Universidade de Aveiro
Departamento de Eletrónica, Telecomunicações e Informática (DETI)
Programação Orientada a Objetos (POO)
Avaliação em Aula (23/05/2024); duração: 1h30

Nome: **Nº Mec.**

Notas importantes:

1. Crie um projeto no seu computador com o seu número mecanográfico.
2. Descarregue os ficheiros de base para esta prova a partir do link de submissão existente no **elearning** "*POO Segunda avaliação em aula – P17*"
3. No final do teste, aceda de novo ao *elearning* e, no mesmo link, submeta o código do projeto compactando-o num único ficheiro com nome igual ao seu número mecanográfico.

Biblioteca

Escreva um programa que permita gerir uma biblioteca. A classe "**Book**" permite representar cada livro que faz parte da biblioteca. O programa deve usar Java Collections para gerir os livros.

Cada livro deve incluir os seguintes atributos:

- Identificador único (int) – deve ser automático, incremental
- Nome do livro (String)
- Autor (String)
- Editora (String)
- ISBN (String)
- Data de lançamento (LocalDate)

O programa deve conter, pelo menos, as seguintes classes:

- Book é a classe que representa cada livro. Esta classe deve ter os atributos mencionados acima, construtor(es), função toString e os setters e getters apropriados, hashCode, equals.
- BookManager é a classe responsável por gerir a biblioteca. Deve implementar os seguintes métodos:
    - addBook(Book b): Adicionar um novo livro à biblioteca. Não deve permitir que seja adicionado um livro mais do que uma vez.
    - removeBook(int id): Remove da lista o livro usando o seu identificador único.
    - getBook(int id): Obtém um livro com base no identificador único. Retorna o livro pedido ou *null* se o livro não existir.
    - calculateBookLoanCost(int days, int id): Uma classe que implementa a **interface IBookCostCalculator. O custo do empréstimo do livro é de 2,00€ para** empréstimos até 5 dias, para durações superiores acrescenta 0,75€ por cada dia extra.
    - printAllBooks(): Imprime a informação de todos os livros que estão na lista.
    - readFile(String file): Importa a informação de um conjunto de livros a partir de um ficheiro. Estes livros devem ser adicionados aos livros já existentes.

- writeFile(String file): Escreve a lista de livros existentes para um ficheiro. O ficheiro deve conter todos os dados armazenados na lista **separados por "tab"**.

Utilize o ficheiro BooksTester.java para testar e demonstrar o uso da classe BookManager *(não precisa de menu).* Pode editar este ficheiro livremente, mas deverá assegurar-se que este continua a testar as seguintes funcionalidades:

- Cria uma nova instância de BookManager.
- Adiciona livros ao sistema.
- Remove livros do sistema.
- Lê a partir de um ficheiro os dados referentes a vários livros.
  - O programa deve ser capaz de ler o ficheiro books.txt
- Obtém um livro específico.
- Lista todos os livros na biblioteca.
- Importar livros de um ficheiro:
- Exportar livros para um ficheiro:
  - Ao terminar a execução, o programa deve escrever o conteúdo atual da biblioteca para um ficheiro, respeitando o formato original.

Deve considerar ainda os seguintes requisitos:

- Não esquecer que não pode ser possível adicionar um livro que já exista, e para tal é necessário verificar se o nome ou o ISBN não existem na lista.
- Será necessário também validar o ISBN (tem de conter 13 dígitos, que podem surgir seguidos ou em 5 **grupos, separados por '-'; por exemplo** "978-972-0-04671-0").

Informação adicional sobre LocalDate

```
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
....
DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
LocalDate date = LocalDate.parse("2023-05-01", formatter);
System.out.println(date.format(formatter));
```