

TEMA PROJETO: Cadeia de Restaurantes(Bob's ou Subway)

Discente: João Batista Araújo de Lima Filho.

Questão 1

Classe Pedido

```
package atividade9;

import java.util.Arrays;

// 1. Utilizando uma classe básica do seu projeto:
//a) Faça com que uma classe do seu projeto implemente a interface
//java.lang.Comparable.

“Aqui foi implementado a interface Comparable a classe Pedido”

public class Pedido implements Comparable<Pedido>{

    String cliente;
    String lanche;
    Double valorPedido;
    int numItensPedidos;
    boolean fazerPedido;

    public Pedido(String cliente, String nomeLanche) {
        this.lanche = nomeLanche;
        this.fazerPedido();
        nomeCliente(cliente);
        addLanche(nomeLanche);
    }

    void numerolItensPedidos() {
        numItensPedidos++;
    }

    void fazerPedido() {
        valorPedido = 0d;
        fazerPedido = true;
    }

    double addRefri() {
        numerolItensPedidos();
        return valorPedido += 3.50;
    }

    String nomeCliente(String nome) {

        return cliente = nome;
    }
}
```

```
}

void addLanche(String lanche) {

    switch (lanche) {

        case "Hamburguer":
            valorPedido += 9.99;
            numeroltensPedidos();
            break;
        case "Sundae":
            valorPedido += 7.99;
            numeroltensPedidos();
            break;
        case "Batata":
            valorPedido += 3.99;
            numeroltensPedidos();
            break;
        case "Casquinha":
            valorPedido += 4.49;
            numeroltensPedidos();
            break;
        default:
            System.out.println("Lanche em falta!");
    }
}
```

```
public String getCliente() {
    return cliente;
}

public void setCliente(String cliente) {
    this.cliente = cliente;
}

public Double getValorPedido() {
    return valorPedido;
}

public void setValorPedido(Double valorPedido) {
    this.valorPedido = valorPedido;
}

public int getNumItensPedidos() {
    return numItensPedidos;
}

public void setNumItensPedidos(int numItensPedidos) {
    this.numItensPedidos = numItensPedidos;
}

public boolean isFazerPedido() {
    return fazerPedido;
}
```

```

public void setFazerPedido(boolean fazerPedido) {
    this.fazerPedido = fazerPedido;
}

@Override
public String toString() {
    return
        "Nome cliente: " + cliente +
        ", Nome do Lanche: " + lanche +
        ", Valor do pedido: R$" + valorPedido +
        ", Fazer pedido: " + fazerPedido ;
}

```

“Sobrescrita do método compareTo da interface Comparable e passagem dos parâmetros para fazer a ordenação”

```

@Override
public int compareTo(Pedido pedido) {
    if(this.valorPedido > pedido.getValorPedido()){
        return 1;
    } else if(this.valorPedido < pedido.getValorPedido()){
        return -1;
    } else {
        return 0;
    }
}

```

“Método main() da Classe Pedido”

```

public static void main(String[] args) {

    System.out.println("----- Questao 1 ----- \n");
    // 1. b) Crie um vetor de objetos desta classe e ordene utilizando Arrays.sort().

    "Aqui foi criado um vetor de objetos pedidos"

    Pedido[] pedidos = {
        new Pedido("João","Sundae"),new Pedido("Tiago","Hamburguer"),
        new Pedido("Maria","Batata")
    };
    // c) Mostre na tela o print do vetor antes e depois da ordenação.

    System.out.println("Vetor Antes da Ordenação por Preço:\n");

    "Vetor antes da ordenação"

    for (Pedido i: pedidos){
        System.out.println(i.toString());
    }
    System.out.println("\nVetor Depois da Ordenção por Preço do Menor pro Maior:\n");

    "Aqui foi feita a ordenação do vetor usando Arrays.sort()"

    Arrays.sort(pedidos); // Ordenação Arrays.sort()

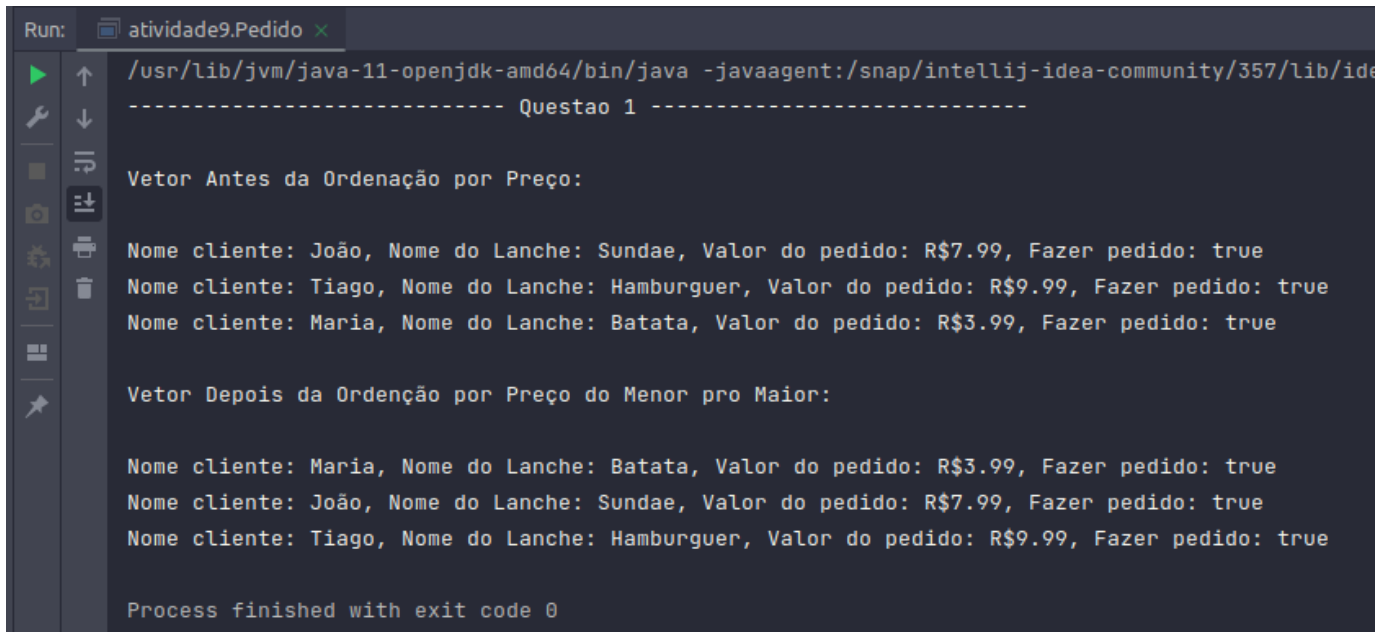
    for (Pedido i: pedidos) {
        System.out.println(i.toString());
    }
}

```

“Vetor depois da ordenação”

```
}  
  
}
```

IMAGENS DAS SAÍDAS NO CONSOLE



```
Run: atividade9.Pedido x  
/usr/lib/jvm/java-11-openjdk-amd64/bin/java -javaagent:/snap/intellij-idea-community/357/lib/ide  
----- Questao 1 -----  
  
Vetor Antes da Ordenação por Preço:  
  
Nome cliente: João, Nome do Lanche: Sundae, Valor do pedido: R$7.99, Fazer pedido: true  
Nome cliente: Tiago, Nome do Lanche: Hamburguer, Valor do pedido: R$9.99, Fazer pedido: true  
Nome cliente: Maria, Nome do Lanche: Batata, Valor do pedido: R$3.99, Fazer pedido: true  
  
Vetor Depois da Ordenção por Preço do Menor pro Maior:  
  
Nome cliente: Maria, Nome do Lanche: Batata, Valor do pedido: R$3.99, Fazer pedido: true  
Nome cliente: João, Nome do Lanche: Sundae, Valor do pedido: R$7.99, Fazer pedido: true  
Nome cliente: Tiago, Nome do Lanche: Hamburguer, Valor do pedido: R$9.99, Fazer pedido: true  
  
Process finished with exit code 0
```

Questão 2

Interface IPagamento

```
package atividade9;  
  
// 2. a) Construa uma interface para seu projeto.  
//    b) A interface deve ter dois métodos.  
  
    “Interface de pagamento com dois métodos.”  
  
public interface IPagamento {  
  
    void formaPagamento(String forma);  
    void valorTotal();  
}
```

Interface IDescricao

```

package atividade9;

// 2. c) Construa uma segunda interface, com dois métodos.

    "Aqui temos a segunda interface com dois métodos"

public interface IDescricao {

    void numeroDoPedido(int numero);
    void descricao();

}

```

Classes que Implementam Ambas as Interfaces

Classe Bebida

```

package atividade9;

// 2. d) Crie classes que implementam ambas as interfaces.

    "Aqui temos a classe Bebida que é a primeira classe que implementa as duas interfaces"

public class Bebida implements IPagamento, IDescricao{

    private String nomeBebida;
    private int numeroPedido;
    private double valorTotal;
    private String formaPagamento;

    public Bebida(String nomeBebida, int numeroPedido, String formaPagamento) {
        this.nomeBebida = nomeBebida;
        numeroDoPedido(numeroPedido);
        formaPagamento(formaPagamento);
        valorTotal();
    }

    public String getNomeBebida() {
        return nomeBebida;
    }

    public void setNomeBebida(String nomeBebida) {
        this.nomeBebida = nomeBebida;
    }

    public int getNumeroPedido() {
        return numeroPedido;
    }

    public void setNumeroPedido(int numeroPedido) {
        this.numeroPedido = numeroPedido;
    }

    public double getValorTotal() {
        return valorTotal;
    }
}

```

```

}

public String getFormaPagamento() {
    return formaPagamento;
}

“Sobrescrita dos dois métodos da interface IDescricao”

@Override
public void numeroDoPedido(int numeroPedido) {
    this.numeroPedido = numeroPedido;
}

@Override
public void descricao() {
    System.out.println("Descrição:\nBebida: "+getNomeBebida()+"", Numero:
"+getNumeroPedido()+"", Valor Total: R$ "+getValorTotal()
+", Forma de Pagamento: "+getFormaPagamento());
}

“Sobrescrita dos dois métodos da interface Ipagamento”

@Override
public void formaPagamento(String formaPagamento) {
    this.formaPagamento = formaPagamento;
}

@Override
public void valorTotal() {
    if(nomeBebida.equals("Refrigerante")){
        this.valorTotal = 3.99;
    } else if(nomeBebida.equals("Agua")){
        this.valorTotal = 1.99;
    } else {
        this.valorTotal = 4.99;
    }
}

“Método main() da classe Bebida”

public static void main(String[] args) {
    Bebida bebida = new Bebida("Refrigerante",23,"Dinheiro");
    bebida.descricao();
}

}

```

IMAGENS DAS SAÍDAS NO CONSOLE

```
Run: Bebida x
/usr/lib/jvm/java-11-openjdk-amd64/bin/java -javaagent:/snap/intellij-idea-community/357
Descrição:
Bebida: Refrigerante, Numero: 23, Valor Total: R$ 3.99, Forma de Pagamento: Dinheiro
Process finished with exit code 0
```

Classe Salada

```
package atividade9;
```

```
// 2. d) Crie classes que implementam ambas as interfaces.
```

“Aqui temos a classe Salada que é a segunda classe que implementa as duas interfaces”

```
public class Salada implements IDescricao, IPagamento {
```

```
    private String nomeSalada;
    private int numeroPedido;
    private double valorTotal;
    private String formaPagamento;
```

```
    public Salada(String nomeSalada, int numeroPedido, String formaPagamento) {
        this.nomeSalada = nomeSalada;
        numeroDoPedido(numeroPedido);
        formaPagamento(formaPagamento);
        valorTotal();
    }
```

```
    public String getNomeSalada() {
        return nomeSalada;
    }
```

```
    public void setNomeSalada(String nomeSalada) {
        this.nomeSalada = nomeSalada;
    }
```

```
    public int getNumeroPedido() {
        return numeroPedido;
    }
```

```
    public void setNumeroPedido(int numeroPedido) {
        this.numeroPedido = numeroPedido;
    }
```

```
    public double getValorTotal() {
        return valorTotal;
    }
```

```
    public String getFormaPagamento() {
        return formaPagamento;
    }
```

“Sobrescrita dos dois métodos da interface IDescricao”

```
@Override
public void numeroDoPedido(int numero) {
    this.numeroPedido = numero;
}

@Override
public void descricao() {
    System.out.println("Descrição:\nSalada: "+getNomeSalada()+"", Numero:
"+getNumeroPedido()+"", Valor Total: R$ "+getValorTotal()
    "+", Forma de Pagamento: "+getFormaPagamento());
}

“Sobrescrita dos dois métodos da interface IPagamento”

@Override
public void formaPagamento(String forma) {
    this.formaPagamento = forma;
}

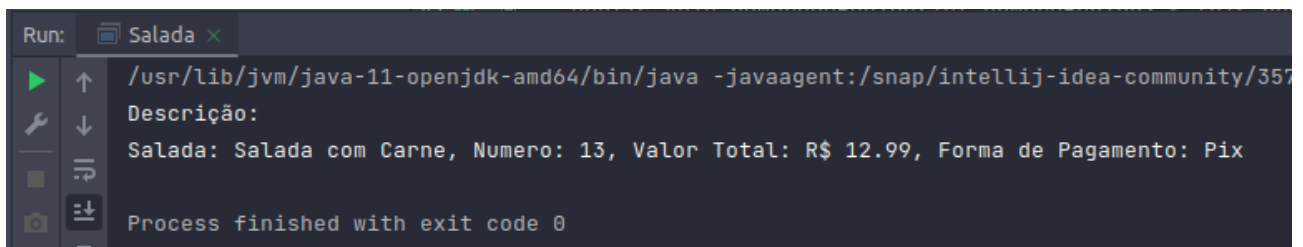
@Override
public void valorTotal() {
    if(nomeSalada.equals("Salada com Frango")){
        this.valorTotal = 8.99;
    } else if(nomeSalada.equals("Salada com Carne")){
        this.valorTotal = 12.99;
    } else {
        this.valorTotal = 6.99;
    }
}

“Método main() da classe Salada”

public static void main(String[] args) {
    Salada salada = new Salada("Salada com Carne",13,"Pix");
    salada.descricao();
}

}
```

IMAGENS DAS SAÍDAS NO CONSOLE



```
Run: Salada x
/usr/lib/jvm/java-11-openjdk-amd64/bin/java -javaagent:/snap/intellij-idea-community/357
Descrição:
Salada: Salada com Carne, Numero: 13, Valor Total: R$ 12.99, Forma de Pagamento: Pix
Process finished with exit code 0
```