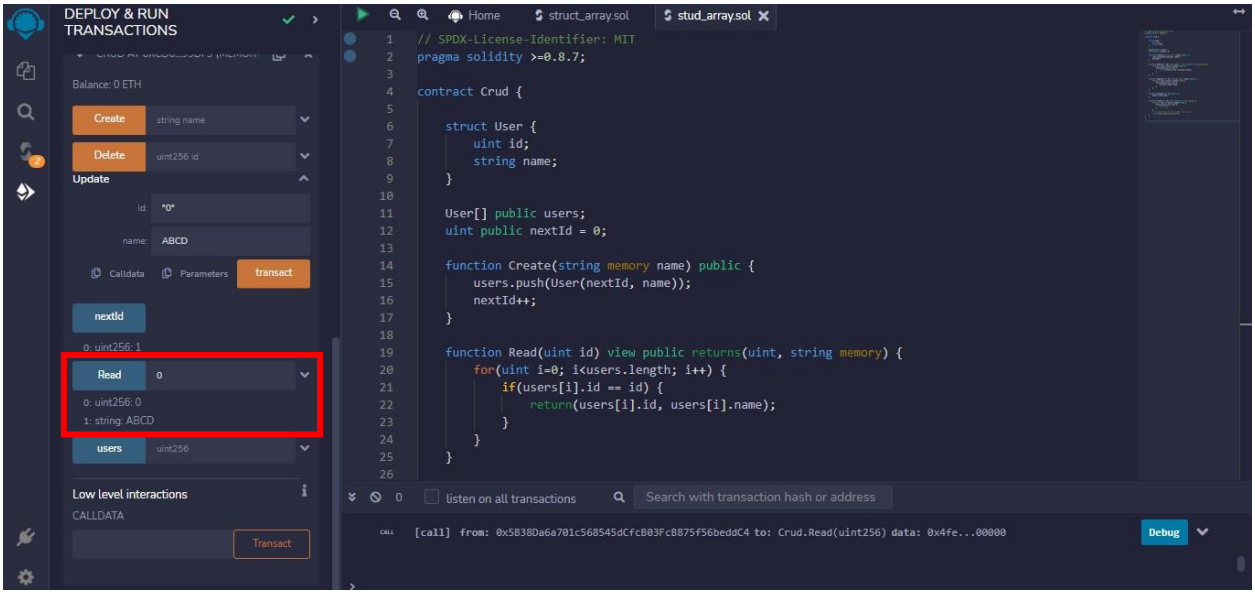
Step 2 – After the Successful compilation, Deploy the contract to see the output.


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Remix VM (London)
VM
ACCOUNT
0x5B3...eddC4 (99.999999
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT (Compiled By Remix)
Crud - contracts/stud_array.sol
Deploy
OR
At Address
Load contract from Address
Transactions recorded 12
Home
struct_array.sol
stud_array.sol
// SPDX-License-Identifier: MIT
pragma solidity >=0.8.7;
contract Crud {
struct User {
uint id;
string name;
}
User[] public users;
uint public nextId = 0;
function Create(string memory name) public {
users.push(User(nextId, name));
nextId++;
}
function Read(uint id) view public returns(uint, string memory) {
for(uint i=0; i<users.length; i++) {
if(users[i].id == id) {
return(users[i].id, users[i].name);
}
}
}
listen on all transactions
Search with transaction hash or address
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Crud.Read(uint256) data: 0x4fe...00000
Debug

Step 3 - Now you can check the output. You can insert, update and delete the student data using your smart contract.



Step 4 – After entering your data, you can read the data using ID.



Conclusion-

In this way we have created array, structure and used fallback function in solidity and mapped with CO6.

Group C

Assignment No: 5

Title of the Assignment: Write a survey report on types of Blockchains and its real time use cases.

Objective of the Assignment: Students should be able to learn about different use cases/ real timeapplication of Blockchain and perform survey on one of the case study.

Contents for Theory:

A blockchain is a digital ledger of all cryptocurrency transactions. It is constantly growing as "completed" blocks are added to it with a new set of recordings. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data. Bitcoin nodes use the block chain to differentiate legitimate Bitcoin transactions from attempts to re-spend coins that have already been spent elsewhere.

Types of Blockchain

There are three types of blockchains- public, private and consortium.

1. **Public Blockchain:** A public blockchain has absolutely no access restrictions. Anyone with an internet connection can send transactions to it as well as become a validator (i.e. participate in the consensus process). Bitcoin is the best example of a public blockchain.
2. **Private Blockchain:** A private blockchain is a little more centralized. Here, a central authority controls who can access the network and who can become a validator. Validators on a private blockchain are typically vetted by the central authority. Permissioned blockchains are often used in enterprise settings where centralized control is necessary. An example of a private blockchain is Hyperledger Fabric.
3. **Consortium Blockchain:** A consortium blockchain is a hybrid of the public and private blockchain. Here, a group of companies or organizations control who can access the network and who can become a validator. The selection of validators is typically done through a voting process. Consortium blockchains are often used in cases where multiple parties need to collaborate but no party is fully trusted. An example of a consortium blockchain is the R3 Corda platform.

Real-Time Use Cases of Blockchain

1. **Supply Chain Management** Blockchain can be used to create an immutable record of all the transactions in a supply chain. This can help to increase transparency and traceability in the supply chain.
2. **Identity Management** Blockchain can be used to create a digital identity for individuals, organizations, and devices. This can be used for KYC (know your customer) and AML (anti-money laundering) compliance.
3. **Payments** Blockchain can be used to process payments. This can be done using cryptocurrencies or fiat currencies.
4. **Data Management** Blockchain can be used to store data in a tamper-proof and decentralized manner. This can be used for data sharing and data security.
5. **IoT** Blockchain can be used to create a decentralized network of IoT devices. This can be used for data sharing and data security.
6. **Predictive Analytics** Blockchain can be used to create a decentralized network of predictive analytics models. This can be used for data sharing and data security.

Conclusion: Hence, we have studied to write a survey report on types of Blockchains and its real time application and mapped with CO6.