The easiest place to simulate trading.

Start coding now

OOP Java Project
2021/2022

In our commitment to keep you safe, you will be creating a trading simulator instead of trading for real.

PROJECT DESCRIPTION

The goal of the project is to create a market simulator. The main entities in this world are markets, indexes, companies, investors, funds, currencies, and commodities.

FOR MORE QUESTIONS OR CONCERNS, KEEP READING THE FOLLOWING PAGES.

A stock market is a market where investors trade (buy and sell) stock (shares, fund participation units). A market should have a name, country, trading currency, city, address, list of stock market indexes, and percentage margin (cost) for each trading operation. A stock market index is a named collection of companies belonging to the market the index is part of. Each company can be part of one or more indexes. A company-index assignment can be constant (set by the index creator) or based on a condition (e.g., an index of 20 largest companies listed on the market). Index values are defined by the sum of share values of all the companies listed in a given index.

Each company is characterized by its name, IPO date, IPO share value, opening price, current price, minimal price, maximal price, profit, revenue, capital, trading volume (the number of times equity was bought or sold), total sales (the value of equity bought or sold).

Apart from stock markets, there are markets that allow investors to trade other assets like currencies and commodities. On currency markets, each currency has its current exchange rates (buy and sell prices in another currency). Additionally, each currency is defined by a list of countries where it is used as legal tender. Each commodity has a name, trading unit (e.g. ounce for gold), trading currency, current price, minimal and maximal price. Just like stock markets, currency and commodity markets have their transaction margins.

Investors buy and sell assets (shares, commodities, currencies) directly or through units of investment funds. Each investor has a first name, last name, trading identifier, and an investment budget. Investment funds also buy and sell assets, but only directly from markets. Each investment fund is characterized by a name and the first and last name of the fund's manager.

Functional requirements

- The user can create markets, indexes, companies, currencies, commodities, investors, and funds through a separate **control panel**
- Markets, indexes, currencies, and commodities are created on the user's command, whereas investors and funds should appear automatically, proportionally to the number of assets available on the markets
- The control panel can be used to set world parameters, such as the number of transactions per minute, or the bear/bull ratio
- When created, each company/currency/commodity/investor/fund should have their properties filled out automatically with semi-random values (even when they are created upon the user's command)
- The main window should have a list of all assets available on the markets
- Selecting an asset from the list should show a **plot of the asset's prices over time**
- There should be an option of **drawing multiple assets on one plot on a percentage scale**
- Every investor, fund, company is a **separate thread**
- The investors'/funds' threads are responsible for generating buy or sell transactions at random time intervals; moreover, an investor can increase their budget at a random moment in time
- The buy-sell price should depend on the number of investors buying/selling particular assets (or you can come up with your own rule governing the market)
- Company threads generate revenue and profit at regular time intervals
- A company thread can randomly increase its total number of shares
- The user should be able to force a buy out operation by the company (where the company buys its own shares) of a user-specified share amount; share reclaimed by the company disappear form the market
- The user can read about the basic properties of an object (investor, fund, company) in a separate **information window/panel**, which appears after selecting an object on a list
- to make multithreaded trading safe, use **semaphores or monitors**

- The application does not need to solve any deadlock problems; if there is a deadlock it will be resolved by removing a company/investor
- The project will be graded based on code quality and functionality; eye-pleasing graphics will be treated as an asset worth improving the grade, but bare-bones visualizations will in no way negatively affect the project's grade
- Similarly to visual aspects, although not required, pop-cultural references will be treated as an asset (e.g., Shark Tank, Dragon's Den, The Wolf of Wallstreet, Duck Tales, Big Short)

Technical requirements

- By **December 1, 2021**, the student is required to upload a **UML class diagram** of the project
- By **December 8, 2021**, the student should have prepared a project template with **Java classes** matching (or improving upon) the UML class diagram
- All non-static fields should be private and, if needed, accessed through **getters and setters**
- All public elements should be documented with **Javadoc** and the project should be accompanied by a Javadoc-generated HTML documentation
- The project should be uploaded to the e-learning site by **January 30, 2022**, in the form of a zip file containing:
  - The **source code** of the project
  - A **runnable *.jar** file
  - A simple **readme** containing the following information: first name, last name of the author, student ID number, short program instructions
- Delayed projects will have a -1 grade modifier for each week of delay
- Certain aspects of the project can be negotiated, a requirement can be traded for an additional feature; any changes to the project requirements must be agreed upon with the lecturer

Any plagiarism will result in a 2.0 grade given to any person copying the code of others as well as any person sharing their code; detecting plagiarism after grading the project results in changing the grade to 2.0

IN CASE OF ANY QUESTIONS
PLEASE WRITE ON THE
E-LEARNING WEB FORUM OR
WRITE TO DB