KALEIDOSCOPE
CAPITAL

Dev Case Study

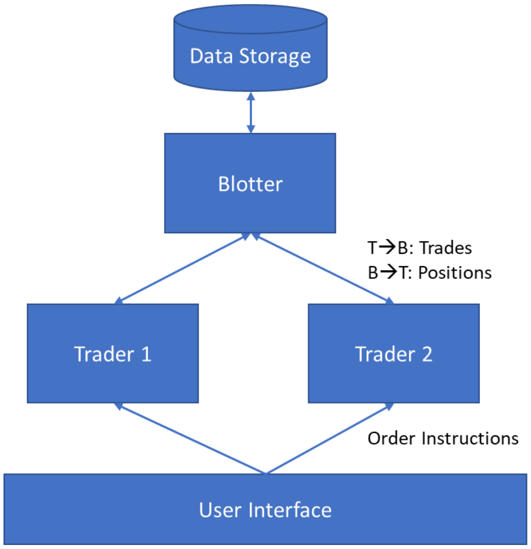
General Instructions

- The time limit of this case study is **3 hours**.
- We do recommend you skim through the case study and think about the general structure and level of complexity before implement. **No need to finish the whole case study within the time limit**. We do evaluate working in progress results.
- Feel free to use search engine to search for the formula/syntax/tools you need to solve the problems, but do **NOT** ask for someone else's assistance.
- Please attached your intermediate solutions and codes when submitting this case study.

Project – Trader Blotter Interaction

Description

In this case study you are asked to build a simple trader – blotter interaction system that executes order instructions given by portfolio managers and keeps transaction records. The system diagram is shown below.



Trader

The functionality of a trader is to translate order instructions into trades and send it to blotter to book them in the system. The inputs of traders are series of order instructions. There are two traders in the system where the order instruction can be sent to **either or both**. **Trader 1 will have the priority** to book first if both are receiving instructions at the same time.

There are two types of order instructions:

- *Actual number of trades*
  - E.g. Buy 18 shares of SPY Equity
- *% of current (absolute) position*
  - E.g. If we are currently shorting 10 shares of QQQ Equity, an order to sell 20% of current position means to sell 20% * abs(-10) = 2 shares of QQQ Equity
  - Round to **nearest integer** if the trade number is fraction
  - Do nothing if we do not have any position

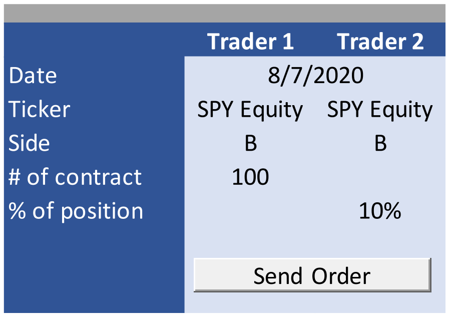
A sample series of order instructions in tabular format is attached in Appendix (also available in *order_flow.csv*).

Blotter

A blotter books the trade sent by the traders and gets the current position of specific contract for trader to determine # of contracts to trade. Feel free to choose whatever data storage methods (e.g. store in cache, flat file, pickle file) you see fit for the trade records.

User Interface

The user interface allows the system user to manually enter order instruction and send them to traders. A graphical user interface is preferred but not required as long as the user interface contains necessary functionality. No limitation on the tech stack involved in the interface (e.g. Web App, Excel, Qt). A sample user interface is shown below.



	Trader 1	Trader 2
Date	8/7/2020	
Ticker	SPY Equity	SPY Equity
Side	B	B
# of contract	100	
% of position		10%

Questions

- Assuming we have zero position at period start, given the order instruction flow in the appendix, output the position we are holding at the end.
- **[Bonus]** Feel free to add additional features into the system that can further demonstrate your capability.
- **[ANSWER ONLY, no need to implement]** What if the order instructions are coming in real time at higher frequency (instead of once a day)? How would that change your implementation?

Requirements

- OOP for implementation
- Python is preferred but feel free to use language you found most handy

Appendix

Sample Order Instruction Flow

Trader	Date	Ticker	Side	# of contract	% of position
Trader 1	7/1/2020	SPY Equity	B	100	
Trader 2	7/1/2020	SPY Equity	B		10%
Trader 2	7/3/2020	QQQ Equity	B	50	
Trader 2	7/4/2020	HYG Equity	B	70	
Trader 1	7/5/2020	QQQ Equity	B	20	
Trader 2	7/5/2020	QQQ Equity	B		40%
Trader 1	7/7/2020	SPY Equity	B	50	
Trader 2	7/8/2020	SPY Equity	B	80	
Trader 2	7/9/2020	HYG Equity	B		40%
Trader 1	7/9/2020	SPY Equity	S	10	
Trader 2	7/11/2020	SPY Equity	B	70	
Trader 2	7/12/2020	QQQ Equity	S	100	
Trader 1	7/13/2020	QQQ Equity	S	80	
Trader 2	7/14/2020	QQQ Equity	B		10%
Trader 1	7/15/2020	HYG Equity	B	30	
Trader 2	7/15/2020	QQQ Equity	B	30	
Trader 1	7/17/2020	QQQ Equity	B		20%
Trader 2	7/18/2020	HYG Equity	S	110	
Trader 1	7/19/2020	QQQ Equity	B		60%
Trader 2	7/20/2020	HYG Equity	S	50	