

# Universidade da Beira Interior

## Programação Orientada a Objetos

*Cursos: 1º ciclo em Eng.ª Informática + 2º ciclo EEC*

**Exame, 2025/01/24**

---

**Sem consulta. Não é permitido durante o teste ter auricular, smartwatch ou telemóvel.**

**Duração: 120 minutos, 15.0 valores**

- 1** – Uma das características da programação orientada a objetos é a partilha de comportamento. Na linguagem Java que estudou, como é implementada essa característica?
  - 2** – Explique o que é um modificador de acesso, quais os que conhece em java, e o seu significado.
  - 3** – Explique o que é o polimorfismo de inclusão? Dê exemplos.
  - 4** – Qual a diferença entre uma interface e uma classe abstrata
  - 5** – O que é um tipo de dados genérico? Dê exemplos.
  - 6** – O que é um tipo referenciado em java? Quais os que conhece?
  - 7** – Pretende-se guardar informação sobre artigos científicos. Um artigo científico tem um título, um conjunto de autores, uma data de publicação e um número de páginas. Em anexo pode consultar a API das classes ArrayList e LocalDate.  
  
- Para uma classe que permita representar um artigo científico, construa:
    - a) O cabeçalho e os atributos da classe.
    - b) O construtor de omissão em que se pretende que nenhum atributo fique com o valor null.
    - c) Os getters, e os setters. Os setters dos atributos que sejam tipos referenciados não devem copiar apenas o endereço, mas o valor da nova variável a atribuir.
    - d) O método toString sobrepondo o método toString da classe Object.
    - e) O método equals sobrepondo o método equals da classe Object.
    - f) O método clone sobrepondo o método clone da classe Object.
    - g) Um método que dado um nome, como parâmetro, verifique se esse nome é autor do artigo. Em caso afirmativo deve devolver true, caso contrário devolve false.
    - h) Um método que permita adicionar um novo autor, dado como parâmetro, ao artigo, caso ainda não faça parte da sua lista de autores. Se o autor já fizer parte da lista, o método deve gerar a exceção AutorDuplicadoException com a mensagem de erro “o autor XXXX já existe”, onde XXXX deve ser o valor do autor que pretendeu adicionar.
  - 8** – Construa a classe de exceção AutorDuplicadoException.
  - 9** – Para uma classe de teste, construa um método, public static, que recebendo como parâmetros uma lista de artigos científicos e uma data, devolva uma lista com os artigos que sejam dessa data ou mais recentes.
-

**10** – Construa um bloco de código que:

- a) Crie um artigo científico, a1, com os seguintes dados: Título = “POO”; Autores: “P.P.” e “P.F”, data = 1/2/2022, número de páginas = 12.
- b) Crie um novo artigo científico, a2, fazendo uma cópia do artigo anterior com o método **clone**. De seguida deve testar se os dois objetos (a1 e a2) são iguais.
- c) Altere o título do artigo a2 para “POO2” e altere a data para 10/5/24.
- d) Adicione ao artigo a2 um novo autor, “MEF”, usando o método do exercício 7 h).
- e) Construa uma lista com os dois artigos e teste o método do exercício 9.

**11** – Um artigo publicado é um artigo científico ao qual foi atribuído um DOI (*Digital Object Identifier*) isto é, um identificador único e permanente para um elemento digital que é usado na Internet. Um DOI é um valor do tipo String. Para a classe ArtigoPublicado construa:

- a) O cabeçalho, os atributos e um construtor que receba como parâmetro um artigo científico.
- b) O método toString.
- c) O método equals.

**12** – Para uma classe de teste construa um método, public static, que recebendo como parâmetro uma lista de artigos científicos devolva quantos artigos da lista são artigos publicados.

**13** – Para uma classe de teste construa um método, public static, que recebendo como parâmetros uma lista de artigos científicos e um doi, devolva os autores do artigo correspondente ao doi, caso exista. Se o artigo não existir deve devolver o valor null.

---

### Classe `java.time.LocalDate`:

```
static LocalDate now ()  
// Constrói um calendário com a data actual do seu computador.  
static LocalDate of ( int ano, int mês, int dia)  
//Constroi o objeto com a data dada nos parâmetros ano, mês e 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 posição 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 position pelo elemento dado  
Object get (int index)//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 ()
```

---