**Introduction**

In this Smart Contract audit we’ll cover the following topics:

1. Disclaimer
2. Overview of the audit and nice features
3. Overall Comments
4. Summary of the audit

**1. Disclaimer**

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

**2. Overview**

The project has 7 files, the CompanyTokenInterface.sol, CompanyToken.sol, CompanyDividendsTracker.sol, CompanyLibrary.sol, CryptoStock.sol, Company.sol, CompanyBasic.sol

***Nice Features****:*  
The contract provides a good suite of functionality that will be useful for the entire contract: It uses [SafeMath](https://github.com/OpenZeppelin/zeppelin-solidity/blob/master/contracts/math/SafeMath.sol) library to check for overflows and underflows which is a pretty good practice.

**3. Overall Comments**

1. **CompanyTokenInterface**
   1. Line 1- Specify specific version
2. **CompanyToken**
   1. Line 1- Compiler version not fixed
   2. Line 51-55 – increaseHolderTokensForSale() - This method should check before increasing token that owner actually has that much balance in his balance. For example: If Alex only owns 100 tokens than he should not able to put 150 tokens on increaseHolderTokensForSale()
   3. Line 51 - increaseHolderTokensForSale- this method should be public as it is being called internally by transferSellOrderTokens
3. **CompanyDividendsTracker**
   1. Line 1- Compiler version not fixed
4. **CompanyLibrary**
   1. Line 1- Compiler version not fixed
   2. Line 249, 237, 246, 240, 243- tx.origin is used. Never use tx.origin instead use msg.sender
   3. Line 83, 106 - Unchecked Math
   4. Not commented
   5. Line 60- Unprotected method – openInitialSellOrder() - As this method places an initial order for the contract. It should be protected and only owner should be able to call this. Currently anybody can call this method and open sell order for the contract
5. **CryptoStock**
   1. Line 35, 96- Costly Loop - If admin.length is large enough, the function exceeds the block gas limit, and transactions calling it will never be confirmed. Avoid loops with big or unknown number of steps.
   2. Line 62- Unchecked Math.
6. **Company**
   1. Line 1- Compiler version not fixed

**4. Summary of the audit**

Overall the code looks good. My final recommendation would be to pay more attention to the visibility of the functions since it’s quite important to define who’s supposed to executed the functions and to follow best practices regarding the use of assert, require etc. (which you are doing). I would suggest to use the SafeMaths library where mathematical operations are being performed.