3.Title:-Smart contract on test network for bank account of customer following operation

1.Deposite

2.Withdraw money

3. Show balance

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;

/**
 * @title Ballot
 * @dev Implements voting process along with vote delegation
 */

 contract SimpleBank{

    struct client_account{
        int client_id;
        address client_address;
        uint client_balance_in_ether;

    }
    client_account[] clients;

    int clientCounter;
    address payable manager;

    modifier onlyManager(){
        require(msg.sender==manager,"only manager can call this!");
        _;
    }
modifier onlyClients(){
    bool isClient=false;

    for(uint i=0;i<clients.length;i++){
        if(clients[i].client_address==msg.sender){
            isClient=true;
            break;
        }
    }
  require(isClient,"only clients can call this!");
  _;
}

constructor(){
    clientCounter=0;
}
receive() external payable{}

function setManager(address ManagerAddress)public returns(string memory){
    manager=payable(ManagerAddress);

    return " ";
}

function joinAsClient() public payable returns(string memory){
    clients.push(client_account(clientCounter++,msg.sender,address(msg.sender).balance));
    return " ";
```

```
}
function deposit() public payable onlyClients{
    payable(address(this)).transfer(msg.value);
}

function withdraw(uint amount) public payable onlyClients{
    payable(msg.sender).transfer(amount*1 ether);


}
function sendInterest() public payable onlyManager{
    for(uint i=0;i<clients.length;i++){
        address initalAddress=clients[i].client_address;


        payable(initalAddress).transfer(1 ether);
    }
}

function getContractBalance() public view returns(uint){
    return address(this).balance;
}
 }
```

4. Write program on solidity to create student data.use following constructs:

1.Structures

2.Array

3.Fallback

Deploy this as smart contract on etherum and observe the transaction fees and gas values.

```
// SPDX-License-Identifier: MIT
//https://betterprogramming.pub/developing-a-smart-contract-by-using-re mix-ide-81ff6f44ba2f
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);
}
}
return (0, "");
}
function Update(uint id, string memory name) public {
```

```
for(uint i=0; i<users.length; i++) {

if(users[i].id == id) {

users[i].name =name;

}

}

}

function Delete(uint id) public {

delete users[id];

}

function find(uint id) view internal returns(uint) {

for(uint i=0; i< users.length; i++) {

if(users[i].id == id) {

return i;

}

}

// if user does not exist then revert back

revert("User does not exist");

}

}
```