

Trabalho Prático: MyMediaManager

Regras de realização:

1. O trabalho é realizado em grupos de dois alunos.
2. Deve ser utilizada a linguagem Java e o IDE Visual Studio Code.
3. O código deve estar corretamente identificado, com o nome, número e curso dos alunos colocados em comentário no início do ficheiro.
4. Não podem ser utilizadas no trabalho funções e/ou estruturas de dados que não tenham sido abordadas nas aulas.
5. A entrega deve ser feita até às 24 horas do dia 28 de Dezembro, sendo as instruções de entrega divulgadas posteriormente.
6. Após a entrega haverá uma defesa oral do trabalho em data e hora a definir.
7. A não observação destas regras, ou a apresentação de trabalhos idênticos, pode implicar a anulação do(s) trabalho(s) ao(s) aluno(s) em causa.

Enunciado:

O MyMediaManager é a primeira versão de uma aplicação simples de gestão de conteúdos multimédia (filmes e séries). Um conteúdo ou item é descrito por um título (String), um tipo (filme ou série), um ano de lançamento (inteiro), informação sobre se já foi ou não visto e uma classificação opcional atribuída pelo utilizador (valor de 1 a 10; 0 significa “sem classificação”).

O programa deve permitir criar e editar os itens, marcar e desmarcar itens como vistos, atribuir/alterar classificações e visualizar a lista de diferentes formas (filtros e pesquisas).

A informação necessária à gestão da aplicação deve ser guardada em diversos arrays, assumindo uma lista com um máximo de 100 itens. A presença e utilização deste segmento de código no trabalho é **obrigatória**:

```
int tamMax = 100;

String[] titulo = new String[tamMax];
char[] tipo = new char[tamMax];
int[] ano = new int[tamMax];
boolean[] visto = new boolean[tamMax];
```

```
int[] rating = new int[tamMax];  
  
int nItens = 0;
```

Neste fragmento de código, **titulo** armazena o nome dos conteúdos; **tipo** guarda o tipo do conteúdo ('S' para série e 'F' para filme); **ano** contém o ano de lançamento do conteúdo e **visto** armazena booleanos que representam se o conteúdo foi ou não visto. A informação sobre o conteúdo **i** está armazenada nas posições **i** de todos os arrays.

Notem a presença da variável **nItens**, a qual deve ser mantida atualizada com o número de itens armazenados. Esta variável deve usada sempre que for necessário saber até que posição os nossos arrays estão preenchidos. Em nenhum caso se deve procurar uma posição a **null** para saber qual a primeira posição vazia.

Uma lista completa de tarefas deverá ser apresentada de forma semelhante à presente mais abaixo, utilizando impressão formatada (não usem tabs!).

Nº	Título	Tipo	Ano	Visto	Rating
1	The Matrix	F	1999	X	9
2	Breaking Bad	S	2008	X	10
3	Oppenheimer	F	2023		0
4	Dark	S	2017		0

Para facilitar o desenvolvimento, a lista de conteúdos pode ser inicializada à priori com alguns itens. Por exemplo, o código seguinte permite criar a lista de conteúdos impressa anteriormente:

```
titulo[0] = "The Matrix";    tipo[0] = 'F'; ano[0] = 1999; visto[0] = true; rating[0] = 9;  
titulo[1] = "Breaking Bad"; tipo[1] = 'S'; ano[1] = 2008; visto[1] = true; rating[1] = 10;  
titulo[2] = "Oppenheimer";  tipo[2] = 'F'; ano[2] = 2023; visto[2] = false; rating[2] = 0;  
titulo[3] = "Dark";         tipo[3] = 'S'; ano[3] = 2017; visto[3] = false; rating[3] = 0;  
  
nItens = 4;
```

As funcionalidades deverão estar organizadas num menu com vários níveis, sendo o primeiro apresentado de seguida:

(V)isualizar
(M)arcar / Classificar
(E)ditar
Es(t)atísticas
(S)air

As quatro primeiras opções deste menu permitem aceder aos submenus descritos de seguida, enquanto a opção (S)air leva o programa a terminar. A escolha da opção (V)isualizar produz o seguinte submenu:

Visualizar (t)odos
Visualizar por (a)no exato
Visualizar por (n)ão vistos
Visualizar por (p)alavra no título
Visualizar por (r)ating mínimo
Visualização (m)ulticritério
(V)oltar

As tarefas implementadas neste submenu são as seguintes:

Visualizar todos – apresenta uma lista de todas os conteúdos armazenados, num formato idêntico ao do exemplo acima.

Visualizar por (a)no exato – lista os conteúdos lançados num ano escolhido pelo utilizador.

Visualizar (n)ão vistos – lista os conteúdos que ainda não foram visualizados.

Visualizar por (p)alavra no título – lista os conteúdos cujo título contenha uma palavra inserida pelo utilizador.

Visualizar por (r)ating mínimo – lista os conteúdos que tenham uma avaliação igual ou superior a um valor introduzido pelo utilizador. Conteúdos sem rating (valor 0) não devem ser impressos.

Visualização (m)ulticritério – nesta opção o utilizador é questionado sobre quais os critérios que quer utilizar (tipo, ano, visto, rating) e os seus valores e são apresentados os conteúdos que satisfazem todos os critérios, permitindo assim ao utilizador aplicar, por exemplo, filtros como tipo = série, ano = 2004, visto = false.

Em qualquer dos submenus, a opção Voltar deve sempre permitir regressar ao menu anterior.

A opção (M)arcar permite aceder a um conjunto de funcionalidades (e respetivo submenu) relacionadas com a marcação de tarefas como realizadas ou não:

Marcar como (v)isto por posição
Marcar como visto por (t)ítulo
(D)esmarcar último marcado como visto
(A)tribuir / alterar rating por número
(V)oltar

As tarefas implementadas no submenu Marcar são as seguintes:

Marcar como (v)isto por posição – O utilizador escolhe a posição de um conteúdo nos arrays, marcando o mesmo como visto, mudando para true a posição correspondente no array visto.

Marcar como visto por (t)ítulo - O utilizador deve introduzir um String de texto e o primeiro conteúdo que o contiver na descrição será marcado como visto.

(D)esmarcar último marcado como visto – O último conteúdo marcado pelo utilizador como visto deve voltar ao estado não visto.

(A)tribuir / alterar rating por posição – Permite atribuir ou alterar a classificação do conteúdo armazenado na posição escolhida pelo utilizador

O submenu Editar agrupa as funcionalidades que permitem adicionar e eliminar conteúdos:

(A)dicionar item no fim
Adicionar (i)tem na posição n
Apagar item na (p)osição n
Apagar itens (v)istos
(V)oltar

(A)dicionar item no fim – Permite adicionar um novo filme ou série no fim da lista de conteúdos, recebendo para tal os dados do utilizador.

Adicionar (i)tem na posição n – Semelhante à anterior mas adicionalmente solicita ao utilizador uma posição da lista, já ocupada, devendo mover os itens já existentes de forma a inserir o novo item na posição escolhida.

Apagar item na (p)osição n – Apaga um item em posição escolhida pelo utilizador. Deve reorganizar a lista de forma a não deixar uma posição em branco.

Apagar itens (v)istos – Apaga todos os itens marcados como realizados. Não devem, de novo, ser deixadas posições na lista por ocupar.

Por fim, o submenu Estatísticas agrupa opções que permitem sumariar e apresentar informação estatística referente à nossa lista de conteúdos:

(P)ercentagem de vistos
(M)édia de rating dos vistos
Distribuição por (d)écada
(V)oltar

P)ercentagem de vistos – Calcula e imprime a percentagem de conteúdos vistos em relação ao total apresentado na lista.

(M)édia de rating dos vistos – Calcula e imprime a média das classificações dos conteúdos vistos.

Distribuição por (d)écada – Apresenta contagens dos conteúdos armazenados por década de lançamento, por exemplo:

1990-1999: 3 itens
2000-2009: 5 itens
2010-2019: 2 itens
2020-2029: 4 itens

Este trabalho destina-se a exercitar a utilização de estruturas de controlo, arrays, ciclos e Strings, bem como métodos definidos pelo programador. A utilização de métodos deve ir de encontro às vantagens dos métodos discutidas nas aulas, nomeadamente o reaproveitamento de código, a sua simplificação, a modularização de funcionalidades, etc...

Para atingir esses objetivos, esta versão do MyMediaManager deve obedecer aos seguintes requisitos (ver os critérios de avaliação):

- No método main apenas deverá estar presente a definição das estruturas de dados principais e a chamada ao método que implementa o menu principal (por exemplo, chamado apresentaMenuPrincipal).
- Cada menu secundário deverá ser implementado (e não só impresso) no seu próprio método. Isto obriga a que todos os arrays de dados necessários tenham de ser passados como argumentos - não pode haver variáveis globais (de classe). Tem também de haver cuidado com a atualização da variável nItens, possivelmente devolvendo-a como valor de retorno dos métodos em que seja alterada.
- As funcionalidades de cada menu devem ser implementadas em métodos separados. Sempre que possível, deve haver separação entre a entrada de dados e o método que implementa a funcionalidade, para que este possa ser chamado em situações diversas, evitando a repetição de código.

- Métodos que calculam valores devem evitar imprimir o resultado, o qual deve ser devolvido e impresso após a chamada do método.
- Devem ser feitos métodos que implementem as entradas protegidas de dados, especialmente para chars e ints e doubles. A proteção deve incluir o tipo e o domínio (intervalo) do parâmetro a ler. A primeira camada de proteção deve garantir a leitura de um dado do tipo correto. Por exemplo, se o utilizador introduzir caracteres quando o programa espera um inteiro, este não deve “estourar”. A segunda camada deve garantir o que o valor lido, por exemplo um inteiro, está no intervalo válido. Pode haver ainda uma terceira camada que imprima uma mensagem e use o método anterior para fazer a leitura. Os métodos devem devolver os valores lidos.
- Os métodos devem ser genéricos, recorrendo aos parâmetros de entrada necessários para garantir essa característica.
- Métodos que executem a mesma funcionalidade, mas necessitem de parâmetros diferentes, devem ser implementados por *overloading*.

Qualquer dúvida ou questão adicional deverá ser colocada aos docentes da disciplina.

Critérios de avaliação (para um total de 20 valores):

1. Indentação, organização do código e nomes das variáveis adequados: 1.0 valor.
2. Comentários adequados: 1.0 valores.
3. Estrutura geral do programa, incluindo passagens entre menus: 2.0 valores.

4. Funcionalidade Visualizar todos: 0.5 valores.
5. Funcionalidade Visualizar por ano exato: 0.5 valores.
6. Funcionalidade Visualizar por não vistos: 0.5 valores.
7. Funcionalidade Visualizar por palavra no título: 0.5 valores.
8. Funcionalidade Visualizar por (r)ating mínimo: 0.5 valores.
9. Funcionalidade Visualização (m)ulticritério: 1.0 valor.

10. Funcionalidade Marcar como visto por posição: 0.5 valores.
11. Funcionalidade Marcar como visto por título: 0.5 valores.
12. Funcionalidade Desmarcar último marcado como visto: 1.0 valores.
13. Funcionalidade Atribuir / alterar rating por número: 0.5 valores.

14. Funcionalidade Adicionar item no fim: 0.5 valores.
15. Funcionalidade Adicionar item na posição n: 1.0 valor.
16. Funcionalidade Apagar item na posição n: 1.0 valor.
17. Funcionalidade Apagar itens vistos: 1.5 valores.

18. Funcionalidade Percentagem de vistos: 0.5 valores.
19. Funcionalidade Média de rating dos vistos: 0.5 valores.
20. Funcionalidade Distribuição por década: 1.0

21. Utilização adequada de métodos: 2.0 valores.
22. Métodos de leitura e proteção de dados: 2.0 valores.



**Politécnico
Castelo Branco**

Escola Superior
de Tecnologia

Escola Superior de Tecnologia
Cursos de 1º Ciclo em Engenharia Informática /
Tecnologias da Informação e Multimédia
1º Semestre do 1º Ano
Unidade Curricular: Introdução à Programação / Algoritmia
e Programação
Docente: Arlindo Silva, Nuno Pinho, Ricardo Serra
Ano Letivo 2025/2026
Trabalho Prático
Data de Entrega: 28/12/2025

A nota final será pesada por uma defesa oral do trabalho. Quer isto dizer que uma nota inicial de 20 pode terminar em 0, caso, por exemplo, um aluno revele um total desconhecimento do código do seu próprio trabalho.