Objetivo

O exercício tem como objetivo implementar uma matching engine simplificada para um único ativo, capaz de receber, armazenar e casar ordens de compra e venda com prioridade preço–tempo, suportando ordens limit, market e pegged, além de operações básicas de cancelamento, modificação e visualização do livro.

Organização do código

O código foi organizado em um pequeno conjunto de módulos dentro de src, separando entre interface de linha de comando, manutenção do estado do livro de ofertas e lógica para ordens limit, market e pegged. Toda a estrutura do repositório, bem como os arquivos-fonte comentados, está disponível publicamente no GitHub.

Complexidade dos algoritmos envolvidos

De forma geral, o sistema não utiliza laços aninhados sobre o livro de ofertas: inserção, cancelamento, modificação e matching percorrem no máximo uma vez a lista de buys ou sells, mantendo essas operações em O(n), onde n é o número de ordens ativas de um lado do book.

A exceção são as rotinas de atualização das ordens pegged, pegged.update_pegged_to_bid e pegged.update_pegged_to_offer. Quando o melhor bid/offer muda, essas funções varrem toda a lista para atualizar o preço de cada ordem pegged e, em seguida, chamam list.sort para reordenar o lado inteiro do livro. Como várias pegged podem “pular” de posição ao mesmo tempo e não há informação prévia sobre onde cada uma deve cair, não é possível corrigir a ordem apenas com ajustes locais; é preciso tratar a lista como potencialmente desordenada e aplicar uma ordenação completa, com custo O(n log n). Pelo teorema do limite inferior para ordenação por comparação, qualquer algoritmo baseado apenas em comparações requer Ω(n log n) comparações para ordenar n itens, de modo que não é possível obter complexidade assintótica melhor que O(n log n), o que justifica o custo das rotinas de atualização das ordens pegged.

Tabela de Complexidade das Funções

Função	Complexidade média	Pior caso
OrderBook.print_book	O(n)	O(n)
OrderBook.best_bid / best_offer	O(n)	O(n)
limit.add_buy_limit / add_sell_limit	O(n)	O(n)
limit.match_limit_buy / match_limit_sell	O(n)	O(n)
limit.cancel_order	O(n)	O(n)
limit.modify_order	O(n)	O(n)
market.match_market_buy / match_market_sell	O(n)	O(n)
pegged.create_pegged_bid_buy / create_pegged_offer_sell	O(n)	O(n)
pegged.modify_pegged_qty	O(n)	O(n)
pegged.update_pegged_to_bid / update_pegged_to_offer	O(n log n)	O(n log n)

Regras de negociação

As regras de comportamento das ordens foram inspiradas e simplificadas a partir do rulebook público da Nasdaq para o The Nasdaq Stock Market, especialmente em relação à prioridade preço–tempo e à semântica de ordens a mercado e limite. Ordens de compra são ordenadas do maior para o menor preço e, dentro do mesmo preço, pela ordem de chegada; ordens de venda seguem a mesma lógica, mas do menor para o maior preço. Ordens market são sempre agressivas, executam imediatamente até consumir a quantidade ou a liquidez disponível e não deixam saldo passivo no livro; ordens limit podem ser agressivas ou passivas, e alterações de preço ou aumento de quantidade fazem a ordem perder prioridade ao ser reinserida, enquanto reduções de quantidade preservam sua posição.

As ordens pegged de compra seguem o melhor bid não pegged e as de venda seguem o melhor offer não pegged, com o motor atualizando o preço dessas ordens e reordenando o livro sempre que há mudança no melhor bid ou offer. A prioridade é mantida de forma que, no mesmo nível de preço, as ordens limit fiquem à frente das pegged, e, caso deixe de existir bid ou offer de referência, as ordens pegged atreladas àquele lado sejam canceladas para evitar ordens indexadas a um preço inexistente.

Bibliografia

NASDAQ. Rulebook – The Nasdaq Stock Market. Nasdaq Listing Center, 2025. Disponível em: https://listingcenter.nasdaq.com/rulebook/nasdaq/rules.

JOSÉ VICTOR L. DE CASTRO. Repositório “MorganStanleyEP”. GitHub, 2025. Disponível em: https://github.com/JoseVictorLDC/MorganStanleyEP/tree/main.