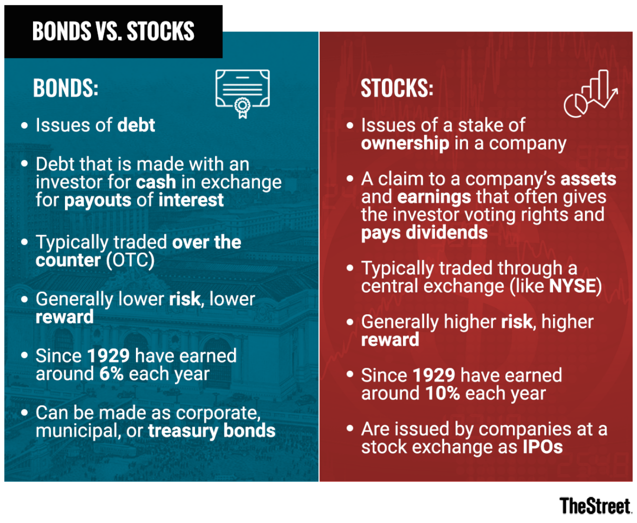
Simple Stock Market Program

Lukas Jonca
ITfu-20
2021-04-30

GitHub Repository: V10n/Project
https://github.com/V10n/Project

Program Functionality:
The purpose of the stock market program was to create an object oriented representation of different investment vehicles (stocks and bonds). The program sanctions a simple simulation of how different asset classes behave and work.



BONDS VS. STOCKS

BONDS:
- Issues of **debt**
- Debt that is made with an investor for **cash** in exchange for **payouts** of **interest**
- Typically traded **over the counter** (OTC)
- Generally lower **risk**, lower **reward**
- Since **1929** have earned around **6%** each year
- Can be made as corporate, municipal, or **treasury bonds**

STOCKS:
- Issues of a stake of **ownership** in a company
- A claim to a company's **assets** and **earnings** that often gives the investor voting rights and **pays dividends**
- Typically traded through a central exchange (like **NYSE**)
- Generally higher **risk**, higher **reward**
- Since **1929** have earned around **10%** each year
- Are issued by companies at a stock exchange as **IPOs**

TheStreet

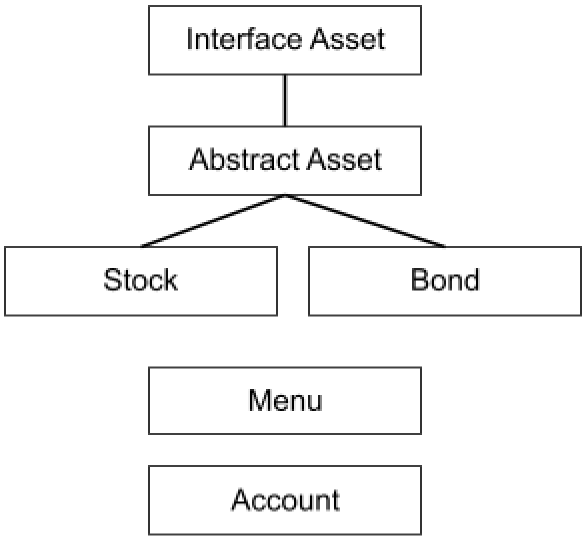
Stocks: are fractions of companies that can be bought and sold, in this case of pseudo publicly available companies.

Bonds: represent a I.O.U's that are created by companies and institutions as a means of borrowing capital at a fixed rate.

Although they are similar as they are both investments, there are some fundamental differences that must be represented in the code.

Image Source

Libraries Used:
- Iostream - provides basic input and output functionality using objects such as (cin and cout).
- Vector - allows for the usage of dynamic data and additional functionality.
- String - a data type allowing for the manipulation and storage of a collection of chars.



Structure of Classes:

IAsset - a header file only containing pure virtual functions, acting as an outline for asset class.

Asset - is an abstract class that contains shared functions and methods from IAsset but as well as some virtual functions. This class implements the methods and variables that will be used by both stock and bond. Methods such as movePrice and variables such as price.

Stock: the derived class of IAsset and Asset contains unique variables and overwritten methods specific to that Asset class such as the movePrice method and dividend.

Bond: the derived class of IAsset and Asset contains unique variables and overwritten methods specific to that Asset class such as the movePrice method and term.

Menu: A standalone class that stores variables such as the user and methods that print and get/interpret user input.

Account: A standalone that stores user data such as balance and the portfolio of stocks, methods such as order and time are used to manipulate the users portfolio and balance.

Limitations and Restrictions:
The program contains only one user so assets cannot be transacted between parties as this would add complexity to the program especially with simulating market price fluctuations (supply and demand).

The program uses a RNG instead to generate price fluctuations in stock prices, this does not model the real world but allows for variation in stock prices with positive, neutral and negative expectations.

Stocks in this case can only be longed (hold a positive number of shares), in order to short a stock (hold a negative number of shares) would require the implementation of margin and would add complexity.

Bonds follow a simplified price model for fluctuations in price and yield, in the real world these could fluctuate in different economic and market periods. But this would be difficult to model as it is often based on real life events and policy makers.

Limited number of stocks and bonds, there are only 3 stocks and bonds available and there is no ability for any to be added while the program is running. The purpose is simply to showcase the program and to help users better understand investing and differences in assets classes.

Inputs:
All user inputs are handled and interpreted by the Menu class, inputs include choosing menu options

1. buy/sell - displays stocks and allows for exchange.
2. viewing positions - showcases positions in the portfolio.
3. progressing time - progresses time with the portofolio.
4. logging out - logs the user out (ends program).

Within the buy and sell option there is user input for the asset to exchange and it’s quantity.

Outputs:
Information such as account balance/value written out with the main menu, stock/bond information is also printed (name, symbol, price, yield, term and etc) as this is important information for the user to know.

1. Balance - the amount of uninvested money the user has.
2. Portfolio value - what is the value of the entire users portfolio.
3. Name - specific name of the asset (company or institution).
4. Price - the current price for one unit of a specific asset.
5. Quantity owned - the amount of units that user owns.
6. Symbol - a unique identifier for stocks (Ex: TSLA).
7. Dividend - the profit share of a stock.
8. Yield - the interest rate of a bond.
9. Term - the length of the bond, principal is paid back at the end.

Notifications
1. The user will be alerted for invalid input (options, insufficient balance and insufficient units).
2. In the case a stock price falls below 1 cent the user will be alerted that the company has bankrupted and the stock will be erased.
3. Users are alerted for interest and principal payments of bonds.

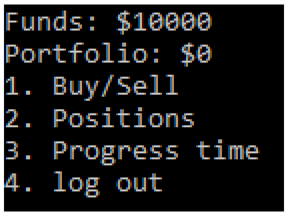
Algorithms:
Price movements and payments is one of the essential functions of the program, the program keeps track of how many days have passed and when payouts of dividends (yearly), interest (yearly) and principal (at the end of term) should be added to the users balance. For stocks the price movement algorithm also generates random price fluctuations based on the high and low values (the min/max amounts the shares can fluctuate in a year).

Orders were processed by the account function and based on user input would determine if the order is valid (has balance or quantity necessary), then the function would execute the order.

The menu also used a simple algorithm to interpret user input using a switch statement which would determine the next course of action, this also included dealing with invalid inputs.

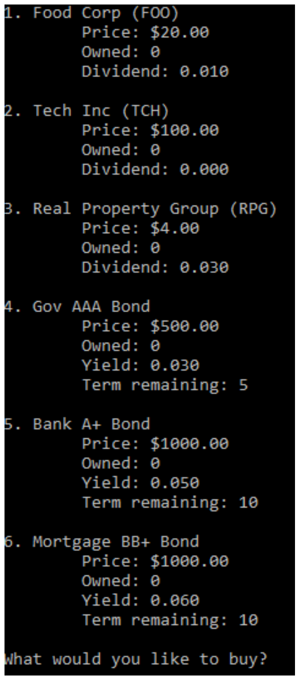
Examples:

1. Program is run and we are greeted by the menu options.



```
Funds: $10000
Portfolio: $0
1. Buy/Sell
2. Positions
3. Progress time
4. log out
```

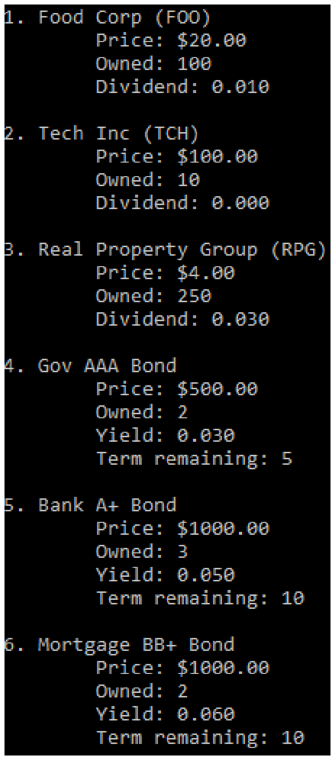
2. We currently have an empty portfolio so we chose buy/sell (1).



```
1. Food Corp (FOO)
        Price: $20.00
        Owned: 0
        Dividend: 0.010

2. Tech Inc (TCH)
        Price: $100.00
        Owned: 0
        Dividend: 0.000

3. Real Property Group (RPG)
        Price: $4.00
        Owned: 0
        Dividend: 0.030

4. Gov AAA Bond
        Price: $500.00
        Owned: 0
        Yield: 0.030
        Term remaining: 5

5. Bank A+ Bond
        Price: $1000.00
        Owned: 0
        Yield: 0.050
        Term remaining: 10

6. Mortgage BB+ Bond
        Price: $1000.00
        Owned: 0
        Yield: 0.060
        Term remaining: 10

What would you like to buy?
```

3. We are shown the menu for the list of possible option to buy, we choose Food Corp (1).
4. We are prompted “How many would you like to buy (+int = buy, -int = sell)?”
5. We buy 100, we are prompted that our order is successful.
6. The process is continued by various types of assets.

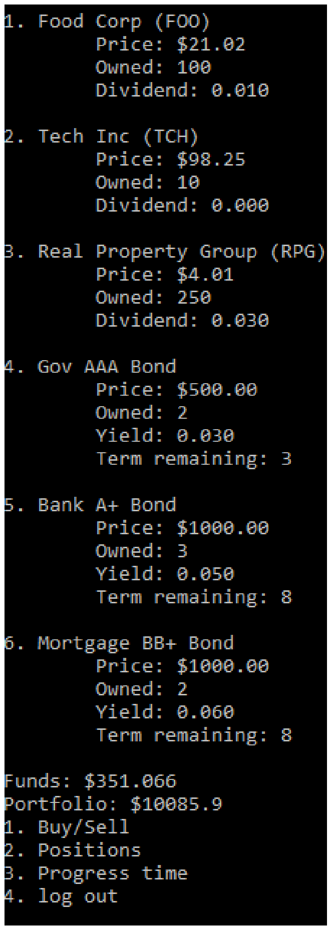
7. We use positions (2) to double check everything is correct.



```
1. Food Corp (FOO)
        Price: $20.00
        Owned: 100
        Dividend: 0.010

2. Tech Inc (TCH)
        Price: $100.00
        Owned: 10
        Dividend: 0.000

3. Real Property Group (RPG)
        Price: $4.00
        Owned: 250
        Dividend: 0.030

4. Gov AAA Bond
        Price: $500.00
        Owned: 2
        Yield: 0.030
        Term remaining: 5

5. Bank A+ Bond
        Price: $1000.00
        Owned: 3
        Yield: 0.050
        Term remaining: 10

6. Mortgage BB+ Bond
        Price: $1000.00
        Owned: 2
        Yield: 0.060
        Term remaining: 10
```

8. Everything seems fine so we decide to progress time (3), we are asked how many days and input 366.

9. We check our positions again and prices have fluctuated also dividends and interest have been paid out.



```
1. Food Corp (FOO)
        Price: $21.02
        Owned: 100
        Dividend: 0.010

2. Tech Inc (TCH)
        Price: $98.25
        Owned: 10
        Dividend: 0.000

3. Real Property Group (RPG)
        Price: $4.01
        Owned: 250
        Dividend: 0.030

4. Gov AAA Bond
        Price: $500.00
        Owned: 2
        Yield: 0.030
        Term remaining: 3

5. Bank A+ Bond
        Price: $1000.00
        Owned: 3
        Yield: 0.050
        Term remaining: 8

6. Mortgage BB+ Bond
        Price: $1000.00
        Owned: 2
        Yield: 0.060
        Term remaining: 8

Funds: $351.066
Portfolio: $10085.9
1. Buy/Sell
2. Positions
3. Progress time
4. log out
```

Conclusion: each of the menu options seem to be working as well as price movement function and buying/selling of positions.