**Lottery contract will contain**

1. Two Players
   1. As soon as someone sends money into the contract, they will be recorded as a player in the game.
2. One Manager
   1. Runs a contract that selects the winner and resets the prize pool.
3. Prize pool
   1. The ether sent into the contract from the players will be held in a prize pool.
   2. Pool resets once the manager runs a contract to select the winner and then becomes ready to accept a new list of players.

**Variables**

1. Manager
   1. Address variable type
   2. Address of person who created the contract
2. Players
   1. Address variable type
   2. Array of addresses of people who have entered

**Reference Types**

1. Fixed array
   1. Array that contains a single type of element. Has an unchanging length
2. Dynamic array
   1. Array that contains a single type of element. Can change in size over time.
3. Mapping
   1. Collection of key value pairs. All keys must be of the same type and all values must be of the same type.
4. Struct
   1. Collection of key value pairs that can have different types

**Global Variables (Don’t have to call them - always available)**

1. Msg.data
   1. ‘Data’ field from the call or transaction that invoked the current function
2. Msg.gas
   1. Amount of gas the current function invocation has available
3. Msg.sender
   1. Address of the account that started the current function invocation
4. Msg.value
   1. Amount of ether (in wei) that was sent along with the function invocation

**Functions**

1. Enter
   1. Payable function type
      1. Used to send ether
   2. Enters a player into the lottery
2. pickWinner
   1. Randomly picks a winner and sends them the prize pool

**Code Notations**

**A - Lottery.sol**

Contract will be called “Lottery”

Set variable type “address” as public with “manager” name



Call javascript function name same as contract “Lottery” to obtain address of contract initiator.

Set manager variable to msg.sender global variable type to bring in address which initiated the contract.



Initialize players as an array of addresses.



Add enter function as public, and type payable allowing ether payment.

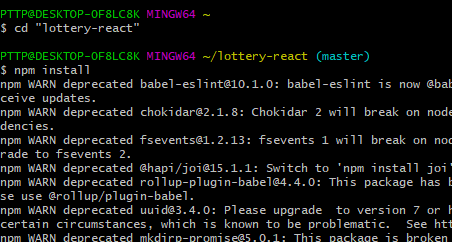
Require msg value > .01 ether.

Push new record into array (sender’s address).

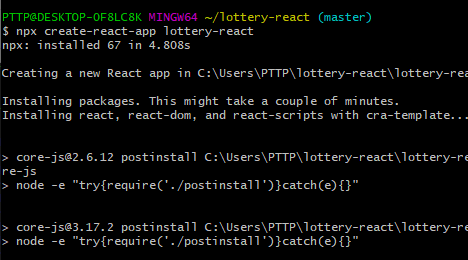


Add function “random” to return some encrypted value based on current block difficulty, current time, addresses of players.

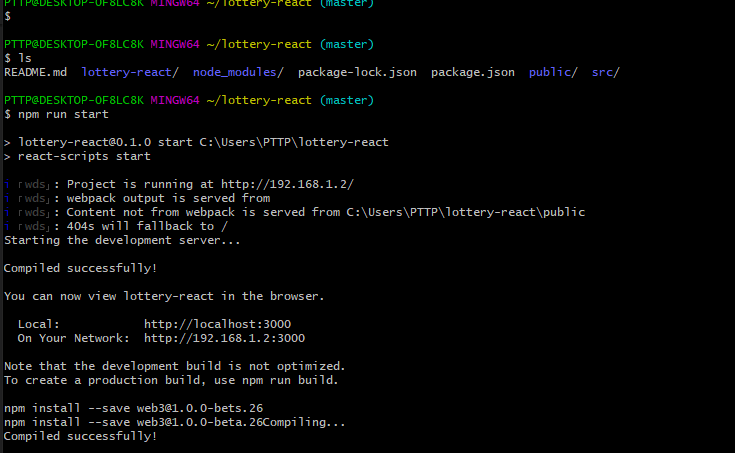
Switch over to gitbash and change directory to project “lottery-react” within github project folder.



Create a react app within project directory.



Script compiled successfully and viewable on the local network.



Localhost connects to metamask for lottery entry.

