CMPE 483 Sp. Top. in CMPE Blockchain Programming
Spring 2023
Homework 1 (due April 26th)
(The project can be done in groups of at most three students)

Implement an autonomous decentralized lottery as a Solidity smart contract. One lottery runs for a period of two weeks. A new lottery round starts right after the previous lottery purchase stage is completed. Lottery tickets are offered in three forms:

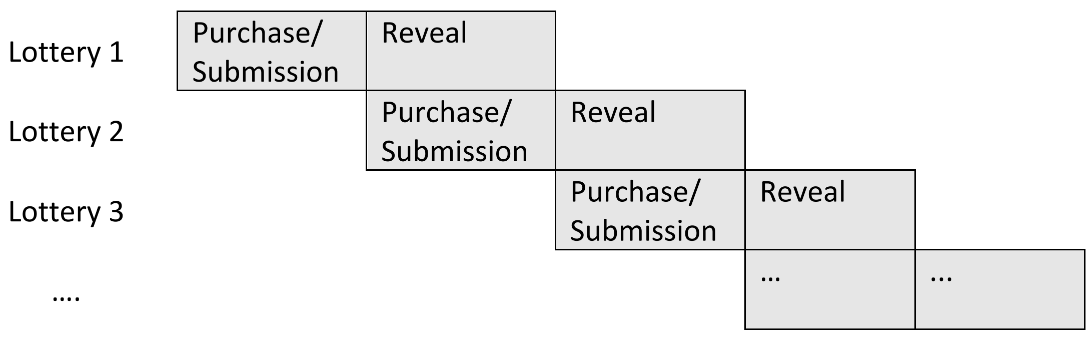
- A full ticket which costs 8 finneys,
- A half ticket which costs 4 finneys,
- A quarter ticket which costs 2 finneys.

Three unique winner tickets will be selected by computing random numbers that determine each winner ticket. Three random numbers are to be supplied by the ticker purchasers. The lottery should employ commitment (purchase submission) and reveal stages. The details of how a random number can be generated is given here:

https://ethereum.stackexchange.com/questions/191/how-can-i-securely-generate-a- random-number-in-my-smart-contract

The stages of each lottery round are scheduled as follows:

a) Ticket purchase and random number submission stage : one week.
b) Random number reveal stage : one week. Note that if previously submitted random numbers are not submitted correctly in the reveal stage, the chance of winning is lost. Also, no ticket refund is made.



Let M be the amount money collected from the sale of tickets at the current lottery round plus the amount that was carried over to the current round from the previous round. The following prizes will be awarded to the winners:

Prize	Full ticket	Half Ticket	Quarter Ticket
1st Prize	M / 2	M / 4	M / 8
2nd prize	M / 4	M / 8	M / 16
3rd Prize	M / 8	M / 16	M / 32

Note that a winning user should be able to withdraw his prize anytime after the lottery round ends.

The lottery implementation should provide the following interface to the external world:

```
function depositEther(uint amnt) public payable
function withdrawEther(uint amnt) public
```

Correction: args should be buyTicket(bytes32 hash_rnd_number,uint ticket_type), i.e. full , half or quarter ticket

```
function buyTicket(bytes32 hash_rnd_number)  public
function collectTicketRefund(uint ticket_no) public
function revealRndNumber(uint ticketno, uint rnd_number)  public
function getLastOwnedTicketNo(uint lottery_no)  public view returns(uint,uint8 status)
function getIthOwnedTicketNo(uint i,uint lottery_no)  public view returns(uint,uint8 status)
function checkIfTicketWon(uint lottery_no, uint ticket_no)  public view returns (uint amount)
function collectTicketPrize(uint lottery_no, uint ticket_no)  public
function getIthWinningTicket(uint i, uint lottery_no)  public view returns (uint ticket_no,uint amount)
function getLotteryNos(uint unixtimeinweek)  public view returns (uint lottery_no, uint lottery_no)
function getTotalLotteryMoneyCollected(uint lottery_no)  public view returns (uint amount)
```

Note that ether (ETH) must be deposited to the lottery contract. The ether balance of each user must be kept. When a user wants to purchase a ticket, the user will pay for it by deducting ether from his/her balance in Lottery contract.

You can also implement other additional functions as you like.

Grading

Your project will be graded according to the following criteria:

Documentation (written document describing how you implemented your project and also showing the correctness of your implementation). You should also provide average gas usages for the interface functions.	20%
Comments in your code	10%
Correctly functioning Solidity code, test scripts, and tests.	70%

Homework Submission:

Please submit your project to Moodle. Before you submit your project, please timestamp (notarize) your project zip file at https://certify.bloxberg.org/ Do NOT lose the certification and the submitted project zip file. The certification is a proof that your project zip file existed during the time of submission.
Please also answer the following questions and submit it with your project.

Task Achievement Table	**Yes**	**Partially**	**No**
I have prepared documentation with at least 6 pages.			
I have provided average gas usages for the interface functions.			
I have provided comments in my code.			
I have developed test scripts, performed tests and submitted test scripts as well as documented test results.			
I have developed smart contract Solidity code and submitted it.			
Function depositEther is implemented and works.			
Function withdrawEther is implemented and works.			
Function buyTicket is implemented and works.			
Function collectTicketRefund is implemented and works.			
Function revealRndNumber is implemented and works.			
Function getLastOwnedTicketNo(uint lottery_no) is implemented and works.			
Function getIthOwnedTicketNo is implemented and works.			
Function checkIfTicketWon is implemented and works.			

Function collectTicketPrize is implemented and works.			
Function getIthWinningTicket is implemented and works.			
Function getLotteryNo is implemented and works.			
Function getTotalLotteryMoneyCollected(uint lottery_no) is implemented and works.			
Function depositTL is implemented and works.			
Function withdrawTL is implemented and works.			
Function buyTicket is implemented and works.			
Function collectTicketRefund is implemented and works.			
Function revealRndNumber is implemented and works.			
Function getLastOwnedTicketNo is implemented and works.			
Function getIthOwnedTicketNo is implemented and works.			
Function checkIfTicketWon is implemented and works.			
I have tested my smart contract with 5 addresses and documented the results of these tests.			
I have tested my smart contract with 10 addresses and documented the results of these tests.			
I have tested my smart contract with 100 addresses and documented the results of these tests.			
I have tested my smart contract with 200 addresses and documented the results of these tests.			
I have tested my smart contract with more than 200 addresses and documented the results of these tests.			