# Homework 7
# Due **Wednesday Nov 4 at 11:59pm EDT**: on coursesite

CSE 297: Fall 2020

This assignment is to use Solidity to create a smart contract that allows people to wager Ether in a random lottery. When 5 people enter the contract, the contract randomly picks a winner and then distributes the collected Ether to the winning address. Of course, the "people" are actually other smart contracts, but first all you need to worry about is testing your own contract. Later, you can publish your contract address on Piazza and perhaps others will enter your lottery. Note: **all this is on a TESTNET. That means the Ether have no real value. Thus, we are not running a gambling operation in this course.**

There are several testnets. I am choosing Ropsten and ask that you use that one so that everyone's contract is on the same testnet and it is easy for you to enter each other's lottery.

The details of actually deploying the contract are conceptually simple but the details are a bit fussy. The Remix IDE has a new interface but most of the online docs describe the old one. To run on the testnet, you need Ether on the testnet, which you can get from a "faucet" (details later). But the faucets limit your access and it is easy to get yourself graylisted. To test your contract fully, you'll need multiple accounts to play your lottery. That's do-able on your one, but might be best done by getting friends to risk their testnet Ether on your contract.

Because all those details are time-consuming the first time around and possibly a bit frustrating, for grading purposes, I will accept a text file with your solidity code. I'll compile it, but won't deploy it myself. If you have a deployed contract, I'll play your lottery (if I come off the gray list and can afford it).

Please do your contract using your current groups. When it is done, have one member submit the text file on coursesite, listing group members in a Solidity comment. Include the deployed contract address on the Ropsten network if you have one.

Some details:

- The smart contract is in a .sol file (but for coursesite add a .txt extension because coursesite does not support a restriction to .sol files)
- The two input fields for the smart contract are the address and amount of Ether to wager. The default is everyone has an equal chance of winning no matter how much Ether you wager but if you want to make the project more interesting, you can design the contract so that the more Ether you wager the more likely you are to win as long as you document that in the submission file.
- Once 5 people enter the contract, the contract will randomly generate a number using the random function provided below. If you want, you may make the number of players a parameter of the constructor, but please deploy your contract with that set to 5.
- After generating a random number, the smart contract will then initiate a transaction that sends the Ether to the winning address and then restarts the lottery. Restart means forgetting the old entrants, resetting the pot of funds in the lottery to zero, and the count of entrants to zero.
- Be careful. Suppose I choose to wager 0 Ether. That would let me enter for free. So consider setting a minimum wager amount (but not so high that you can't afford to play it).
- You'll need to pick a random winner. If the variable *players* holds the number of players (fixed at 5 in the assignment, but you might want to code it as a variable), then you may use:

```
function random () private view returns(uint)
return uint(keccak256(block.difficulty, block.timestamp, players));
```

More details:

- I recommend that you use remix.Ethereum.org as your IDE although you can use whatever IDE you prefer.
- The Remix IDE is at remix.Ethereum.org. The icon buttons on the left of the remix interface were labeled in English in the prior version. Hover over the icon to get an English translation of the icon. You'll find a Solidity compiler there allowing you to enter your code, compile under various versions of the compiler (choose the newest). Once the code compiles and you are fairly confident in it you can deploy it. Once you have done that, you will see buttons allowing you to invoke public functions from the contract.
- First you need to set up an Ethereum wallet. I suggest Metamask (https://metamask.io/), but you may use whatever wallet you prefer. Next, you are going to need to get some free Ether to test your smart contracts. Go to https://faucet.ropsten.be/ or https://faucet.metamask.io/ and enter your address from Metamask, then click to button to have Ether sent. Now that you have Ether in your wallet, you can deploy and run your smart contract.

I won't write my own Remix and MetaMask tutorials here. There are lots online – but no single one that I think is best. (Bug: I look only for text, as I lack the patience to watch videos. Ironic, considering I am recording video lectures for this course.) Please post recommendations to Piazza for tutorials that you found useful.