matching_engine

Como funciona

Executar o arquivo main.py para rodar o programa.

Após iniciar a seguinte mensagem irá aparecer

```
 Enter a numer or write the order
1- to exit:
2- to read a file:
3- to show orders in the book of orders
```

Funções

- É possível entrar diretamente com a operação, basta seguir o seguinte padrão
    - limit side price qty
    - market side price qty
    - change order identificador_4 qty @ price
    - cancel order identificador_4
- É possível ler um arquivo externo digitando *2*, assim o sistema irá executar uma sequência de operações.
- Também é possível verificar as orders do tipo **limit** que já foram cadastradas, mas ainda não foi feito o trade completo, ou seja a quantidade ofertada não chegou a 0. Basta digitar ***3***
- Para fechar o sistema basta digitar *1*, lembrando que neste caso as ordens nos ***order book*** serão perdidas.

Observações

Não foram implementadas validações de dados, ou seja, digitar algo fora do padrão fará o programa parar de funcionar

Nesta versão é possível na instrução de **change order, alterar os campos **qty** ou **price** ou **qty e price**, ao alterar a ordem o identificar é incrementado****

Todos os itens bônus foram implementados

O sistema pode fazer trades parciais, ou seja, enquanto tiver ordens disponíveis (e os preços forem iguais -para limit order apenas), ele continua o trade até que a quantidade solicitada seja zerada

Exemplos

1 - alterar ordem

```
 >>>limit buy 10 100
Order created: buy 100 @ 10 identificador_2

>>>change order identificador_2 50 @ 2
Order changed: buy 50 @ 2 identificador_4

>>>change order identificador_4 50
Order changed: buy 50 @ 2 identificador_6

>>>change order identificador_6 @ 10
Order changed: buy 50 @ 10 identificador_8
```

2 - alterar ordem e cancelar ordens com IDs inválidos

```
 >>>limit buy 10 100
Order created: buy 100 @ 10 identificador_2

>>>change order identificador_2 50 @ 2
Order changed: buy 50 @ 2 identificador_4

>>>change order identificador_2 @ 10
Invalid ID or Order is not in book more

>>>cancel order identificador_2
Invalid ID or Order is not in book more
```

3 - cancelar uma ordem

```
 >>>limit buy 10 100
Order created: buy 100 @ 10 identificador_2

>>>change order identificador_2 50 @ 2
Order changed: buy 50 @ 2 identificador_4

>>>change order identificador_4 50
Order changed: buy 50 @ 2 identificador_6

>>>cancel order identificador_6
Order cancelled
```

4 - Operação completa

```
 >>>limit buy 10 100
Order created: buy 100 @ 10 identificador_2

>>>limit sell 10 100
Order created: sell 100 @ 10 identificador_1
Trade, price: 10, qty: 100

>>>limit sell 20 200
Order created: sell 200 @ 20 identificador_3

>>>market buy 150
Order created: market buy 150
Trade, price: 20, qty: 150

>>>market buy 200
Order created: market buy 200
Trade, price: 20, qty: 50

>>>market sell 200
No orders to trade

>>>limit buy 10 100
Order created: buy 100 @ 10 identificador_4

>>>change order identificador_2 50 @ 2
Invalid ID or Order is not in book more

>>>change order identificador_4 50 @ 2
Order changed: buy 50 @ 2 identificador_6

>>>cancel order identificador_4
Invalid ID or Order is not in book more

>>>cancel order identificador_6
Order cancelled
```

Descrição do problema:

#

Uma ordem, no mercado financeiro, significa uma manifestação de interesse de compra ou venda de um determinado ativo. Para fins de simplicidade, iremos trabalhar com dois tipos de ordem: uma ordem à mercado (ou Market Order) e ordem limite (ou Limit Order).

Uma Matching Engine (ou Order Matching System) é um sistema desenvolvido para cruzar ordens em uma exchange de forma rápida e justa. O seu objetivo é estruturar uma matching engine simples, de acordo com algumas premissas:

1- A engine trabalhará com apenas 1 ativo

2- As ordens possíveis são limit (uma ordem passiva colocada a um preço fixo) e market (uma ordem que deve ser preenchida no melhor preço disponível imediatamente)

3- Não é necessário ter armazenamento perene das ordens e trades, todas as informações podem ser mantidas em memória

volátil

4- Não é necessário pensar em escalabilidade de hardware e ferramentas de nuvem e elasticidade (como Kubernetes)

O objetivo é apenas desenhar um projeto de software utilizando formas eficientes de estruturação de dados, algoritmos e engenharia de software. Os algoritmos devem ter, quando possível, ordem N. O software pode ser escrito em forma estrutural ou orientada à objetos, no entanto, lembre-se que a aplicação deve ser escalável, ou seja, evitar repetições de código.

Deve ser possível inserir ordens com as informações:

```
 - Tipo (limit/market)
- Side (buy/sell)
- Price (quando order for limit)
- Qty
```

Limit orders com preços que gerariam trades podem ser ignoradas ou preenchidas, porém o comportamento escolhido deve ser justificado.

Quando um trade for realizado, deve-se mostrar na saída:

"Trade, price:

<

preço do trade>, qty: <número de shares>"

Exemplos de entrada e saída:

```
         >>> limit buy 10 100
        >>> limit sell 20 100
        >>> limit sell 20 200
        >>> market buy 150
        - Trade, price: 20, qty: 100
        - Trade, price: 20, qty: 100
        >>> market buy 200
        - Trade, price: 20, qty: 100
        >>> market sell 200
        Trade, price: 10, qty: 100
```

Bônus:

- Soluções que respeitem a ordem de chegada das ordens. No exemplo anterior, isso significa que a primeira ordem de venda com quantidade 100 deve ser preenchida antes da segunda, com quantidade 200;
- Implementação de cancelamento. Uma ordem ao ser cancelada, deve ser retirada do match engine. Exemplo:

  ```
           >>> limit buy 10 100

          Order created: buy 100 @ 10 identificador_1

          >>> cancel order identificador_1

          Order cancelled
  ```

- Implementação de alteração de ordem. Uma ordem alterada tem seu preço, quantidade ou ambos modificados. Caso tenha implementado o primeiro item de bönus, lembre-se que a ordem com alteração de preço deve ser recolocada para a faixa de preço adequado. Em um livro hipotético:

Ordens de Compra	Ordens de Venda
200 @ 10	100 @ 10.5
	100 @ 9.99 | -

Ao alterar a primeira ordem de compra (200 quantidades ao preço R$ 10,00) para um preço de 9.98, devemos ter a seguinte configuração do livro:

Ordens de Compra	Ordens de Venda
100 @ 9.99	100 @ 10.5
	200 @ 9.98 | -