

# Universidade da Beira Interior

Programação Orientada a Objectos

*Cursos: 1º ciclo:*

*Eng.ª Informática, Tecnologias e Sistemas de Informação, Informática Web*

Exame, 2016/01/22

---

**Sem consulta e sem telemóvel**

**Duração: 1 hora e 45 minutos**

I

1 – Um produto tem um código, um nome, um preço e a quantidade existente. O código do produto é um inteiro que deve ser atribuído de forma automática cada vez que é criado um novo produto.

a) Para a classe produto defina o cabeçalho, os atributos e um construtor que receba como parâmetro o nome do produto.

b) Construa os getters e setters para os atributos da classe produto.

c) Para a classe Produto construa o método toString.

d) Para a classe Produto construa um método chamado subirProduto, que permita subir o preço do produto de uma dada percentagem dada como parâmetro. O valor do parâmetro deverá ser um valor entre 0 e 100. Se o valor do parâmetro estiver fora desse intervalo o método deverá gerar um exceção do tipo ValorInvalido. Quando é gerada a exceção deve ser-lhe associada a mensagem de erro: “O valor introduzido, xxx, não está entre 0 e 100”. Onde xxx representa o valor dado para o parâmetro do método. A exceção deverá ser capturada posteriormente na função que invocar o método.

e) Construa a classe de exceção ValorInvalido.

2) Construa uma classe de teste para a classe produto. Nessa classe deverá:

i) Construir um Produto com o nome “Portátil Asus” e preço 600€ e outro produto com o nome “Portátil HP” e preço 800€ .

ii) Aumentar o preço do Portátil Asus para 660€ usando o método subirProduto.

iii) Represente as variáveis que existem no programa que criou e qual o seu valor.

3 – Construa um método *public* e *static* que receba como parâmetro um objeto do tipo `java.util.ArrayList<Produto>`. O método deverá devolver o nome do produto mais caro que exista na `ArrayList`.

4 - Construa um método *public* e *static* que receba como parâmetro um objeto do tipo `java.util. ArrayList <Produto>` e uma `String` com o nome de um produto. O método deverá devolver o número de produtos que existem na lista de produtos com nome igual ao nome dado como parâmetro.

---

5 – Um produto alimentar é um Produto que tem um prazo de validade, isto é, uma data até à qual deverá ser consumido. A data deverá ser um objeto do tipo `LocalDate`.

a) Construa a classe `ProdutoAlimentar` definindo o cabeçalho, os atributos e um construtor que receba como parâmetros um objeto do tipo `Produto` e a sua data de validade.

b) Construa os métodos *get* e *set* para o atributo `dataValidade`.

c) Construa o método *equals* para a classe `ProdutoAlimentar`. Note que para a classe `Produto` não foi construído o método *equals* e portanto não poderá usá-lo para responder à pergunta.

6 – Considere a classe `Inventário` listada abaixo:

```
public class Inventario {  
    private ArrayList<Produto> produtos;  
    public Inventario () {  
        produtos = new ArrayList<> ();  
    }  
}
```

- Para a classe `Inventário`, construa um método que calcule e devolva o **valor total das existências de produtos alimentares fora de prazo**. Um produto fora de prazo é um produto cujo prazo de validade é inferior à data em que o programa é executado.

Nota:

*(Para comparar datas, use o método `compareTo` da classe `LocalDate` cuja descrição tem em anexo)*

## II

1 – Explique a diferença entre uma variável de instância e uma variável de classe? Dê um exemplo de cada.

2 - Explique a diferença entre sobrecarga (overloading) e sobreposição (overriding) de métodos. Dê um exemplo de cada.

3 – Explique a diferença entre uma classe abstrata e uma interface. Explique para que servem.

4 – Explique o que é uma exceção? Qual a diferença entre uma exceção verificável e uma exceção não verificável.

---

## Classe java.time.LocalDate

```
static LocalDate now ()
// Constrói um calendário com a data actual do seu computador.

static LocalDate of(int year, int month, int dayOfMonth)
//Constroi com a data dada nos parâmetros ano, mês, dia.

public boolean equals(Object obj)

public int compareTo (LocalDate d )
// compara dois calendários. Devolve 0 se a data que recebe a mensagem é igual à data
parâmetro. Devolve um valor <0 se a data que recebe a mensagem é menor que a data
parâmetro e devolve um valor >0 caso contrário.
```

## Classe java.util.ArrayList:

```
ArrayList() // construtor vazio, dimensão inicial zero.
boolean add(Object element)
// adiciona o elemento especificado ao final da lista
void add( int index, Object obj)
//insere o elemento especificado na posição index
Object remove(int index )//remove o elemento da posiçã index
boolean remove( Object o)
//remove a primeira ocorrência do objecto dado como parâmetro
Object set (int position, Object obj )
// substitui o elemento da posição index pelo elemento dado
Object get (int position)//devolve o elemento da posição index
void clear() // remove todos os elementos da lista
Object clone() // devolve uma cópia da lista
boolean contains(Object element)
// devolve true se a lista contém o elemento especificado
boolean equals ( Object obj)
// permite comparar duas listas
int indexOf(Object element)
// procura o índice da 1ª ocorrência de elemento
boolean isEmpty() // verifica se a lista não tem componentes
int size() // devolve a dimensão actual
String toString ()
```

---