## IE421 Blockchain Systems: Assignment 1

## Using Ethereum and Ethereum Token Implementation

This assignment contains two tasks. To develop and test your solution, it is strongly suggested to use the following environment:   
Firefox or Chrome browser, Remix IDE (remix.ethereum.org), Metamask browser extension wallet.

**Task 1: Using Ethereum**

You must carry out the following tasks (in the order in which they are numbered):

1. Create three accounts on the Goerli Ethereum test network. We will refer to these accounts as Venus, Serena, and Roger.
2. Obtain at least ~~0.7~~ 0.08 ETH for each of the three accounts that you created from the Goerli ETH faucet <https://goerli-faucet.pk910.de>
3. Execute transaction T1 that transfers ~~0.09~~ 0.02 ETH from Serena to Roger
4. ~~Initialize (i.e., create and confirm) a chess game between Serena and Roger.   
   The BC4C-ETH contract to use is already deployed at this address on the Goerli Ethereum network:~~ **~~0x5DF3CE0800654553CF9574198691066e8Aba259A~~**  
     
   Deploy the contract BC4C-ETH using any of the users that you created. The Solidity code of the contract BC4C-ETH-sol is available here: <https://1drv.ms/u/s!Am9gvaL7i67Rg4BqgBmLE-SDv8016w?e=KzIdet>   
     
   ~~That is, do not re-deploy the smart contract BC4C-ETH, use instead the contract above that I have already deployed.~~  
   Then, initialize (i.e., create and confirm) a chess game between Serena and Roger.The game is created by Serena (transaction T2) and confirmed by Roger (T3). Use the last 4 digits of your student id as the id of the game (if the 4th-last digit is a 0, then use 1 instead: e.g., 0065 >> 1065; if that id is already taken use a different one that works).
5. T4: Serena pays the deposit of the game that you created at the previous step

**Task 2 Ethereum Token Implementation**

Your task is to create, deploy and use an ERC-20 Ethereum token that satisfies the requirements listed below.

It is also strongly suggested to solve this assignment extending/modifying the code of ~~ubstv1.sol~~ and ubstv2.sol (available here: <https://1drv.ms/u/s!Am9gvaL7i67Rg4F3PFTCfVp_fjEJfA?e=zp3gIB> ) demonstrated during the lectures.

As part of the assignment, you also must deploy the token that you implemented on the Ethereum Goerli test network and show evidence of its use.

**REQUIREMENTS**

**R1: Token name and symbol**

The token name is “<Your student id> IE421Assignment 1”   
The token symbol is: IEA1

The token has 5 decimals.

**R2: Initial supply**

The total initial supply of 900 tokens is divided equally among the 3 users that you have created for Task 1: Venus, Serena, and Roger. The token can be initialized only once and only by Venus.

**R2: Standard functions**

The token implements the following functions and events in the same way as the token of ubstv1.sol:

function transfer(address recipient, uint amount)

function transferFrom(address from, address to, uint value)

function approve(address spender, uint value)

The token also implements the following function, which returns the number of tokens owned by an Ethereum account \_account

function balanceOf(address \_account) public view returns (uint);

**R3: Additional functions**

The token implements the following functions:

function extendSupply() public payable

function reduceSupply() public payable

The function extendSupply() allows Venus to extend the token supply of 100 tokens. The token supply is extended only if ~~0.025~~ 0.009 ETH are transferred to the token contract when calling this function. The 100 newly created tokens are assigned to Venus. If any other user other Venus calls this function, an error message is returned and the token supply is not extended.

A global counter in the contract uint extendNo is initialized to 0 when the contract is created and increased of one unit each time the function extendSupply() is called.

The function reduceSupply() allows Venus to reduce the token supply of 100 tokens. If Venus has 100 tokens or more, than the execution of this function will decrease Venus’s token balance of 100. At the same time, Venus ETH balance is increased of ~~0.015~~ 0.007. This function can be called only if extendNo >= 1. The execution of this function decreases the value of extendNo of one unit (note that in this way the contract will always have the 0.007 ~~tokens~~ ETH that must be paid to Venus). If Venus has less than 100 tokens, or if any other than Venus invokes this function, then an error message is returned and the token supply is not modified.

**Using the token**

After having coded the smart contract, you must execute the following transactions (in the order in which they are numbered):

1. T1: Venus deploys the contract.
2. T2: Venus extends the supply once.
3. T3: Venus extends the supply a second time.
4. T4: Venus reduces the supply once.
5. T5: Venus transfer 50 tokens to Roger.
6. T6: Venus transfers 50 tokens to Serena.
7. T7: Serena transfers 27 tokens to Roger
8. T8: Venus transfers 235 tokens to Serena

For verification, while you are executing the assignment above, fill out the following table:

|  |  |
| --- | --- |
|  | URL on Etherscan |
| Venus account |  |
| Serena account |  |
| Roger account |  |
| BC4C-ETH SC address |  |
| Task 1, T1 |  |
| Task 1, T2 |  |
| Task 1, T3 |  |
| Game ID | 4 digits id of the game created: |
| Task 1, T4 |  |
| IEA1 Contract address |  |
| Task 2, T1 |  |
| Task 2, T2 |  |
| Task 2, T3 |  |
| Task 2, T4 |  |
| Task 2, T4 |  |
| Task 2, T5 |  |
| Task 2, T6 |  |
| Task 2, T7 |  |
| Task 2, T8 |  |

***Submission instructions***

Submit **on blackboard**

1. one file assignment1<student id>.sol containing the code of your contract
2. one pdf file demo.pdf containing the table above filled in with appropriate values   
   The due date of your submission is **Tuesday, November 1st 2022 at 10 PM**.

You may be randomly asked to provide a live demo of the contract that you implemented in the week after the submission deadline. More details will be communicated in due time.