

Colocando em prática: isMesmoDia

Hora de colocar em prática as funcionalidades do capítulo. Como sempre, segue abaixo um pequeno guia para você se orientar. Objetivo é prepara o nosso modelo para separar uma lista de negociações de dias diferentes em uma lista de candles. Vamos começar com nossa `Negociacao`. Mãos à obra!

1) Comece escrevendo um teste na classe `NegociacaoTest`. Este método deve identificar negociações de um mesmo dia. Ainda lembra do método `isMesmoDia`?

```
@Test
public void mesmoSegundoEhDoMesmoDia() {

    LocalDateTime agora = LocalDateTime.now();
    LocalDateTime mesmoMomento = agora;

    Negociacao negociacao = new Negociacao(100.0, 20, hoje);

    Assert.assertTrue(negociacao.isMesmoDia(agora));
}
```

2) Teste datas com horários diferentes e no mesmo dia, novamente com `Assert.assertTrue`:

```
@Test
public void comHorariosDiferentesEhNoMesmoDia() {

    LocalDateTime manha = LocalDateTime.of(2016, 02, 25, 8, 30);
    LocalDateTime tarde = LocalDateTime.of(2016, 02, 25, 15, 30);

    // seu teste aqui
}
```

3) Crie o `mesmoDiaMasMesesDiferentesNaoSaoDoMesmoDia` na classe de testes `NegociacaoTest`, veja o teste falhar e, então, implemente o necessário para que ele passe. Note que, dessa vez, o valor esperado é o `false` e, portanto, utilize o `Assert.assertFalse`:

```
@Test
public void mesmoDiaMasMesesDiferentesNaoSaoDoMesmoDia() {

    LocalDateTime manha = LocalDateTime.of(2016, 02, 25, 8, 30);
    LocalDateTime tarde = LocalDateTime.of(2016, 03, 25, 8, 30);

    // seu teste aqui

}
```

3) Crie um teste para verificar o dia e mes é o mesmo, mas anos diferentes:

```
@Test
public void anosDiferentesNaoEhMesmoDia() {

    LocalDateTime manha = LocalDateTime.of(2016, 02, 25, 8, 30);
    LocalDateTime tarde = LocalDateTime.of(2017, 02, 25, 8, 30);

    // seu teste aqui
}
```

4) Agora já temos 4 testes escritos que nem compilam, razão suficiente para implementar o método `isMesmoDia` na classe `Negociacao`.

Faça que todos os testes passem!