09

Colocando em prática

Mais um fim de capítulo, mais um Colocando em prática! Lembrando que este exercício é para você realizar as mudanças no projeto feitas pelo instrutor no vídeo, para que você tenha um projeto equivalente e possa aprender na prática! Mas em caso de qualquer dúvida, não deixe de consultar a explicação ou ao vídeo. Vamos lá:

Seguem as dicas e linhas gerais:

1- Inicie criando a classe `Candlestick` no pacote `br.com.alura.modelo` . Ela deve ter os seguintes atributos: **abertura**,**fechamento**, **minimo**, **maximo**, **volume** e **data**. Seus atributos devem ser privados e faça uma reflexão sobre os tipos de cada um. Esta classe também deve ser imutável. Você se lembra o que fizemos para a classe `Negociacao` ser imútavel? Utilize estas mesmas medidas aqui! Não se esqueça de adicionar os *getters* e o **construtor** também.

2- Com as classes `Candlestick` e `Negociacao` criadas, vamos começar adicionando alguns métodos às nossas classes. Primeiramente, na classe `Negociacao` implemente o método `getVolume()` , que deve retornar o volume daquela Negociação. Agora na classe `Candlestick` implemente os métodos `isAlta()` e `isBaixa()` . O método `isAlta()` deve retornar um **boolean**, que indica se aquele é um Candlestick de alta, ou seja , se o valor do seu fechamento é **maior** que o valor da sua abertura. O método `isBaixa()` é o oposto ao `isAlta()` , retornando também um boolean, mas este indicando se .

3- Agora vamos começar a implementar o algoritmo responsável por criar um `Candlestick` a partir de uma lisa de negociações. Vamos utlizar o design pattern (padrão de projeto) **Factory** para isto, que será a classe responsável por gerar os nossos Candlesticks. Crie a classe `CandlestickFactory` no pacote `br.com.caelum.argentum.modelo` . Nesta classe, crie o método `constroiCandleParaData` , que deve receber como argumento uma **lista de negociações** e uma **data** e deve retornar um **Candlestick**. Vamos partir para a implementação no próximo passo.

4- Para a implementação, o seu algoritmo deve conseguir construir um Candlestick a partir de uma lista de negociações seguindo as seguintes regras:

- **Abertura**: preço da primeira negociação do dia;
- **Fechamento**: preço da última negociação do dia;
- **Mínimo**: preço da negociação mais barata do dia;
- **Máximo**: preço da negociação mais cara do dia;
- **Volume**: quantidade de dinheiro que passou em todas as negociações nesse dia;
- **Data**: a qual dia o resumo se refere.

Aconselho a você começar a implentação a partir das coisas **mais simples** de se obter, como por exemplo a **data**, que já recebemos no argumento do próprio método. **Abertura** e **fechamento** também são bem fáceis, visto que basta extrair estes valores das primeiras e últimas posições da lista. Em seguida obtenha o **volume** do Candlestick, que é a soma do volume individual de todas as negociações do dia, que vai requerer que você percorra a lista. Por último, obtenha o **maximo** e **minimo**, que você pode se aproveitar do loop que gera o volume para buscar o preço da maior e da menor negociação do dia respectivamente.

Ficou com alguma dúvida na implementação ou no racíocinio ? Consulte a explicação ou o vídeo para saná-la! Não é vergonha nenhuma recorrer ao material, só não vale copiar direto de lá sem tentar antes :)