package supernosso;

public class Produto {

//Atributos da classe Produto:

private String nome;

private double preco;

private int estoque;

//Construtor da classe produto:

public Produto(String nome, double preco, int estoque) {

this.nome = nome;

this.preco = preco;

this.estoque = estoque;

}

//Construtor padrÃ£o da classe produto:

public Produto() {

}

// Getter para o atributo nome:

public String getNome() {

return nome;

}

// Setter para o atributo nome:

public void setNome(String nome) {

this.nome = nome;

}

// Getter para o atributo preco:

public double getPreco() {

return preco;

}

// Setter para o atributo preco:

public void setPreco(double preco) {

this.preco = preco;

}

// Getter para o atributo estoque:

public int getEstoque() {

return estoque;

}

// Setter para o atributo estoque:

public void setEstoque(int estoque) {

this.estoque = estoque;

}

//Adiciona uma quantidade ao estoque:

public void incluiEstoque(int quantidade) {

estoque += quantidade + estoque;

}

//Subtrai uma quantidade do estoque, se houver estoque suficiente:

public void tirarEstoque(int quantidade) {

if (estoque <= quantidade) {

estoque -= quantidade;

}

}

//// Sobrescrita do mÃ©todo toString para exibir informaÃ§Ãµes do produto:

@Override

public String toString() {

return "Nome: " + nome + ", Valor: " + preco + ", Estoque: " + estoque;

}

//Retorna a quantidade existente:

int getQuantidade() {

return 0;

}

}

package supernosso;

import java.util.ArrayList;

import java.util.List;

public class Carrinho {

private final List<Produto> itens;

Produto adicionarItem;

//Este é um construtor público da classe Carrinho, o que significa que ele pode ser chamado para criar novos objetos Carrinho.

public Carrinho() {

//new ArrayList<>(): Cria uma nova instância da classe ArrayList, que é uma implementação da interface List em Java. Esta lista será utilizada para armazenar os itens no carrinho de compras.

this.itens = new ArrayList<>();

}

public void adicionarItem(Produto produto, int quantidade) {

// Verifica se a quantidade é válida:

if (quantidade <= 0) {

System.out.println("Quantidade inválida. O item não foi adicionado ao carrinho.");

return;

}

// Adiciona o produto ao carrinho e exibe a informação na tela:

for (int i = 0; i < quantidade; i++) {

this.itens.add(produto);

}

System.out.println(quantidade + " unidades de " + produto.getNome() + " adicionadas ao carrinho.");

}

public void removerItem(Produto produto) {

// Remove o produto do carrinho, se existir

if (this.itens.contains(produto)) {

this.itens.remove(produto);

System.out.println(produto.getNome() + " removido do carrinho.");

} else {

System.out.println("O produto não está no carrinho.");

}

}

public void limparCarrinho() {

// Remove todos os itens do carrinho

this.itens.clear();

System.out.println("Carrinho limpo.");

}

//Soma os preços dos produtos no carrinho:

public double calcularTotal() {

double total = 0.0;

total = this.itens.stream().map((produto) -> produto.getPreco()).reduce(total, (accumulator, \_item) -> accumulator + \_item);

return total;

}

}

package supernosso;

public class FormadePagamento {

//Atributos da classe forma de pagamento:

private String cartao;

private int opcao;

private String cartao\_de\_credito;

private String cartao\_de\_debito;

//contrutor da forma de pagamento:

FormadePagamento(String cartao, int opcao) {

this.cartao = cartao;

this.opcao = opcao;

}

// Este método está definido para retornar uma string, mas atualmente lança uma exceção indicando que não está implementado.

String getValor() {

throw new UnsupportedOperationException();

}

// Este método está definido para receber um valor double, mas também lança uma exceção indicando que não está implementado.

void getValor(double total) {

throw new UnsupportedOperationException();

}

//Este método está definido para retornar uma string que representa o cartão de crédito:

public String getCartao\_de\_credito() {

String cartao\_de\_credito = null;

return cartao\_de\_credito;

}

//Define o valor do atributo cartao\_de\_credito:

public void setCartao\_de\_credito(String cartao\_de\_credito) {

this.cartao\_de\_credito = cartao\_de\_credito;

}

//Este método está definido para retornar uma string que representa o cartão de debitoto:

public String getCartao\_de\_debito() {

String cartao\_de\_debito = null;

return cartao\_de\_debito;

}

//Define o valor do atributo cartao\_de\_credito:

public void setCartao\_de\_debito(String cartao\_de\_debito) {

this.cartao\_de\_debito = cartao\_de\_debito;

}

public class FormaPagamento {

//Uma string que representa o tipo de pagamento.

String tipo;

//Um valor double que representa o valor associado ao pagamento.

double valor;

//Inicializa o atributo tipo com o valor fornecido.

public FormaPagamento(String tipo) {

this.tipo = tipo;

}

//Retorna o valor do atributo valor.

public double getValor() {

return valor;

}

//Define o valor do atributo valor.

public void setValor(double valor) {

this.valor = valor;

}

// Retorna uma representação em string do objeto FormaPagamento, mostrando o valor.

@Override

public String toString() {

return "valor= " + valor;

}

//Retorna o valor do atributo tipo.

public String getTipo() {

return tipo;

}

// Chama o tipo de pagamento e o valor a ser pago.

public String setTipo() {

return "tipo de pagamento" + tipo + "valor: " + valor;

}

}

}

package supernosso;

import java.util.Scanner;

public class SuperNosso {

private static Produto produtoSelecionado;

public static void main(String[] args) {

Scanner entrada = new Scanner(System.in);

Carrinho carrinho = new Carrinho();

// Criar um produto:

Produto p1 = new Produto("Arroz", 29.99, 5);

Produto p2 = new Produto("Feijão", 8.99, 2);

Produto p3 = new Produto("Carne", 30.00, 10);

Produto p4 = new Produto("Fubar", 10.99, 8);

Produto p5 = new Produto("Macarrão", 5.99, 4);

Produto p6 = new Produto("Oléo", 7.99, 2);

Produto[] produtos = {p1, p2, p3, p4, p5, p6};

boolean continuarComprando = true;

//metodo de repetição para selecionar mais produtos:

while (continuarComprando) {

System.out.println("Selecione um Produto (ou digite 0 para finalizar a compra): ");

int opcao = entrada.nextInt();

if (opcao > 0 && opcao <= produtos.length) {

// Ajuste para indexar corretamente o array:

Produto selecionado = produtos[opcao - 1];

if (selecionado != null) {

System.out.println("Digite a quantidade desejada: ");

int quantidadeDesejada = entrada.nextInt();

if (quantidadeDesejada > selecionado.getEstoque()) {

System.out.println("Desculpe, não há estoque suficiente desse produto.");

} else {

carrinho.adicionarItem(selecionado, quantidadeDesejada);

selecionado.tirarEstoque(quantidadeDesejada); // Remover a quantidade do estoque

System.out.println("");

}

}

} else if (opcao == 0) {

continuarComprando = false;

} else {

System.out.println("Opção inválida!!!");

}

}

//Metodo para calcular o total adicionado no carrinho:

double total = carrinho.calcularTotal();

System.out.println("Valor total a pagar: n% R$ " + total);

System.out.println("");

System.out.println("");

//Metodo para informar as formas de pagamento:

FormadePagamento pagamento1 = new FormadePagamento("cartao de credito", 1);

FormadePagamento pagamento2 = new FormadePagamento("cartao de debito", 2);

//Exibe ao usuario qual opção selecionar:

System.out.println("Selecione uma forma de pagamento: ");

System.out.println("1. CARTAO\_DE\_CREDITO");

System.out.println("2. CARTAO\_DE\_DEBITO");

//Solicita ao usuario a opção desejada para o pagamento:

int opcaoPagamento = entrada.nextInt();

//Metodo usado para guardar e exibir a solicitação dada pelo usuario:

switch (opcaoPagamento) {

case 1:

pagamento1.getCartao\_de\_credito();

break;

case 2:

pagamento2.getCartao\_de\_debito();

break;

default:

System.out.println("Opção invalida, por favor escolha uma opção valida");

}

//Ao selecionar um forma de pgamento exibe qual foi escolhida e pede para o mesmo realizar o pagamento:

if (opcaoPagamento == 1) {

System.out.println("Forma de pagamento selecionado : " + opcaoPagamento + " " + "Cartao de credito. " + " Aproxime ou insira o cartão: ");

} else {

System.out.println("Forma de pagamento selecionado : " + opcaoPagamento + " " + "Cartao de debito. " + " Aproxime ou insira o cartão: ");

}

{

}

}

}