



UNIVERSIDADE CATÓLICA DE ANGOLA
FACULDADE DE ENGENHARIA

Curso Engenharia Informática
Disciplina Fundamentos de Programação II
Ano Lectivo 2021/22

Relatório

Projecto de Supermercado Automatizado

Autor:

ID 1000026185
Nome Gilberto Alexandre Adão de Jesus
Turma AINF

Data 10/06/2022
Docente Engº Pedro Mbote

Resumo

Para consolidar os conhecimentos adquiridos de Programação Orientada a Objetos em Linguagem Java, decidiu-se criar um programa para gerenciar um supermercado automatizado de modos a que se faça o controlo eficiente do supermercado em Geral, as entradas, as saídas e as outras operações internas.

1. Introdução

Uma aplicação é um software usado nas mais diversas áreas da vida quotidiana, como forma de automatizar certas resoluções de certos problemas, sendo usado para a resolução de problemas técnicos, sociais, psicológicos entre outras, esses problemas podem ser o tédio e a depressão, inclusive, resolvendo com entretenimento e socialização, jogos e mídias sociais, problemas de aprendizagem desenvolvendo aplicações de acompanhamento psicológico e aprendizagem, mas para esse caso não é nada de diferente do que já disse anteriormente, resolução de problemas, mas nesse caso foi pedido que se desenvolvesse uma aplicação para um Supermercado, de modos a monitorar de forma eficiente e rigorosa, as transações realizadas no interior do supermercado.

2. Descrição do Problema

Foi solicitado que a aplicação seja projetada com certos parâmetros, precisa ser feita na Linguagem Java, em consequência, criar classes específicas mencionadas abaixo:

- A Classe Produto, com o objetivo de conter informação sobre certa mercadoria existente no Estoque do Supermercado
- A Classe Item, com o objetivo de conter informações referente a um produto que se está prestes a comprar
- A Classe ItemDesconto, que tem o objetivo de calcular um tipo de desconto baseados nos atributos da Classe Item
- A Classe Cliente que tem o objetivo de definir o comportamento do Cliente no Supermercado ou durante as compras e mostrar informações relativas ao Cliente
- A Classe Automovel que tem o objetivo de mostrar as informações de um automóvel e certo tipo de comportamento referente á transporte
- A Interface TransporteCarga que se dedica á implementar comportamentos de carga e descarga de mercadorias
- A Classe MenuPrincipal que se dedica a gerenciar e manipular as classes mencionadas anteriormente para por fim, concluir um Sistema Automatizado
- A Classe Executavel que se dedica a executar a classe MenuPrincipal de modos á transforma-lo em um programa

3. Equipamentos e Ferramentas

Computador usado para o desenvolvimento da aplicação: Computador Portátil **INOVIA** – Especificações:

- **Processador:** Intel Core i3-3120M CPU 2.50GHz, 2 núcleos, 4 processadores lógicos
- **Arquitetura:** 64 bits
- **RAM:** 6 GB
- **GPU:** Intel HD Graphics 4000
- **Sistema Operativo:** Microsoft Windows 10 Pro

Linguagem usada para a implementação da Aplicação: **JAVA** - JDK e JRE - Especificações:

- **Versão:** 1.8.0

Ambiente de Desenvolvimento usado: **Notepad++** – Especificações:

- **Versão:** 8

4. Procedimento

Inicialmente no documento do projecto já foi pré-definido para que seja implementado um “Set” de classes, para que o Sistema Automatizado tenha um comportamento específico que não fuja dos critérios definido pelo Supermercado e mais duas classes foram adicionadas para colocar um interface para o sistema e executá-lo como tal, isto para suprir as necessidades do cliente e realizar as operações corretamente.

Foi pré-definido que o “Set” de classes tivesse os itens abaixo:

- Produto
- Item
- Item Desconto
- Carrinho de Compras
- Cliente
- Automóvel
- Interface Transporte de Carga
- Interface Interacao
- Menu Principal
- Executável

É importante referir que o Projecto foi implementado com conceitos de Programação Orientada a Objectos. para que fosse possível realizar as interações entre todas as classes de forma eficiente e sem qualquer equivocação.

4.1. Classe Produto

A classe contém os seguintes atributos e métodos:

- **int** id: Um atributo unívoco usado para efeitos de consulta e busca e identifica o produto como sendo único no Estoque.
- **String** descrição: Contém o nome ou uma informação que identifica o produto.
- **int** quantStoque: Contém a quantidade de produtos existentes no estoque.
- **float** precoPorUnidade: Contém o preço unitário do produto
- **Métodos acessores**(get) dos atributos.
- **Métodos modificadores**(set) dos atributos.
- **Métodos construtores:** implementou-se um construtor padrão e um outro construtor que recebe como parâmetro os 4 atributos
- **Método resumo:** consiste num método que retorna uma string mostrando de forma detalhada o conteúdo dos atributos da classe Produto.

4.2. Classe Item

Na classe contém os seguintes atributos e métodos:

- **Produto** produto: Contém um objeto da classe Produto
- **int** quantidade: Contém a quantidade de produtos existentes no item.
- **Métodos construtores:** implementou-se um construtor que recebe como parâmetro os 2 atributos
- **Métodos assessores**(get) dos atributos.
- **Métodos modificadores**(set) dos atributos.
- **Método calcularPreco:** Calcula o preco total do item, baseado na quantidade existente no carrinho e no preco unitário do produto

4.3. Classe ItemDesconto

Na classe ItemDesconto houve uma mudança do padrão, a classe herdou a Classe Item e, portanto, os seus atributos e métodos, a classe contém os seguintes atributos e métodos:

- **int** n: consiste em um atributo que define um tipo de desconto, mais especificamente o desconto **LEVA n** produtos e **PAGUE apenas n-1** produtos
- **Métodos construtores:** implementou-se um construtor que recebe como parâmetro 3 atributos, dois da SuperClasse e um da SubClasse, onde para tal foi usando o construtor da SuperClasse que recebe 2 parâmetros.
- **Método calcularPreco:** subscreeve o método calcularPreco da sua SuperClasse, para poder gerar o desconto equivalente, para tal foi usado uma fórmula baseado em análises de que, **se n=4, n-1=3, e tivermos uma quantidade de 8 itens implica que apenas se pagará 6 destes itens, isto**

porque á cada 4 itens, apenas se pagará 3 destes itens, o 8 tem dois quatros então $3+3=6$, mas e se fosse 9 itens, quantos seriam pagos? Seriam pagos apenas 7 isto porque teriam dois quatros e mais um, um não é divisível por 4, portanto o desconto não vale para tal. Pela premissa anterior obteve-se a seguinte fórmula:

$$QtdAPagar = \left(\frac{QtdItens}{n} \right) \times (n - 1) + Resto \left(\frac{QtdItens}{n} \right);$$

Neste caso basta multiplicar o QtdAPagar pelo preco unitário do produto.

4.4. Classe CarrinhoCompras

Na classe CarrinhoCompras contém os seguintes atributos e métodos:

- **int** quantidade: contém a quantidade de itens contidos no carrinho de compras, tendo sido inicializado com zero.
- **Array do tipo Item** itens(máximo 200 itens): são possíveis armazenar até 200 itens no carrinho de compras, neste elemento estão contidos todos os itens que o cliente adicionou ao seu carrinho .
- **Método acessor(get):** apenas para o atributo quantidade.
- **Métodos procurarOcorrencia:** recebe um produto como parâmetro e retorna uma posição (um inteiro maior ou igual a zero) no array de itens caso este produto existir no array de itens, no caso contrário, retorna uma posição negativa
- **Método apagarItemArray:** remove um item do array de itens movendo todos os itens posteriores a posição do item que se deseja remover, uma posição anterior. E por fim, diminuindo a quantidade de itens no carrinho de compras.
- **Método adicionar:** adiciona um produto e quantidade recebidos como parâmetro ao array de itens, deste modo, para que não haja mais de um produto ocupando mais de uma posição no array de itens, decidiu-se usar o método **procurarOcorrencia** para pesquisar se existe ou não um mesmo produto no array de itens, em que caso existir, adicione apenas a quantidade passada como parâmetro que é menor ou igual que a quantidade no estoque ao item na posição encontrada adicionando a quantidade ao item do carrinho e removendo a quantidade que se deseja adicionar ao estoque, caso não existir, adicione um novo criando um novo objeto do tipo **Item** ou do tipo **ItemDesconto** e removendo a quantidade que se deseja adicionar ao estoque, caso tiver desconto, na última posição do array que equivalente ao valor do atributo **quantidade**.
- **Método remover:** remove uma quantidade de um produto recebidos como parâmetro ao array de itens, deste modo, para que se verifique se o produto que se deseja remover existe realmente no array de itens ou não, fez-se o uso do método **procurarOcorrencia**, caso não existir, não se remove nada, mas

caso existir, é verificado se a quantidade que se deseja remover é maior ou igual que a quantidade existente no carrinho de compras, para que caso o seja, remova imediatamente do array usando o método **apagarItemArray** e adicionando ao estoque do produto a quantidade removida, ou no caso contrário, remova a quantidade desejada, adicionando ao estoque do produto a quantidade removida.

- **Método transferir:** é um método usado para remover todos os itens contidos no carrinho de compras e transferir a um objeto que implementa a interface **TransporteCarga**, recebido como parâmetro
- **Método abandonar:** consiste em um método que remove todos os produtos do carrinho e retorna a quantidade de cada produto no carrinho a cada produto correspondente no Estoque, recebe como parâmetro um objeto que implementa a interface **Interacao**.
- **Método obterTotal:** soma o valor retornado pelo método **calcularPreco** de cada item no carrinho e retorna o total.
- **Método calcularPreco:** consiste em um método que mostra na tela o que tem e quanto tem no carrinho de compras e o total a se pagar.

4.5. Classe Cliente

A classe Cliente implementa uma interface do tipo **TransporteCarga** e contém os seguintes atributos e métodos:

- **String nome:** contém o nome do cliente
- **long cpf:** contém o cadastro de pessoa física do cliente
- **String enderecoCasa:** contém a casa, rua, bairro do cliente
- **Automovel carro:** contém caso existir, o automóvel que o cliente pretende usar para carregar as mercadorias compradas.
- **Métodos construtores:** recebe um método construtor com os 4 atributos anteriores como parâmetro.
- **CarrinhoCompra carrinho:** contém o carrinho que o cliente pegou
- **Int quantItem:** a quantidade de itens contidos no armazenamento
- **Array do tipo Item armazenamento(200 itens no máximo):** contém todos os itens que o cliente comprou e pretende levar para casa.
- **Métodos acessores(get)** de todos atributos.
- **Métodos modificadores(set)** de todos atributos.
- **Método abandonarCarrinho:** recebe como parâmetro um objeto que implementou uma interface do tipo **Interacao** e chama o método **abandonar** da classe **CarrinhoCompras**, abandona apenas caso o carrinho for diferente de **null**

- **Método pegarCarrinho:** consiste em atribuir um novo carrinho de compras ao atributo **carrinho** caso não tiver nenhum carrinho em mãos, no caso contrário, abandona o carrinho actual e atribui um carrinho.
- **Método comprar:** o método calcula o preço a pagar a partir do método **obterTotal** da classe **CarrinhoCompras**, mostrando quanto se entregou ao caixa e transfere todos os produtos ao armazenamento do cliente para poder levar em casa.
- **Método resumo:** contém informações detalhadas do cliente baseado no conteúdo de cada atributo
- **Métodos implementados da interface TransporteCarga:** o **Carregar** adiciona os produtos comprados ao armazenamento do cliente caso ele não tenha carro, e do carro, no caso contrário. O **Descarregar** retira todos os produtos comprados contidos no armazenamento e imprime na tela todos eles.

4.6. Classe Automovel

A classe Automovel implementa uma interface do tipo TransporteCarga e contém os seguintes atributos e métodos:

- **String nome:** contém o nome do cliente
- **String marca:**
- **String matricula:**
- **Métodos construtores:** recebe um método construtor com os 3 atributos anteriores como parâmetro.
- **Int quantItem:** a quantidade de itens contidos no armazenamento
- **Array do tipo Item armazenamento(200 itens no máximo):** contém todos os itens que o cliente comprou e pretende levar para casa.
- **Métodos acessores(get)** de todos atributos.
- **Métodos modificadores(set)** de todos atributos.
- **Método resumo:** contém informações detalhadas do automovel baseado no conteúdo de cada atributo
- **Métodos implementados da interface TransporteCarga:** o **Carregar** adiciona os produtos comprados ao armazenamento do automovel. O **Descarregar** retira todos os produtos comprados contidos no armazenamento e imprime na tela todos eles.

4.7. Interface TransporteCarga

A Interface TransporteCarga implementa os seguintes métodos:

- **Método Carregar:** adiciona os produtos comprados ao armazenamento do automóvel do cliente caso ele tiver, e caso não adiciona ao armazenamento do cliente e recebe como parâmetro um item.

- **Método Descarregar:** retira todos os produtos comprados contidos no armazenamento e imprime na tela todos eles.

4.8. Interface Interacao

A Interface Interacao implementa os seguintes métodos:

- **Método deixar:** adiciona ao estoque a quantidade de todos os produtos abandonados no carrinho e recebe como parâmetro um item.

4.9. Classe MenuPrincipal

A classe Menu implementa uma interface do tipo Interacao e contém os seguintes atributos e métodos:

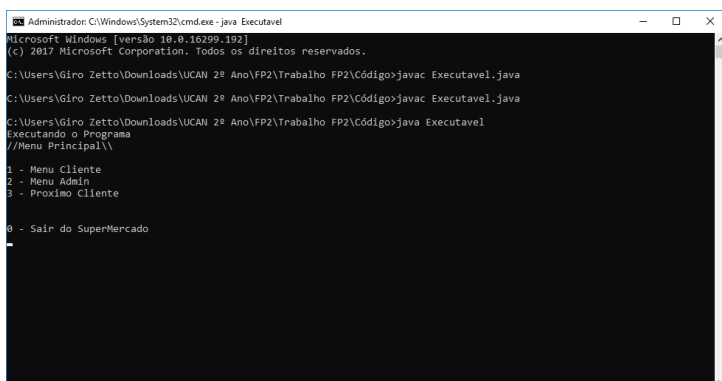
- **Array do tipo Produto** produtos(200 itens no máximo): contém todos os produtos disponíveis no Estoque
- **Array do tipo int** descontos: contém todos os descontos de produtos disponíveis no Estoque.
- **Int** quantidade: quantidade de produtos no Estoque.
- **Cliente** cliente: contém o cliente que está sendo atendido no instante.
- **Método menuTela:** o método genérico serve para mostrar um dado menu, na posição colocada como parâmetro, sendo, 0 – para Menu Principal, 1 – para Menu Cliente, 2 – para o Menu Admin e 3 – para o Menu Compras
- **Método listarProdutos:** método retorna uma String contendo a lista de todos os produtos existentes no Estoque
- **Método executar:** é o método que é chamado para executar o sistema completo começando do menuPrincipal, onde deverá ser escolhidos entre os menus abaixo citados e sair do supermercado e inclusive chamar o próximo cliente a ser atendido ou seja, gerando um aleatório.
- **Método menuCliente:** é o método contendo as opções de operação do cliente, onde deverá ser escolhido entre terminar a compra dos produtos, pegar um carrinho, abandonar um carrinho, o menu compras, listar os produtos no estoque e voltar para o menu principal
- **Método menuAdmin:** é o método contendo as operações do administrador e poderão ser realizadas as operações de listagem, edição, remoção e adição de produtos ao estoque.
- **Método menuCompras:** é o método contendo as operações feitas no carrinho pelo cliente
- **Método editarProduto:** edita as informações de um produto existente no estoque.
- **Método removerProduto:** remove um produto existente no estoque.
- **Método adicionarProduto:** adiciona um novo produto existente no estoque.
- **Método adicionarItem:** escolher o produto e a quantidade que deseja retirar do estoque e adicionar ao carrinho de compras.

Relatório – Projecto de Supermercado Automatizado

- **Método removerItem:** escolher o produto e a quantidade que deseja retirar do carrinho de compras e adicionar ao estoque.
- **Método gerarCliente:** gera um cliente com atributos aleatórios para poder dinamizar as operações feitas no sistema automatizado e adiciona ele ao atributo cliente.
- **Método gerarAutomovel:** gera um automóvel com atributos aleatórios para poder dinamizar as operações feitas no sistema automatizado ou não gera.
- **Métodos implementados a partir da interface Interacao:** o método **deixar** que recebe um item e adiciona sua quantidade ao estoque de modo a recuperar os produtos abandonados.

Obs: A classe Executavel, é a classe que contém o método main e unicamente este método, portanto é a única que deve ser executada.

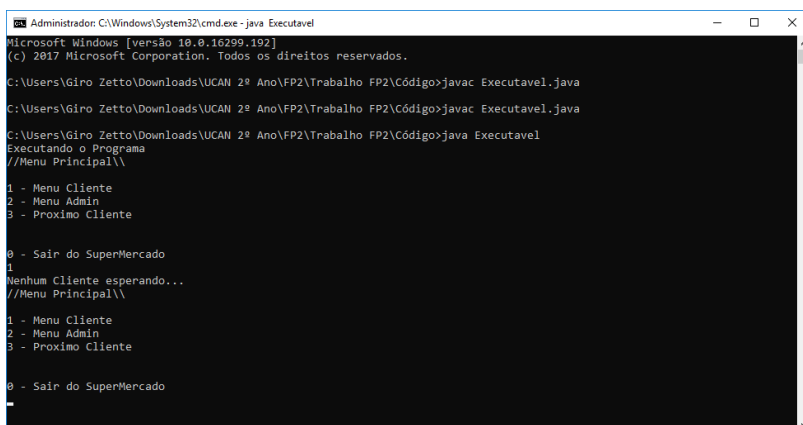
5. Resultados e Análise dos Resultados



```
Administrador: C:\Windows\System32\cmd.exe - java Executavel
Microsoft Windows [versão 10.0.16299.192]
(c) 2017 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Giro Zetto\Downloads\UCAN 2º Ano\FP2\Trabalho FP2\Código>javac Executavel.java
C:\Users\Giro Zetto\Downloads\UCAN 2º Ano\FP2\Trabalho FP2\Código>javac Executavel.java
C:\Users\Giro Zetto\Downloads\UCAN 2º Ano\FP2\Trabalho FP2\Código>java Executavel
Executando o Programa
//Menu Principal\\
1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente
0 - Sair do SuperMercado
```

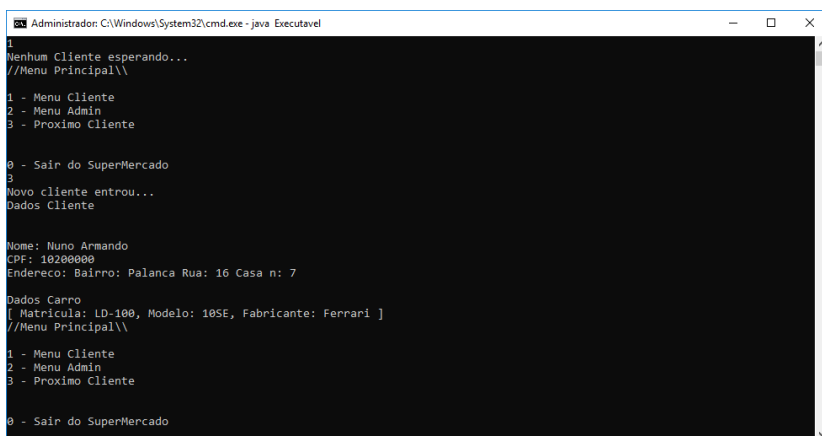
É executado com um menu simples com apenas 4 opções.



```
Administrador: C:\Windows\System32\cmd.exe - java Executavel
Microsoft Windows [versão 10.0.16299.192]
(c) 2017 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Giro Zetto\Downloads\UCAN 2º Ano\FP2\Trabalho FP2\Código>javac Executavel.java
C:\Users\Giro Zetto\Downloads\UCAN 2º Ano\FP2\Trabalho FP2\Código>javac Executavel.java
C:\Users\Giro Zetto\Downloads\UCAN 2º Ano\FP2\Trabalho FP2\Código>java Executavel
Executando o Programa
//Menu Principal\\
1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente
0 - Sair do SuperMercado
1
Nenhum Cliente esperando...
//Menu Principal\\
1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente
0 - Sair do SuperMercado
```

Não se pode entrar no Menu Cliente sem ter um cliente



```
Administrador: C:\Windows\System32\cmd.exe - java Executavel
1
Nenhum Cliente esperando...
//Menu Principal\\
1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente
0 - Sair do SuperMercado
3
Novo cliente entrou...
Dados Cliente

Nome: Nuno Armando
CPF: 10200000
Endereço: Bairro: Palanca Rua: 16 Casa n: 7
Dados Carro
[ Matrícula: LD-100, Modelo: 10SE, Fabricante: Ferrari ]
//Menu Principal\\
1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente
0 - Sair do SuperMercado
```

Ao seleccionar a opção Próximo Cliente, é gerado um novo cliente

Relatório – Projecto de Supermercado Automatizado

```
Administrador: C:\Windows\System32\cmd.exe - java Executavel
3
Novo cliente entrou...
Dados Cliente

Nome: Nuno Armando
CPF: 10200000
Endereco: Bairro: Palanca Rua: 16 Casa n: 7

Dados Carro
[ Matricula: LD-100, Modelo: 10SE, Fabricante: Ferrari ]
//Menu Principal\\

1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente

0 - Sair do SuperMercado
3
Atenda o Cliente em Espera...
//Menu Principal\\

1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente

0 - Sair do SuperMercado
```

E se voltar á tentar gerar novamente um outro cliente sem ter atendido o actual, não lhe é permitida a geração.

```
Administrador: C:\Windows\System32\cmd.exe

Nome: Nuno Armando
CPF: 10200000
Endereco: Bairro: Palanca Rua: 16 Casa n: 7

Dados Carro
[ Matricula: LD-100, Modelo: 10SE, Fabricante: Ferrari ]
//Menu Principal\\

1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente

0 - Sair do SuperMercado
3
Atenda o Cliente em Espera...
//Menu Principal\\

1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente

0 - Sair do SuperMercado
0
Desligando o sistema...
C:\Users\Giro Zetto\Downloads\UCAN 2º Ano\FP2\Trabalho FP2\Código>
```

Como já é sabido, o sair do sistema funciona correctamente.

```
Administrador: C:\Windows\System32\cmd.exe - java Executavel
2 - Menu Admin
3 - Proximo Cliente

0 - Sair do SuperMercado
0
Desligando o sistema...
C:\Users\Giro Zetto\Downloads\UCAN 2º Ano\FP2\Trabalho FP2\Código>java Executavel
Executando o Programa
//Menu Principal\\

1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente

0 - Sair do SuperMercado
2
//Menu Admin\\

1 - Adicionar Produto ao Estoque
2 - Remover Produto do Estoque
3 - Editar Produto no Estoque
4 - Listar Produtos do Estoque

0 - Voltar ao Menu Principal
```

Ao entrar no Menu Admin, pode se realizar 5 operações de gestão do Stoque.

Relatório – Projecto de Supermercado Automatizado

```
Administrador: C:\Windows\System32\cmd.exe - java Executavel

1 - Adicionar Produto ao Estoque
2 - Remover Produto do Estoque
3 - Editar Produto no Estoque
4 - Listar Produtos do Estoque

0 - Voltar ao Menu Principal

1
Insira a Descricao do Produto
Massa 340g
Insira a quantidade do Produto
50
Insira o Preço por Unidade
250
Insira o desconto - (numero menor que 2 - para produto sem desconto)
4
Adicionando Produto ao Estoque...
//Menu Admin\\

1 - Adicionar Produto ao Estoque
2 - Remover Produto do Estoque
3 - Editar Produto no Estoque
4 - Listar Produtos do Estoque

0 - Voltar ao Menu Principal
```

Ao adicionar é necessário preencher todos os campos e incluindo mencionar o desconto que foi dado ao produto ou não, inserindo um valor menor do que 2. Para este caso foi inserido 4 para testar se realmente o desconto funciona ou não.

```
Administrador: C:\Windows\System32\cmd.exe - java Executavel

1 - Adicionar Produto ao Estoque
2 - Remover Produto do Estoque
3 - Editar Produto no Estoque
4 - Listar Produtos do Estoque

0 - Voltar ao Menu Principal

2
Lista Produtos
1 - [ ID: 1, Descricao: Massa 340g, Quantidade em Stoque: 50, Preço por Unidade: 250.0 ] Desconto: LEVE 4 e PAGUE 3
2 - [ ID: 2, Descricao: Arroz 4kg, Quantidade em Stoque: 10, Preço por Unidade: 3000.0 ] Desconto: LEVE 3 e PAGUE 2
3 - [ ID: 3, Descricao: Gelado de Mucua, Quantidade em Stoque: 30, Preço por Unidade: 59.0 ]

Insira a Posicao do Produto que deseja eliminar
2
[ ID: 2, Descricao: Arroz 4kg, Quantidade em Stoque: 10, Preço por Unidade: 3000.0 ]
Produto Removido
//Menu Admin\\

1 - Adicionar Produto ao Estoque
2 - Remover Produto do Estoque
3 - Editar Produto no Estoque
4 - Listar Produtos do Estoque

0 - Voltar ao Menu Principal
```

Ao listar pode se verificar que foi adicionado com sucesso outros produtos além do anterior e foi realizada uma operação de remoção.

Relatório – Projecto de Supermercado Automatizado

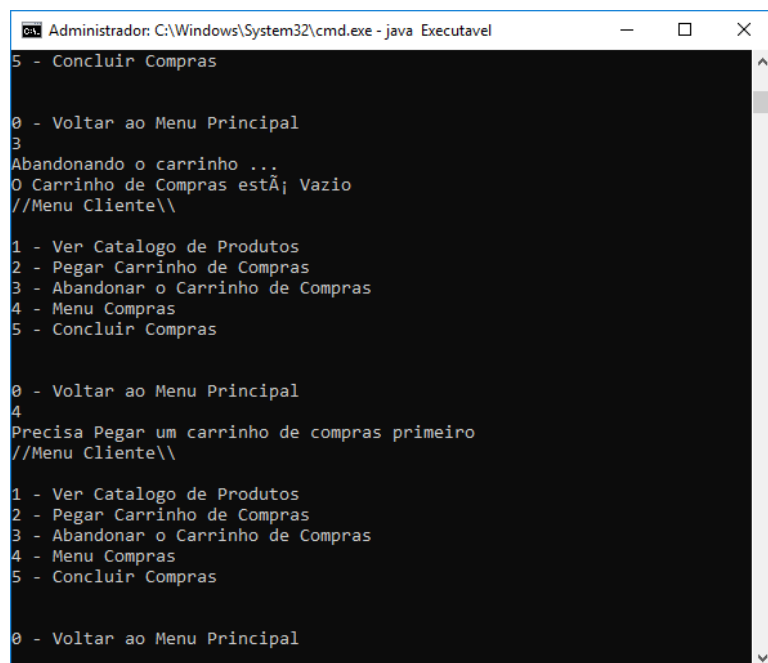
```
4
Lista Produtos
1 - [ ID: 1, Descricao: Massa 340g, Quantidade em Stoque: 50, Preco por
Unidade: 250.0 ] Desconto: LEVE 4 e PAGUE 3
2 - [ ID: 3, Descricao: Gelado de Mucua, Quantidade em Stoque: 30, Prec
o por Unidade: 59.0 ]

//Menu Admin\\

1 - Adicionar Produto ao Estoque
2 - Remover Produto do Estoque
3 - Editar Produto no Estoque
4 - Listar Produtos do Estoque

0 - Voltar ao Menu Principal
```

Ao listar pode se verificar que o produto seleccionado foi removido.



```
5 - Concluir Compras

0 - Voltar ao Menu Principal
3
Abandonando o carrinho ...
0 Carrinho de Compras está Vazio
//Menu Cliente\\

1 - Ver Catalogo de Produtos
2 - Pegar Carrinho de Compras
3 - Abandonar o Carrinho de Compras
4 - Menu Compras
5 - Concluir Compras

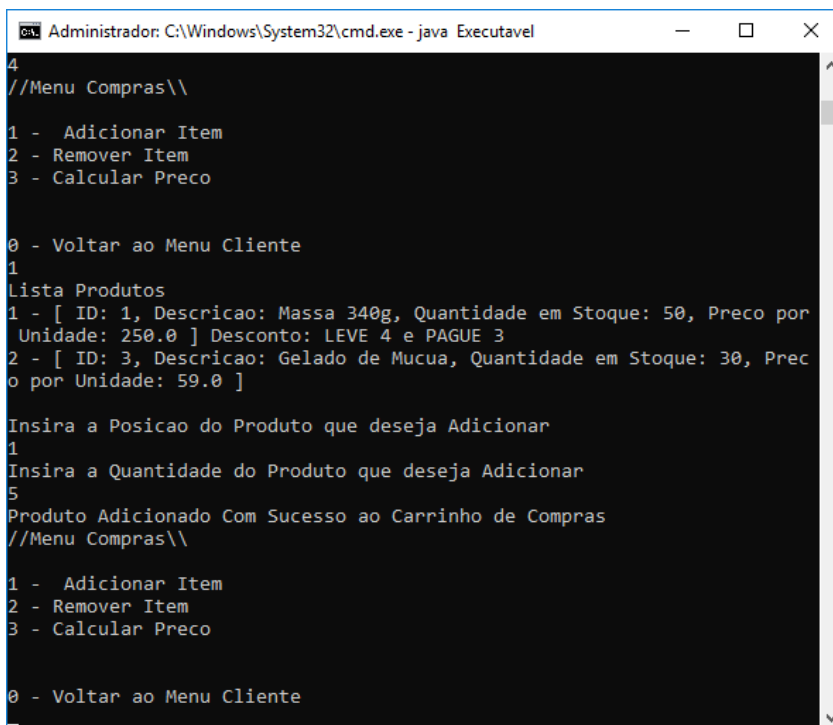
0 - Voltar ao Menu Principal
4
Precisa Pegar um carrinho de compras primeiro
//Menu Cliente\\

1 - Ver Catalogo de Produtos
2 - Pegar Carrinho de Compras
3 - Abandonar o Carrinho de Compras
4 - Menu Compras
5 - Concluir Compras

0 - Voltar ao Menu Principal
```

Não se pode abandonar um carrinho que o cliente não tenha, nem se pode ir ao menu de compras sem ter pego um carrinho de compras.

Relatório – Projecto de Supermercado Automatizado



```
Administrador: C:\Windows\System32\cmd.exe - java Executavel
4
//Menu Compras\\
1 - Adicionar Item
2 - Remover Item
3 - Calcular Preco

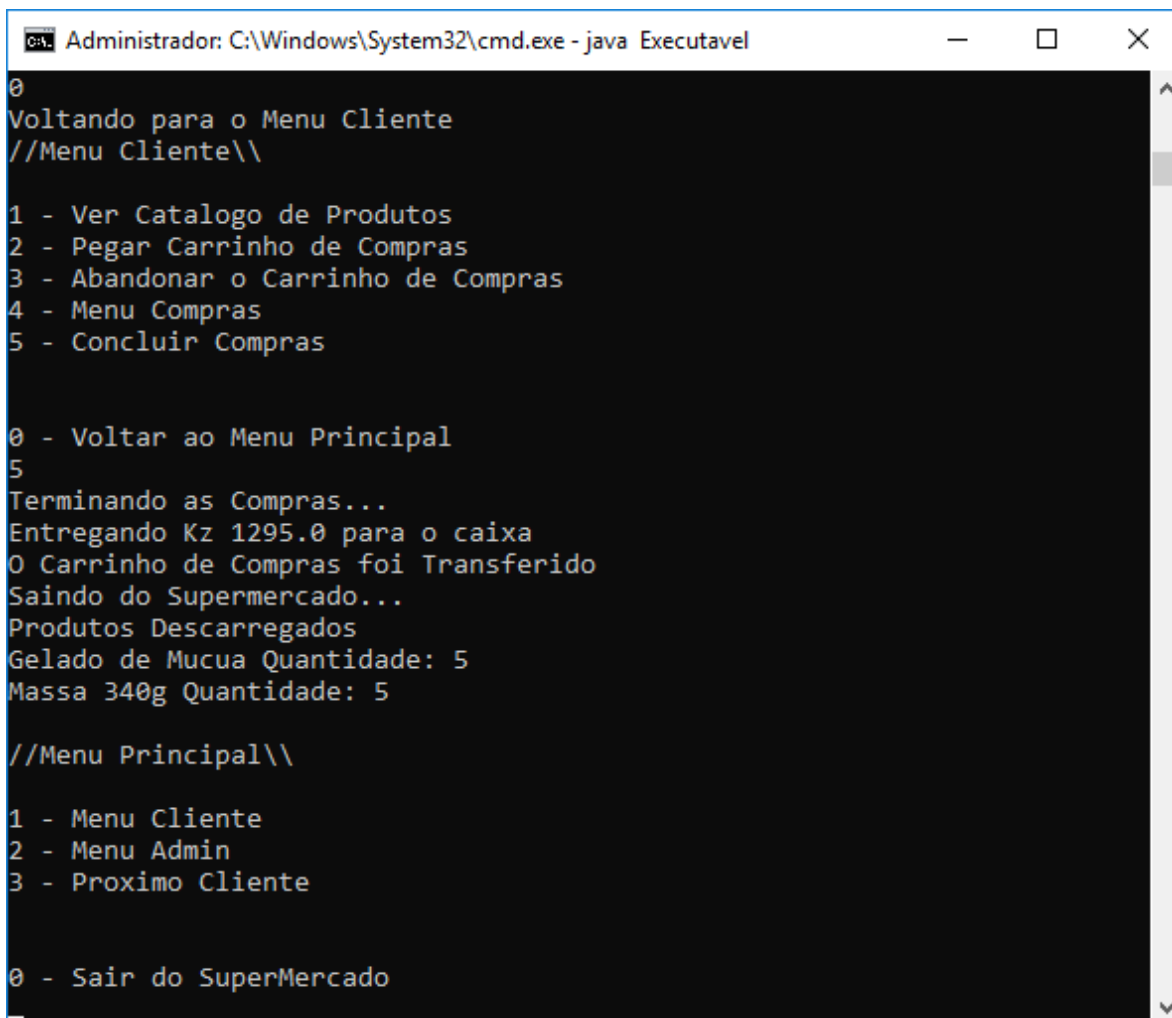
0 - Voltar ao Menu Cliente
1
Lista Produtos
1 - [ ID: 1, Descricao: Massa 340g, Quantidade em Stoque: 50, Preco por
Unidade: 250.0 ] Desconto: LEVE 4 e PAGUE 3
2 - [ ID: 3, Descricao: Gelado de Mucua, Quantidade em Stoque: 30, Prec
o por Unidade: 59.0 ]

Insira a Posicao do Produto que deseja Adicionar
1
Insira a Quantidade do Produto que deseja Adicionar
5
Produto Adicionado Com Sucesso ao Carrinho de Compras
//Menu Compras\\
1 - Adicionar Item
2 - Remover Item
3 - Calcular Preco

0 - Voltar ao Menu Cliente
```

Foi adicionado ao carrinho itens e removidos.

Relatório – Projecto de Supermercado Automatizado



```
Administrador: C:\Windows\System32\cmd.exe - java Executavel
0
Voltando para o Menu Cliente
//Menu Cliente\\

1 - Ver Catalogo de Produtos
2 - Pegar Carrinho de Compras
3 - Abandonar o Carrinho de Compras
4 - Menu Compras
5 - Concluir Compras

0 - Voltar ao Menu Principal
5
Terminando as Compras...
Entregando Kz 1295.0 para o caixa
0 Carrinho de Compras foi Transferido
Saindo do Supermercado...
Produtos Descarregados
Gelado de Mucua Quantidade: 5
Massa 340g Quantidade: 5

//Menu Principal\\

1 - Menu Cliente
2 - Menu Admin
3 - Proximo Cliente

0 - Sair do SuperMercado
```

No final de tudo quando se quer concluir a compra pode ser verificado que conclui as compras.

6. Considerações Finais

Este programa serviu de simulação para a realização de forma eficaz do Sistema Automatizado, se baseando nos parâmetros dados pelo enunciado, executando todas e quaisquer operações referentes ao monitoramento e gerenciamento dos carrinhos de compra.

7. Referências Bibliográficas

- Fonseca P. – Guia para a redação de Relatórios – 11ª Edição – 2012
- Deitel P., Deitel H. – JAVA - Como Programar – 10ª Edição – Pearson Education do Brasil, 2017

8. Anexos