Homework 1 (due Mar. 24, 2021)

Your first homework assignment contains small Ethereum development exercises for practice, getting familiar with tools, and experimenting. Submit via email to the TA and instructor.

Write Solidity programs (the Remix online IDE is recommended) to implement the following concepts:

1) ERC20 Token Contract:
Implement an ERC-20 Token contract according to the standard. You should implement at least functions `balanceOf`, `approve`, `transfer`, `transferFrom`, `allowance`. The events specified are not important. You can look up implementations of existing tokens.
The contract should have an `owner` storage variable, set during its construction, as well as a fallback function, to accept Ether.

Additionally, your contract should implement the following methods:

- `function mint(address recipient, uint256 amount)`
  This can only be called by the contract's owner, mints `amount` tokens, adding them to the `recipient`'s balance
- `function burn(address recipient, uint256 amount)`
  This burns `amount` of the sender's token balance, sending the ether proportionately corresponding to the burned tokens to `recipient`. That is, the Ether amount to be sent is computed using (amountBurned/totalSupply)*contract-balance. (But do the math right, since Solidity has no floating point.)

2) Reentrant burn:

Change the `burn` method of the ERC20 contract, making it vulnerable to a reentrancy attack. Use an explicit `call`, instead of `transfer`, which is desirable anyway if the recipient's fallback function may be non-trivial.

3) Exploit it:

Implement an attacker contract which, given a vulnerable ERC20 target contract, and an active balance on that contract, performs a reentrancy attack, resulting in the extraction of more funds than intended.

Bonus:
Deploy your contracts on the Ropsten testnet, perform the attack, submit the transactions as proof.