Centro de Engenharia Elétrica e Informática – CEEI
Unidade Acadêmica de Sistemas e Computação – UASC
Disciplina: Laboratório de Programação 2

Laboratório 06

Como usar esse guia:

- Leia atentamente cada etapa
- Referências bibliográficas incluem:
  - material de referência desenvolvido por professores de p2/lp2 em semestres anteriores (ONLINE)
  - o livro Use a cabeça, Java (LIVRO-UseCabecaJava)
  - o livro Java para Iniciantes (Livro-JavaIniciantes)
- Quadros com dicas tem leitura opcional, use-os conforme achar necessário
- Preste atenção nos trechos marcados como importante (ou com uma exclamação)!

Sumário

Acompanhe o seu aprendizado

Conteúdo sendo exercitado

- Padrões GRASP: ênfase no uso de expert da informação, creator, baixo acoplamento, alta coesão e controladores
- Uso de interfaces: Comparable<> e Comparator<>

- Polimorfismo com interfaces
- Uso de interface com composição como uma forma de ter menor acoplamento e polimorfismo (padrão de projeto strategy)
- Uso de testes de aceitação com easy accept

Objetivos de aprendizagem

- Temos dois objetivos principais nesse lab: 1) ter a experiência de um lab bem parecido com um projeto em termos de estruturação (um conjunto de casos de uso, testes de aceitação, sem interface com usuário, mais funcionalidades); 2) explorar o reuso de software com composição e interfaces, identificando vantagens e desvantagens de cada estratégia.

Perguntas que você deveria saber responder após este lab

- O polimorfismo é uma consequência do reuso com interfaces. Como usamos o polimorfismo no código?
- Como usar o padrão GRASP Controller?
- Considerando as implementações de coleções de Java, em quais situações cada coleção é mais apropriada para ser usada?
- Qual a relação da interface Comparable<> com os mecanismo de ordenação de coleções?
- Qual a relação da interface Comparator<> com os mecanismo de ordenação de coleções?

Para se aprofundar mais...

- Referências bibliográficas incluem:
    - material de referência desenvolvido por professores de p2/lp2 em semestres anteriores (ONLINE)
    - o livro Use a cabeça, Java (LIVRO-UseCabecaJava)
    - o livro Java para Iniciantes (Livro-JavaIniciantes)
- Projetando com interfaces
- Um pouco mais sobre GRASP

Introdução

Neste laboratório queremos vivenciar uma experiência nova. Como evoluir um software OO? Partiremos de um sistema pronto e iremos introduzir uma nova funcionalidade nesse sistema e garantir que o sistema esteja correto após as modificações. Vamos exercitar a leitura (e uso) de código escrito por outra pessoa. Nesse processo, o uso de testes automáticos é essencial.

O sistema que iremos evoluir é o SAGA, bem conhecido nosso, com que trabalhamos no laboratório anterior.

Mantemos a mesma orientação de que o sistema não tem interface com usuário, além disso, vamos continuar usando os testes de aceitação com o easy accept e ainda será requerido o uso de testes de unidade

Você irá baixar o código base do sistema SAGA NESTE repositório online. Definiremos um novo caso de uso a ser implementado no SAGA. Para introduzir a nova funcionalidade no sistema você precisará alterar classes existentes como Facade, Controllers e ainda as classes mais relacionadas ao domínio do problema. Lembre-se ainda que novos testes de unidade deverão ser implementados para verificar as novas funcionalidades implementadas. Os testes de aceitação estão disponíveis AQUI.

Sistema para Auto-Gestão de comércio de Alimentos - SAGA

A comercialização de lanches acontece de maneira auto-gerida em alguns laboratórios de graduação da nossa universidade. Atualmente, vários fornecedores ofertam os seus produtos alimentícios nos laboratórios, mas não permanecem no local para receber o dinheiro da compra. Na verdade, eles deixam um caderninho, onde cada cliente mantém uma página (a chamada **conta**) para anotar as suas compras. Ao retirar um produto, o cliente anota a data, quantidade e o nome do produto na página de sua **conta**, no caderninho do fornecedor. Periodicamente, os clientes realizam o pagamento de suas **contas**. Essa dinâmica de comercialização, apesar de funcionar em termos gerais, dificulta o controle dos processos de venda de produtos e pagamento. Diante disto, surgiu a necessidade de informatizar este processo.

A atividade proposta por este Lab é implementar um sistema de comércio eletrônico para dar suporte à comercialização de lanches nos laboratórios. As necessidades de fornecedores e clientes motivou o surgimento do SAGA - Sistema para Auto-Gestão de comércio de Alimentos.

Neste sistema, é possível cadastrar os clientes, fornecedores dos lanches e os produtos oferecidos por eles. Através do sistema, é possível manter o registro das contas e compras dos clientes junto aos seus fornecedores. Além disso, do ponto de vista do administrador, será possível fazer consultas sobre clientes, fornecedores, produtos e compras de um determinado cliente.

O sistema SAGA que servirá como base para este laboratório implementa os seguintes casos de uso (os mesmos que foram desenvolvidos no laboratório anterior): (1) CRUD de Clientes; (2) CRUD de Fornecedores; (3) CRUD de Produtos de fornecedores; (4). Ordenação da listagem de clientes, fornecedores e produtos; (5) Cadastro de compra em uma conta; e (6) Listagem de compras de um cliente.

A nova funcionalidade a ser implementada diz respeito à representação de produtos, introduzindo a ideia de combo de produtos.

US7. Cadastrar combo de produtos

User story: Como administrador do sistema, quero adicionar produtos que são um combo.

Os combos de produtos são usados para oferecer descontos na compra dos lanches. Um combo pode ser entendido como um produto composto por mais de um produto com um preço especial. Exemplo: A coxinha e o suco custam, individualmente, R$3,00 cada. No combo "Coxinha + Suco", o cliente pagará R$4,50.
Seu sistema deve ser capaz de lidar com esta situação. Para tanto, você pode especializar o Produto de modo a permitir que possam existir dois tipos de produto: simples e combo. Por simplicidade, um combo é formado por dois ou mais produtos simples. Não pode ser formado por outros combos.
Os combos, assim como os Produtos, tem nome, descrição e preço. O **nome** e a **descrição** serão passados pelo usuário. O **preço** do combo é calculado de acordo com um **fator** de promoção que deve ser informado. Este valor é um fator percentual, um número decimal maior que zero e menor que 1. **Exemplo**: Fator 0,25, refere-se ao desconto que os produtos terão em seu preço. Para calcular o preço do combo, somam-se os preços de todos os produtos e aplica-se o desconto. Para este exemplo, usamos o **fator** 0,25. Exemplo: Coxinha - R$3,00 e Suco R$3,00. No combo, Coxinha + Suco R$ 4,50.

Considerando o combo de produtos, a Facade do sistema deve ter métodos para cadastrar, dar a representação textual, editar e remover o combo de produtos para um dado fornecedor. É importante lembrar que nessas operações o fornecedor é identificado pelo seu **nome**.

- Cadastrar combo para um fornecedor; Nesta operação, deve ser indicado o fornecedor, os produtos que fazem parte do combo (identificados por nome e descrição) além do nome, descrição e fator de promoção do combo. Não é preciso informar o preço, pois este atributo é derivado, calculado de acordo com os preços e o fator.
  EXEMPLO1: <nome_fornecedor, nome_combo, descricao_combo, fator, (prod1, prod2)>
  `Dona Ines, Lanche FIT, Tapioca com Salada, 0.25, (Tapioca simples - Tapioca com manteiga, Salada - Salada de frutas)`
  EXEMPLO2:
  `Josenilda, Promo 1, Bolachinha de cebola + Mousse de limão, 0.10, (Biscoito Salgado - Bolachinha de cebola, Sobremesa - Mousse de limão)`

  OBS.: Para viabilizar o teste automático, há apenas um parâmetro para informar os produtos que fazem parte do combo. Ele deve ser do tipo String. Os produtos devem estar separados por vírgulas. Os parênteses que aparecem no exemplo acima (prod1, prod2), são para facilitar o entendimento do texto. Não precisam ser usados! Veja novamente a representação textual mostrada nos exemplos.

- Retornar a representação textual de um produto combo de um fornecedor. Note que a representação textual dos produtos, sejam eles simples ou combo, é a mesma. No exemplo abaixo representamos um combo;
  EXEMPLO <nome_fornecedor, nome_prod, descrição_prod><preço>:
  "Lanche FIT - Tapioca com Salada - R$4,50"

- Retornar a representação textual de todos os produtos de um fornecedor. Esta operação também funciona da mesma forma para produtos combo e simples. Use o separador | para separar os dados retornados, em ordem alfabética do nome do produto.
  EXEMPLO:
  "Dona Inês - Lanche FIT - Tapioca com Salada - R$4,50 | Dona Inês - Salada - Salada de frutas - R$3,00 | Dona Inês - Tapioca simples - Tapioca com manteiga - R$3,00"cd

- Retornar a representação textual dos produtos (combos ou simples) de todos os fornecedores cadastrado no sistema (um por linha, em ordem alfabética do nome do fornecedor);
  EXEMPLO:
  "Dona Inês - Lanche FIT - Tapioca com Salada - R$4,50 - Dona Inês - Tapioca simples - Tapioca com manteiga - R$3,00 | Dona Inês - Salada - Salada de Frutas - R$3,00 | Josenilda - Biscoito Salgado - Bolachinha de cebola - R$3,00 | Josenilda - Promo 1 - Bolachinha de cebola + Mousse de limão - R$6,30 | Josenilda - Sobremesa - Mousse de limão - R$4,00 "

- Editar produto combo. Só é possível alterar o fator promocional. Para outras alterações, sugere-se excluir o produto e inseri-lo novamente;

- Remover produto combo do fornecedor.
  OBS: É importante notar que a remoção de um produto Simples, não interfere na existência de um Combo com este produto. É uma decisão de negócio do fornecedor! Por exemplo: O fornecedor vendia Pão e Sopa individualmente. Após o incremento de vendas obtido com o combo Sopa + Pão, ele desistiu de vender apenas Pão.

Entrega

Evolua o sistema SAGA com a funcionalidade descrita na US7, nesse caso, seu sistema deve estar completo (incluindo as funcionalidades do código base, US1, US2, US3, US4, US5, US6). É importante que o código esteja devidamente documentado.

Ainda, você deve entregar um programa com testes de unidade para as classes com lógica testável. Tente fazer um programa que possa ser devidamente testado e que explore as condições limite e as regras de negócio apresentadas.

A facade do seu sistema deve ser capaz de testar cada uma das US já especificadas com os testes de aceitação definidos pela equipe aqui.

Faça um **zip** da pasta do seu projeto. Coloque o nome do projeto para:

- LAB6_SEU_NOME e o nome do zip para LAB6_SEU_NOME.zip. Exemplo de projeto: LAB6_LIVIA_SAMPAIO.zip. Este zip deve ser submetido pelo Canvas.

Seu programa será avaliado pela corretude e, principalmente, pelo DESIGN do sistema. É importante:

- Usar nomes adequados de variáveis, classes, métodos e parâmetros.
- Fazer um design simples, legível e que funciona. É importante saber, apenas olhando o nome das classes e o nome dos métodos existentes, identificar quem faz o que no código.