ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
Departamento de Engenharia de Computação e Sistemas Digitais

PCS 3111 - Laboratório de Programação Orientada a Objetos para Engenharia Elétrica

2023

Aula 09 – Programação Defensiva

Atenção

- Código inicial para resolução dos exercícios encontra-se disponível no e-Disciplinas.
- Para a resolução dos exercícios, adicione atributos privados às classes conforme necessário, desde que se mantenham as assinaturas e funcionamento especificados no enunciado.
- Submeta um arquivo comprimido (faça um ".zip" – **não pode ser ".rar"**) colocando apenas os arquivos ".cpp" e ".h". Não crie pastas no "zip".

AVISO: para evitar problemas de compilação no Judge, envie os exercícios à medida que implementá-los!

Exercício 01
Altere o código fornecido da classe **Produto** da seguinte forma:

1. O construtor deve jogar uma exceção do tipo `invalid_argument` (da biblioteca padrão) caso o valor de `preco` seja menor ou igual a zero. O argumento da exceção deve ser `"Preco invalido"`.

Na função `teste1`:

- Tente criar um `Produto` com `preco` -1 (o nome não é relevante);
    - Capture a exceção, mostre a sua mensagem na tela (pule uma linha depois) e, então, delete-a;

Dica: Não se esqueça de incluir o cabeçalho `stdexcept` onde necessário.

Exercício 02
Crie e implemente a exceção `ProdutoIndisponivel` de modo que:

- Seja uma classe filha de `logic_error` (da biblioteca padrão).
- A classe contenha apenas seu destrutor e construtor, o qual recebe como argumento uma mensagem (`string`), como exposto em aula.

Modifique a classe **Produto** fornecida para que o método `getPreco` gere uma exceção do tipo `ProdutoIndisponivel` caso o **Produto** não esteja disponível (`disponivel == false`). O argumento das exceções deve ser "`Produto indisponivel`".

Complete a função `teste2`, para tanto considere os seguintes passos:

1. Crie o `Produto` de nome *Gatorade* e de `preco` 7.50 reais;
2. Imprima o preço do produto criado pulando uma linha no final;
3. Chame o método `setDisponivel` do produto criado passando o parâmetro `false`;
4. Tente chamar o método `getPreco` do produto criado;
    a. Capture a exceção, mostre a sua mensagem na tela (pule uma linha depois) e, então, delete-a;

Exercício 03
Na classe **Pedido**, **defina e** implemente o método `calcularPrecoTotal` de forma que ele retorne o preço total do **Pedido**.

```
double calcularPrecoTotal();
```

O método deve somar os preços dos **Items**, usando, para isso, o método `calcularPrecoTotal` de **Item**. Caso o **Produto** não estiver disponível, desconsidere-o no cálculo do preço total. Para isso, caso ao chamar o método `calcularPrecoTotal` de **Item** aconteça uma exceção `ProdutoIndisponivel` (oriunda do método `getPreco` de **Produto**), capture e destrua esta exceção. Caso o **Pedido** não tenha produtos, retorne 0.

Complete a função `teste3`:

1. Crie um `Pedido` de `quantidadeMaxima` 4;
2. Crie o `Produto` *Cerveja* com preço 4.35 reais;
3. Crie o `Produto Bis` com preço 3.90 reais;
4. Crie o `Produto` *Frigideira* com preço 80.79 reais;
5. Crie o `Produto` *Vassoura* com preço 30.50 reais;
6. Adicione ao `Pedido` os Produtos na ordem em que foram criados com as quantidades 5, 2, 1 e 1, respectivamente;

7. Imprima a saída do método `calcularPrecoTotal` do pedido;
8. Faça com que *Cerveja* e *Frigideira* fiquem indisponíveis através do método `setDisponivel`;
9. Repita o passo 7;
10. Faça com que *Bis* e *Vassoura* fiquem indisponíveis através do método `setDisponivel`;
11. Repita o passo 7.

Testes do Judge

Exercício 1

- Produto: exceções do construtor
- Teste da função `teste1`

Exercício 2

- ProdutoIndisponivel é filha de `logic_error`
- ProdutoIndisponivel: `Produto` indisponível joga `ProdutoIndisponivel`
- Teste da função `teste2`

Exercício 3

- Pedido: `calcularPrecoTotal` com um Pedido vazio;
- Pedido: `calcularPrecoTotal` com todos os produtos disponíveis;
- Pedido: `calcularPrecoTotal` com produto indisponível no começo do vetor;
- Pedido: `calcularPrecoTotal` com produto indisponível no fim do vetor;
- Pedido: `calcularPrecoTotal` com produto indisponível no meio do vetor;
- Pedido: `calcularPrecoTotal` com todos os produtos indisponíveis;
- Teste da função `teste3`