Ajudinha para segunda prova de LP2, estou fazendo nas pressas mas acho que vai ser melhor para entender o texto e para abstrair a lógica para a prova. Como vai ser na pressa, desconsiderem qualquer erro de concordância e informalidade, ok? (Tô fingindo que estou falando comigo).

0 -> OBSERVAÇÃO IMPORTANTE: no fim do texto fala sobre onde dar o `throws` (se liguem para pegar ponto aqui dos testes, mesmo que não faça nenhum teste, isso vai ajudar). Então vejam logo como tratar as exceptions.

Lembrar de sempre checar se existe algo que vocês estão procurando.

Você deve lançar exceção quando houver uma tentativa de cadastrar usuário com CPF ou nome `null` ou vazio.

OMG, faz o quê aqui? -> Com um `if`, tu compara o `.isBlank()`, qualquer parâmetro ou `null` (pode fazer igual ao Lab5 e fazer uma função auxiliar booleana, vou mostrar do mais básico).

Exemplo:

```
ConstrutorUsuario(String x, String y, String z) {
   if (x.isBlank() || y.isBlank() || z.isBlank()) {
      throw new IllegalArgumentException();
   }
   if (x == null || y == null || z == null) {
      throw new IllegalArgumentException();
   }
}
```

1. Neste clube, os clientes da loja podem se cadastrar, acumular cupons e aplicar cupons sobre as compras. Os cupons podem ser de diferentes tipos e oferecem diferentes benefícios, como frete grátis, desconto fixo ou progressivo.

Sempre vai ter a especificação do sistema no começo para dar uma luz sobre o funcionamento. Aqui dá para ver que vai ter cadastro, acúmulo (vão ser armazenados), mudança de comportamento de algo em algo (cupom em compra). *Sempre confie em você, então crie uma classe de COMPRA*. Dá para ver que também existem TIPOS diferentes de cupom (dica de que vai ser interface ou herança).

2. Para ter acesso aos descontos, é preciso ser usuário cadastrado no XaBuMDescontos

Temos que ter usuário NO SISTEMA CADASTRADO LÁ, ou seja, uma classe (nice).

3. Um usuário é identificado unicamente pelo seu CPF (exemplo: 111.222.333-00) e também precisa da informação do nome.

CPF e nome vão ser atributos de usuário.

O texto aqui diz o comportamento de identificação ÚNICA (é só um parâmetro). Armazenar eles num `HashMap` ou `HashSet` pode fazer sentido (prefiro `HashMap`). Como é o identificador dele, a gente já sabe que o `hashCode` e `equals` são com ele.

4. A representação textual de um usuário é da forma:

Lívia Campos - 111.222.333-00

LITERALMENTE tá mostrando o:

toString() { return this.nome + " - " + this.cpf; }

5. Usuário realiza uma compra na loja[...] gera um tipo de resumo desta compra e envia para o sistema do clube de descontos a fim de aplicar cupons do usuário sobre esta compra.[...] Esse resumo da compra inclui o código da compra (numérico), o CPF do usuário, o valor total da compra (R$) e o valor do frete (R$).

Na classe compra você armazena os valores para aplicar nela.

Atributos dela:

- int codigo
- String usuarioCPF
- double valorTotal
- double valorFrete

6. Ao aplicar um cupom sobre a compra, os seus valores de compra ou de frete poderão ser alterados, de acordo com o tipo do cupom.

Aqui dá para saber que o cupom muda o comportamento da compra de acordo com o tipo do cupom.

7. A ideia é que será solicitado ao usuário para aplicar um dos seus cupons sobre uma determinada compra.

Ou seja, a RESPONSABILIDADE é do USUÁRIO pegar um CUPOM que pertence a ELE e aplicar na COMPRA.

NO XABUM:

```
public String aplicarDescontoCompra(Compra compra, int indexCupomUsuario, String cpf){
    // Pega o usuário pelo CPF e chama o método DELE aplica na compra passada
    this.usuarios.get(cpf).aplicaCupom(compra, indexCupomUsuario);
    return "Literalmente qualquer mensagem positiva não diz no texto\n FUNFOU!";
}
```

8. Os cupons do clube podem ser acumulados pelos usuários, de modo que, quando um cupom é atribuído a um usuário, ele é removido do sistema **XaBuMDescontos e passa a ser gerenciado pelo usuário.**

Ou seja, você vai remover aquele cupom do sistema:

```
Cupom x = cuponsSistema.remove(indexCupom);
usuario.addCupom(x);
```

9. Não existem limites para a quantidade de cupons para um usuário[...] possível identificar os cupons do usuário pela ordem em que foram adicionados, ou seja, o cupom 1, 2, 3,…, do usuário X.

Ou seja, é um `ArrayList<Cupom>`, e você manipula ele.

10. Existem, atualmente, 3 tipos de cupons disponíveis: frete grátis, desconto fixo e desconto progressivo.

Aqui a gente sabe que vão ser três tipos de cupons, então vai ter polimorfismo.

11. Todo cupom é capaz de oferecer uma representação textual com informações sobre o mesmo e de ser aplicado sobre uma compra da loja. Ao ser aplicado sobre uma compra, o cupom vai modificar o valor da compra ou o seu valor de frete.

Até agora dá para saber que tem uma assinatura de `toString()` e um método para aplicar um desconto numa compra, certo? Ou seja:

aplicaDesconto(Compra compra) {}

Dentro dele, temos, por efeito colateral, a mudança dos status dele.

12. **Tipo | Atributos | Aplicação do desconto | Representação textual**

Aqui temos os comportamentos de cada TIPO DE CUPOM. Na tabela tem a "cola" de como tem que se comportar. Não temos nenhum atributo em comum, nem existe a criação de cupom, ou seja, tem o `aplicaDesconto()` que é implementado de forma diferente e o `toString` diferente, mas precisamos armazená-los. Logo, faz sentido ser uma **Interface**, e no armazenamento ser `ArrayList<Cupom>` (assim, o `ArrayList` vai aceitar qualquer cupom que implemente a interface), e cada classe implementa a Interface.

13. Sempre que um cupom do usuário é usado, ou seja, o cupom é aplicado a uma determinada compra, este cupom é removido da lista de cupons do usuário.

Isto quer dizer que quando você atribui um cupom a um usuário, você remove ele do `ArrayList` do sistema e adiciona ele com um `.addCupom()` (método do usuário) para adicionar ele no `ArrayList` do USUÁRIO. Assim, é responsabilidade do usuário tratar dos seus cupons e não do sistema.

14. Na listagem dos cupons do usuário deve ser exibido o nome e CPF do usuário e a quantidade de cupons que o usuário já teve e quantos cupons ele tem no momento, da seguinte forma:

Lívia Campos - 111.222.333-00
100 cupons - 20 cupons ativos

Aqui temos a descrição do comportamento do `listarCuponsDoUsuario()`. Temos que armazenar um contador que guarde a informação de quantos cupons foram cadastrados para AQUELE usuário, e para pegar os atuais seria um `.size()` do `ArrayList`.

15. Uma outra funcionalidade do sistema é a possibilidade de listagem de usuários por filtros. Os usuários podem ser listados em ordem alfabética ou em ordem decrescente do número de cupons que o usuário já teve.

Outra coisa que vai ser cobrada: o uso de `Comparable` e `Comparator`. Lembrem que, como vocês estudaram, isto é um bom exercício. Como temos duas opções de listagem e uma é por algo que não é padrão, vamos usar o `Comparator`.