Relatório do projeto

DEI – Departamento de Engenharia Informática - FCTUC

Estudantes:

Gonçalo Fernandes Diogo de Almeida, 2020218868

João Bernardo de Jesus Santos, 2020218995

Professor:

Fernando Silva (TP1)

Disciplina:

Programação Orientada aos Objetos

Curso – Ano:

Licenciatura em Engenharia Informática – 2021/2022

Índice

Conteúdo

Introdução

O objetivo deste projeto consiste em desenvolver uma aplicação para gerir compras online de produtos de uma cadeia de supermercados, tal como as suas promoções e clientes associados a cada compra. A aplicação tem de permitir realizar o login, realizar uma compra e consultar as compras realizadas. Deverá também registar, no fim de cada compra, a informação em ficheiros, para uso futuro. Por fim, a interação com o utilizador deverá ser realizada através da consola.

Neste relatório será feita uma breve explicação do uso e da constituição de cada classe como também a apresentação do funcionamento do programa.

Classes

Classes *Promocao*, *PagueMenos* e *Pague3Leve4*

Gerem as promoções aplicadas aos produtos, sendo que as classes *PagueMenos* e *Pague3Leve4* herdam da classe *Promocao* (que é uma classe abstrata, visto que uma promoção tem de obrigatoriamente ser uma dos dois tipos apresentados anteriormente). Cada promoção tem uma data de início e fim, e considera-se que as datas são válidas apenas se a data final for posterior ou igual à inicial.

As promoções são aplicadas através do método *desconto* (método abstrato da classe *Promocao*), que, tendo em conta a promoção aplicada, calcula o desconto a ser aplicado ao produto e à quantidade em questão.

No caso da classe *PagueMenos*, o desconto é aplicado da seguinte forma: a primeira unidade é paga a 100% do preço, sendo que em cada unidade é decrescido o custo em 5% do preço inicial até atingir um preço mínimo de 50%. Por exemplo, num produto que custe 10€, a primeira unidade custa 10€, a segunda 9,50€, … a décima 5,50€, a décima primeira 5€, a décima segunda 5€, e todas as seguintes continuam a custar 5€.

No caso da classe *Pague3Leve4*, os clientes pagam 3 em cada 4 unidades dos produtos. Por exemplo, se um cliente comprar 5 unidades, pagará apenas 4.

Os métodos *toString* também são diferentes nas duas classes, sendo que retornam o seu nome, com vista a ser utilizado na classe *InterfaceUtilizador* para mostrar o seu tipo ao utilizador.

Classe *Dimensao*

Gere as dimensões de um produto de mobiliário. Contém a sua altura, largura e profundidade.

Classe *Item*

Gere os itens do carrinho de uma compra. Contém um produto e a sua quantidade, e também dois métodos para incrementar ou decrementar a quantidade num certo valor (parâmetro dos métodos *incrementarQuantidade* e *decrementarQuantidade*).

Classe *Data*

Gere as datas do programa. Contém um dia, um mês e um ano, e também métodos para verificar a validade de uma data (eValida), comparar duas datas (compareTo), verificar se um ano é bissexto (eBissexto) e obter uma data no formato Dia/Mês/Ano em String (toString).

Classe *Compra*

Gere as compras de um cliente. Contém uma data, uma lista de itens (*carrinho*) e o preço final de uma compra com e sem portes. Contém também métodos para verificar se o carrinho possui um produto (*contemProduto*), para verificar se o stock atual de um produto é suficiente para a quantidade pedida (*stockAtualSuficiente*), para adicionar um item ao carrinho (*adicionarCarrinho*), para remover uma quantidade de um produto no carrinho (*removerCarrinho*), para obter a quantidade de um produto no carrinho (*obterQuantidade*), para obter o preço de envio, tendo em conta o tipo de produtos no carrinho e o tipo de cliente a realizar a compra (*precoDeEnvioTotal*), para obter o preço de uma compra sem portes (*precoSemEnvio*), para remover todos os itens do carrinho (*clear*), para obter os preços finais do carrinho no formato *String* (*obterPrecos)* e para obter o carrinho no formato *String* (*toString*).

Classes *Produto*, *Alimentar*, *Limpeza* e *Mobiliario*

Gerem os produtos da cadeia de supermercados, sendo que as classes *Alimentar*, *Limpeza* e *Mobiliario* herdam da classe *Produto* (que é uma classe abstrata, visto que um produto tem de obrigatoriamente ser um dos três tipos apresentados anteriormente).

Cada produto tem um identificador, um nome, um preço unitário, uma lista de promoções e um stock inicial. E além destes atributos, específicos da classe *Produto*, cada tipo de produto possui ainda os seus próprios atributos, e também modificações nos métodos *precoDeEnvio* e *toString*:

- Os produtos alimentares (classe *Alimentar*) possuem um atributo para o número de calorias por 100g (*calorias*) e outro para a percentagem de gordura (*gordura*);

- Os produtos de limpeza (classe *Limpeza*) possuem um atributo para o grau de toxicidade que pode variar entre 0 e 10 (*toxicidade*);

- Por fim, os produtos de mobiliário (classe *Mobiliario*) possuem um atributo para o peso (*peso*) e outro para a dimensão (*dim*).

O método *precoDeEnvio* devolve 0 exceto se o produto for do tipo mobiliário e o seu peso for superior a 15, nesse caso o método devolve 10. Já o método *toString* é diferente nas 3 classes devido aos seus atributos.

Classes *Cliente*, *Frequente* e *Regular*

Gerem os clientes da cadeia de supermercados, sendo que as classes *Frequente* e *Regular* herdam da classe *Cliente* (que é uma classe abstrata, visto que um cliente tem de obrigatoriamente ser um dos dois tipos apresentados anteriormente). Cada cliente tem um nome, uma morada, um email, um número de telefone, uma data de nascimento e uma lista de compras realizadas.

Além dos atributos da classe *Cliente*, cada tipo de cliente possui o seu próprio método *precoDeEnvio*. Este método, no caso de ser de um cliente frequente, devolve 0 caso o preço total da compra seja superior a 40 (parâmetro do método) e 15, caso contrário; e, no caso de ser um cliente regular, devolve 20.

Classe *GestorFicheiros*

Gere o registo e a leitura dos dados utilizados no programa. Contém 5 constantes relativas ao caminho e nome dos ficheiros utilizados (obtidas no construtor da classe), e métodos para interagir com os ficheiros:

- Método *obterClientes*: Lê os clientes contidos no ficheiro de texto cujo nome está definido na constante *fClientes*. Cada linha do ficheiro contém um cliente que está no formato “nome;morada;email;telefone;data de nascimento;tipo de cliente(regular/frequente)”.

- Método *obterProdutos*: Lê os produtos contidos no ficheiro de texto cujo nome está definido na constante *fProdutos*. Cada linha do ficheiro contém um produto que está num de 3 formatos: (1) “alimentar;identificador;nome;preço unitário;calorias;gordura;stock inicial”, (2) “limpeza;identificador;nome;preço unitário;toxicidade;stock inicial” ou (3) “mobiliario;identificador;nome;preço unitário;peso;dimensão (altura;largura;profundidade);stock inicial”.

- Método *obterPromocoes*: Lê os produtos contidos no ficheiro de texto cujo nome está definido na constante *fPromocoes*. Cada linha do ficheiro contém uma promoção que está no formato “tipo de promoção;produto;data inicial (dia/mês/ano); data final (dia/mês/ano)”.

- Métodos *escreverCadSup* e *lerCadSup*: Escreve e lê, respetivamente, uma instância da classe *CadeiaSupermercados* num ficheiro de objetos cujo nome está definido na constante *fCadSup*. Todas as classes cujos dados estejam registados na classe CadeiaSupermercados têm de conter implementar a classe Serializable para poderem ser escritas e lidas, à exceção das que herdam de outras que já o tenham (Ex.: A classe *Alimentar* não necessita de implementar pois herda da classe *Produto*).

- Método *lerFichObj*: Verifica se o ficheiro de objetos cujo nome está definido na constante *fCadSup* já existe. O objetivo deste método, é, implicitamente, verificar se já foram registados dados de compras anteriores, visto que, em caso negativo, este ficheiro não foi criado.

Classe *InterfaceUtilizador*

Gere a interação com o utilizador através da consola. A cada instância desta classe corresponde uma instância da classe *CadeiaSupermercados*, *Scanner* e *GestorFicheiros*.

Os métodos *readString* e *readIntProtection* leem, respetivamente, uma *String* e um inteiro positivo introduzido pelo utilizador. Foram criados com o objetivo de resolver casos de inputs inválidos que perturbem o funcionamento do programa, como por exemplo, em casos que o método *nextInt* não receba um inteiro, o que pode causar um “loop” nos menus apresentados ao utilizador.

Esta classe contém também métodos para apresentar menus ao utilizador, cuja utilidade e ligação serão apresentados posteriormente.

Classe *CadeiaSupermercados*

Gere a lista de clientes e produtos da cadeia de supermercados. Contém como atributos as listas referidas anteriormente e contém também métodos para verificar se um cliente possui um email igual à *String* passada como parâmetro (*contemEmail*), para obter um produto que possua nome igual à *String* passada como parâmetro (*obterProduto*), para obter uma lista de compras realizadas por todos os clientes (*obterCompras*) e para obter um catálogo dos produtos com as suas descrições no formato String (obterCatalogo).

Funcionamento do programa

Ao iniciar o programa, começa-se por criar uma instância (*gf*) da classe *GestorFicheiros*, para depois se interagir com ficheiros através desta, e uma instância *(cad*) da classe *CadeiaSupermercados*. De seguida, utiliza-se o método *lerFichObj* de *gf*, que verifica se o ficheiro *cadSup.obj* existe, o que nos permite saber se já foram registados dados de compras ou não. Em caso afirmativo, leem-se as informações do ficheiro *cadSup.obj*; caso contrário, leem-se os ficheiros dos clientes, produtos e promoções, cujos nomes e formatos foram referidos anteriormente, através dos métodos de *gf*. Guardam-se, depois, as informações em *cad*.

Com *cad* já pronta, cria-se uma instância da classe *InterfaceUtilizador*, registando *gf* e *cad* nesta, e, com a mesma, chama-se o método *menu*. Este método irá começar por mostrar o menu de início de sessão ao utilizador, podendo este escolher entre iniciar sessão e sair do programa. Se este escolher iniciar sessão, é pedido o seu email: caso seja inválido, é mostrada uma mensagem que o informe; caso seja válido, pede-se a data do dia atual através do método *readData* (se o utilizador introduzir uma data inválida, é-lhe pedido de novo a data).

Seguidamente, é apresentado um menu com 4 opções: realizar uma compra, consultar as compras realizadas (mostra as compras realizadas pelo cliente até à data atual, através do método *imprimirComprasRealizadas*), mudar a data atual (através do método *readData*) e terminar sessão (volta para o menu de início de sessão). No caso de pretender realizar uma compra, será chamado o método *menuCompra*.

Neste método, será mostrado um novo menu com 4 opções: ver catálogo e comprar, ver carrinho, fazer pagamento e fechar o menu de compra (fecha a compra e volta para o menu inicial).

No primeiro caso, é chamado o método *realizarCompra*, que mostra o catálogo (produtos e suas descrições) da cadeia de supermercados e permite ao utilizador escolher o produto e a quantidade que deseja adicionar ao carrinho através do método *adicionarItemCarrinho*, atualizando o carrinho e o stock do produto em causa.

No segundo caso, chama-se o método *verCarrinho*, que mostra os produtos adicionados ao carrinho, bem como os preços individuais, preço total sem por-

tes e preço total com portes (os últimos dois já incluem possíveis promoções aplicadas aos produtos na data da compra), e permite ainda ao utilizador remover uma quantidade de um produto do carrinho, remover todos os produtos do carrinho ou voltar ao menu de compra.

No terceiro caso, faz-se o pagamento, pedindo ao utilizador para verificar os conteúdos da sua compra, através do método *confirmarCompra*, podendo, se pretender, voltar para o menu da compra. Caso este confirme a compra, é mostrada uma mensagem a informar o utilizador que a sua encomenda será enviada para a sua morada (método *printFinal*) e, voltando ao método *menuCompra*, atualiza-se o ficheiro *cadSup.obj* (ou cria-se, caso não exista), através do método *escreverCadSup* de *gf*.

Conclusão

Para o desenvolvimento da aplicação aplicámos conceitos como classe, *String*, herança, polimorfismo e ficheiros. Usámos classes e *strings* em todo o projeto, mas *strings* especialmente no método *toString* de cada classe; herança e polimorfismo nas classes *Alimentar*, *Limpeza* e *Mobiliario*, que herdam da classe *Produto* os seus métodos e atributos, nas classes *PagueMenos* e *Pague3Leve4*, que herdam da classe *Promocao*, e também nas classes *Frequente* e *Regular*, que herdam da classe *Cliente*; por fim, usámos o conceito de ficheiros na leitura e escrita de informação acerca dos clientes, promoções e produtos da cadeia de supermercados.