ISEL	ESTRUTURAS DE DADOS E ALGORITMOS
Sumário: Ficha 2	Duração: 2 Aulas

PARTE I

Esta ficha consiste na criação de uma classe para representar uma **matriz de inteiros** com as operações típicas que se realizam com matrizes. Os elementos da matriz devem ser guardados num *array* dinâmico de duas dimensões de acordo com a ordem da matriz.

Ficheiro Matriz.h

```
#pragma once
// Definição da classe Matriz que contem as operações
// tipicas de matrizes

class Matrix {
private:
	int **elems;
	int nLines;
	int nCols;

	// Funções privadas
	void Delete();                      // Apagar a memória dinâmica criada para a matriz
	void Create(int lines, int cols); // Criar a memória dinâmica e inicializar a matriz

public:
	Matrix();                   // Construtor de defeito
	Matrix(const Matrix& m1);   // Construtor Copy
	Matrix(int lines, int cols);
	~Matrix();           // Destrutor

	bool   CanMultiply(const Matrix* pm);
	bool   CanAdd(const Matrix* pm);

	const Matrix& operator = (const Matriz& m1);
	Matrix operator + (const Matriz& m1);
	Matrix operator * (const Matriz& m1);
	const Matrix& operator += (int k);     // Soma cada elemento da matriz com o valor k
	const Matrix& operator *= (int k);     // Multiplica cada elemento da matriz por k

	void Transpose();
	bool ReadFromFile(char* file_name);
	bool SaveToFile(char* file_name);
	void Output();
};
```

A classe deve designar-se por ***Matrix*** e possuir os seguintes atributos privados:

- ***elems*** – elementos da matriz.
- ***nLines*** – número de linhas da matriz.
- ***nCols*** – número de colunas da matriz.

A classe deve conter:

- Um construtor que recebe a ordem da matriz e cria uma matriz com elementos a zero.
- Um construtor cópia que cria uma nova matriz a partir de outra já existente.
- Um construtor por omissão (a ordem da matriz é definida posteriormente).
- Um destrutor para eliminar os dados dinâmicos da classe.

E os seguintes métodos:

- ***ReadFromFile*** – Lê os dados da matriz de um ficheiro(1).
- ***SaveToFile*** – Grava os dados da matriz para um ficheiro(2).
- ***Output*** – Escreve no ecrã o conteúdo da matriz.
- ***CanAdd*** – Verifica se duas matrizes podem ser somadas.
- ***CanMultiply*** – Verifica se duas matrizes podem ser multiplicadas.
- ***operador atribuição* (=)** – atribuição de matrizes(3).
- ***operador soma* (+)** – soma de matrizes(4) e soma dos elementos da matriz por um valor constante.
- ***operador produto* (*)** – produto de matrizes(4) e produto dos elementos da matriz por uma constante.
- ***Transpose*** – Deve realizar a operação de transpor a própria matriz, i.e., deve trocar as linhas pelas colunas.

Deve realizar uma função main que chame estas funções definidas na classe Matriz.

Notas:

(1) Esta função deve receber como parâmetro de entrada o nome do ficheiro e devolver um valor booleano para indicar se a leitura foi correta ou não. O formato do ficheiro deve ser o seguinte:

Ficheiro Matriz.txt

```
# Comentário       // linhas iniciadas com # são consideradas comentário
2  2          // 1ª linha com o nº de linhas e colunas da matriz
1            // 2ª linha e seguintes com cada elemento da matriz
2
-3
0
```

(2) Esta função deve gravar um ficheiro de texto com o formato indicado em (1).

(3) O conteúdo da matriz da direita é copiado para a matriz da esquerda. Se a matriz da esquerda contiver dados, estes devem ser removidos previamente.

(4) As matrizes devem ser de dimensões compatíveis com a operação a realizar.