Laboratório de Programação 2 - 2015.2
Prova 2 - 14/04/2016

Turma 1: Lívia Sampaio
Turma 2: Raquel Lopes
Turma 3: Francisco Neto

Roteiro de Implementação

Sua mãe é uma artista e confecciona **pratos decorativos personalizados**. Para ajudar sua mãe, você resolveu criar uma aplicação que gerencia o **estoque de pratos** já decorados e personalizados.

Passo 1:
Você precisa criar **uma entidade para representar um prato personalizado**, que possui um **preço base** e a personalização do prato, que pode ser uma de três opções: **estampa**, **foto** ou uma **pintura feita a mão**. Cada opção de personalização não afeta o comportamento do prato personalizado. Apresente mensagens de erros caso sejam especificados **preço negativo** ou uma **personalização que não seja uma dessas três**.

Além do construtor é necessário saber informar a personalização do prato, o preço, e poder definir uma nova personalização para o prato. Dois pratos são iguais se tiverem a mesma personalização, e a representação em String do prato:

```
"Prato com <personalizacao> custa R$ <preco>."
```

Parte 2:
Com os negócios dando certo, sua mãe expandiu a produção e decidiu confeccionar pratos de diferentes formatos. Agora, existem 3 tipos de pratos personalizados: **triangular**, **retangular** ou **circular**, e ao criar um prato de formato específico devem ser informadas as **dimensões do prato de acordo com o seu formato**. Faça o tratamento de erros com Exception para **dimensões com valores negativos.** São elas:

```
Triangular: base e altura;
Retangular: base e altura;
Circular: raio do círculo;
```

Atualize a representação em string desses pratos para incluir o formato do prato da seguinte forma:

```
"Prato com <personalizacao> custa R$ <preco>. Formato <formato>."
```

Atualize o equals, pois para sua mãe, os pratos são iguais apenas pela personalização. Para isso, use o **instanceof** no seu equals apenas no local do seu código para identificar pratos iguais **independente do formato**.

Passo 3:

A única diferença entre os pratos de diferentes formatos é a forma de calcular o preco. O preço de cada formato é definido pela área necessária para personalizar o prato, de forma que para as três formas é cobrado R$ 0,01 por centímetro quadrado (cm²) personalizado. As fórmulas são:

```
precoTotal = precoBase + (0.01 * areaTotal)
```

Por sua vez, as áreas são calculadas da seguinte forma:

```
Triangular: areaTotal = ( base * altura ) / 2.0;
Retângular: areaTotal = base * altura;
Circular  : areaTotal = pi * raio²; //pi = 3.14
```

Passo 4:
Você deve escrever uma classe chamada **EstoqueDePratos**, que vai manter uma **coleção de pratos personalizados independente do seu formato.** Devido aos pratos possuírem diferentes formatos, essa coleção **deve permitir vários pratos com a mesma personalização.** Além disso deve ser capaz de:

- Criar um novo prato personalizado especificando: formato, dimensões, preço e a personalização;
- Adicionar e remover (retornando boolean) um prato ao estoque;
- Obter o número de pratos no estoque,
- Consultar se o estoque possui um determinado prato com uma personalização especifica;
- Retornar um prato (o primeiro prato que encontrar) com uma personalização de interesse;
- Calcular o total que pode ser vendido dentre os pratos no estoque.

Passo 5:
Sua mãe deseja, automaticamente, **comparar pratos**. Para que isso seja realizado você deve modificar seu código para permitir que **um prato possa ser comparado com outro prato** considerando **o preço deles**. Note que o preço a ser comparado é o preço final do prato considerando seu formato.

Sobre esse passo, responda: (escreva as respostas em comentários na classe PratoPersonalizado)
Com a sua implementação deve ser possível até comparar um prato circular com um prato triangular. Isso é possível? Sendo possível, os pratos serão comparados usando que cálculo de preco?

Passo 6:
Para concluir a implementação do EstoqueDePratos, deve ser adicionado a essa classe o método `getPratosOrdenadosPorPreco` que retorna uma lista de todos os pratos personalizados no estoque ordenados pelo preço;

Dica: Dado que os pratos são comparáveis pelo preço, ordenar uma lista de pratos é trivial, você pode usar recursos de Java para isso, **não precisa criar seu próprio método de ordenação.**

Passo 7:
Indique como comentário no seu código onde estão sendo usados: Polimorfismo, chamada polimórfica, e detalhes de composição (quem é o composite, e onde é feito o forwarding). Lembrem que o composite (também chamado de wrapper) é o objeto dono da composição.

Link para a classe de teste: **`https://goo.gl/JyHzAT`**
Extras: Criação de testes de unidades adicionais; Uso de Factory; Javadoc e Hierarquia de Exceptions
Instruções para entrega estão disponíveis no Canvas.