Documento Explicativo

Índice

1.Objetivo geral

Este trabalho tem como intuito a implementação, na linguagem Java, de um sistema de passagens aéreas utilizando os conceitos de Programação Orientada a Objetos (POO).

2.Main

As considerações iniciais quanto a implementação do código no *Main* são referentes à distinção de objetos a serem instanciados sem nenhuma entrada do usuário sendo um exemplo deste os voos que a empresa fictícia tem disponíveis; de objetos que seriam instanciados após alguma entrada do usuário, exemplificando isso a instanciação de adultos e crianças; e quanto a objetos que seriam instanciados somente após alguma entrada do usuário, porém também teriam certos aspectos pré-definidos que não dependeram de entrada alguma do usuário, sendo retrato disso a criação do cartão de crédito ou débito que tem atributos que dependem de entradas do usuário ao mesmo tempo que tem atributos que não dependem destas.

Além disso, é nessa classe onde está presente o algoritmo de pesquisa que o usuário irá utilizar para encontrar os voos que lhe são relevantes. Tendo isso em vista, foram utilizados alguns arraylists para o eficiente armazenamento das informações geradas pelo programa.

A seleção dos voos ocorre pelo fornecimento de informações do usuário depois que ele vê os voos que atendem às especificações iniciais. Logo em seguida o usuário deverá informar a quantidade de passagens e o tipo delas(adulto ou criança), assim como nome e o Id relevante (CPF ou Matrícula da Certidão de Nascimento). Após esta fase, ele deverá escolher o assento da passagem através do display no console, e somente após ele selecionar um assento válido ele deverá fornecer as informações do cartão obedecendo às especificações de construção do cartão (data válida, código válido e saldo suficiente).

As passagens então serão adicionadas à compra e o preço total será subtraído do cartão informado. A compra será finalizada , as passagens compras aparecerão no console e o programa será finalizado.

3.Voo

A classe voo diz respeito a onde boa parte das informações não provenientes do cliente serão armazenadas: preço, origem, destino, data de partida, data de volta, assentos disponíveis etc. Sendo assim, essa classe é a que menos usa métodos de outras classes à medida que é a classe da qual os métodos são mais utilizados e, por essa razão, boa parte dos atributos é inicializada no método construtor da classe. Tendo isso em vista, os métodos dessa classe objetivam ser facilmente utilizados para que seus atributos possam ser acessados sem dificuldade, entre esses métodos, temos:

- Métodos get de todos os atributos da classe
- Um método para ocupar um assento no voo baseado num argumento fornecido pelo usuário.

4.Passagem

A classe *Passagem* tem a função de armazenar e imprimir atributos relevantes de uma pessoa, seu assento, e o voo em que estará embarcando. Visto que sua finalidade é tanto orientar o cliente ao embarcar, quanto comprovar sua identidade, a passagem impressa apresenta as seguintes informações:

- Nome da empresa aérea (Venti Airlines);
- Nome do cliente;
- Data do voo;
- Horário do voo;
- Local de origem;
- Local de chegada;
- Assento do cliente;

Duas passagens são impressas em conjunto, a de ida e a de volta, vez que nossa empresa aérea apenas trabalha com voos de ida e volta.

4.1.Passagem de Criança

Passagem de Criança (*PassagemDeCrianca*) se diferencia de passagem em sua impressão, que a identifica como uma passagem infantil, e em apresentar um desconto que é aplicado caso o dono da passagem tenha menos de seis anos de idade.

5.Compra

A classe Compra objetiva armazenar as informações relevantes à compra de passagens pelo usuário. A abstração da classe a dividiu em atributos sendo:

- *voucherCompra*: é um número gerado aleatoriamente que representa a compra, uma espécie código de recibo.
- *gerador*: um objeto da classe *Random* que gera o código do *voucherCompra.*
- *passagens:* uma *ArrayList* que armazena objetos da classe *Passagem* para referência nos momentos de obter o preço da compra e a quantidade de passagens compradas

A compra é munida de métodos que executam as ações essenciais para sua execução, sendo estes métodos para a visualização final das passagens compras, obtenção do preço final e da quantidade de passagens compras.

As passagens compras foram armazenadas em uma *Arraylist* pela facilidade de utilização dela e de obtenção dos dados nela armazenados através de loops de repetição e da possibilidade de aplicação de métodos da classe dos objetos sobre ela.

Cada compra é inicializada somente com o voucher e é no *Main* em que durante a execução do usuário que as passagens serão adicionadas ao *ArrayList* com o método da classe.

6.Cartão

A classe cartão, que é abstrata, tem como objetivo definir atributos e métodos em comum aos cartões de crédito e débito. Isso é importante, já que permite a utilização dos conceitos de Herança.

Dentre os atributos estão: Número do cartão, Código de segurança, Validade e uma variável do tipo LocalDateTime que é utilizada para verificar se o cartão está vencido comparando a validade informada pelo usuário com a data atual em tempo real.

No construtor, são informados os atributos do cartão. O dia em que o cartão vence é sempre o último dia do mês informado (pode ser 29,30 e 31).

Sobre os métodos, temos os abstratos e os não abstratos. Os não abstratos consistem basicamente dos getters e setters, que servem para modificar ou visualizar os atributos do cartão. Já os abstratos são os métodos que devem ser sobrescritos nas subclasses do cartão, já que sua execução difere no cartão de crédito e débito, o que permite durante a execução do Main a utilização dos conceitos de polimorfismo.

6.1.Cartão de Crédito

A classe cartão de crédito é uma classe não abstrata que herda a classe cartão e que pode ser instanciada. Aqui temos atributos específicos à cartões de crédito, construtor em que esses atributos específicos são definidos, e seus métodos.

Os atributos são: Fatura (será incrementada de acordo com as compras feitas) e limite da fatura(não será possível realizar mais compras caso o limite for atingido).

No construtor, os atributos (da superclasse cartão) são informados, a fatura é inicializada como R$0,00 e o limite foi definido por nós (achamos R$5000,00 uma quantia razoável para comprar passagens).

Sobre os métodos, temos os que sobrescrevem os abstratos do main e os que não sobrescrevem nada. Os que sobrescrevem são: verificar, que verifica se determinada compra será possível (será possível se o cartão não estiver vencido e se a compra não ultrapassará o limite do cartão) e pagar, que realizará o pagamento caso o preço da compra seja válido(maior que R$0,00) e o método verificar retorne verdadeiro. Os que não sobrescrevem são, basicamente, os getters e os setters.

6.2.Cartão de Débito

A classe cartão de débito é uma classe não abstrata que herda a classe cartão e que pode ser instanciada. Aqui temos o atributo específico de cartões de débito, construtor em que esse atributos específicos é definido, e seus métodos.

O atributo é o saldo do cartão, que será debitado a partir do valor de cada compra.

No construtor os atributos(da superclasse cartão) são informados e o saldo do cartão é inicializado como R$5000,00 , o que novamente, achamos que é uma quantia razoável para que a compra seja feita.

Sobre os métodos, temos os que sobrescrevem os abstratos do main e os que não sobrescrevem nada. Os que sobrescrevem são: verificar, que verifica se determinada compra será possível(será possível se o cartão não estiver vencido e se a compra não ultrapassará o saldo disponível) e pagar, que realizará o pagamento caso o preço da compra seja válido(maior que R$0,00) e o método verificar retorne verdadeiro. Os que não sobrescrevem são, basicamente, os getters e os setters.

7.Pessoa

A utilização da interface *Pessoa* possibilita diminuir o acoplamento facilitando a manutenção e gerenciamento das classes *Adulto* e *Criança* já que uma mudança em uma delas não afetaria a outra, mas ao mesmo tempo ambas deverão implementar o método *getIdentificacao()* que tem como finalidade retornar uma *String* de identificação que no caso de Adulto corresponde ao CPF (numeroDeIdentificacaoExclusivo) e no caso de Criança é referente a matrícula da certidão de nascimento (matriculaDeCertidaoDeNascimento). Sendo a distinção entre adultos e crianças necessária para a aplicação do desconto na passagem de criança.

7.1.Adulto

A classe *Adulto* é utilizada para armazenar informações de nome e CPF das pessoas adultas e por meio do método *getIdentificacao()* retornar o CPF do adulto em questão.

7.2.Criança

A classe *Criança* é utilizada para armazenar informações de nome, idade e matrícula da certidão de nascimento das pessoas crianças e por meio do método *getIdentificacao()* retornar a matrícula da certidão de nascimento da criança em questão.

8.Diagrama UML