Departamento de Matemática da Universidade de Coimbra		
2003/2004	**Métodos de Programação II**	**1**

1 Tipo de Dados Compostos, Tabelas ("Array")

Exercício 1.1 Escreva um sub-programa que, dados um inteiro positivo k e um vector de n inteiros positivos, calcule a média dos múltiplos de k e a média dos submúltiplos de k, existentes no vector.

Exercício 1.2 São dados os n elementos inteiros de um vector x (n constante e igual a 100) e ainda um valor inteiro k. Escreva um programa para imprimir todos os pares de números x_i, x_j, tais que $x_i + x_j = k$.

Exercício 1.3 Escreva um programa para:

1. ler o inteiro n;
2. ler os $n \times n$ elementos inteiros de uma matriz A;
3. contar quantos elementos nulos existem acima da diagonal principal;
4. se o número de elementos nulos for par, escrever a matriz, caso contrário escrever a sua transposta.

Exercício 1.4 Faça sub-programas para:

1. Somar duas matrizes $A(m, n)$ e $B(m, n)$.
2. Multiplicar duas matrizes $A(m, n)$ e $B(n, k)$.
3. Dada uma matriz $A(m, n)$ de elementos reais, determinar a linha cuja soma dos seus elementos é máxima.
4. Calcular o determinante de uma matriz triangular $A(n, n)$ de elementos reais.

Exercício 1.5 Escreva um programa que, dada uma matriz A de $m \times n$ elementos reais, determine quantos são superiores ao valor da média de todos os elementos da matriz.

Exercício 1.6 Um treinador de atletismo treina 5 atletas e faz 12 sessões de treino por semana. Em cada sessão, cada atleta percorre uma distância que é cronometrada. Os valores dos tempos, em segundos, são registados sob a forma de uma matriz $T(5 \times 12)$, onde cada linha diz respeito a um atleta e cada coluna a uma sessão de treino. Supondo já feita a leitura da matriz, escreva uma secção de programa para:

1. calcular e escrever a média dos tempos realizados em cada sessão de treinos;
2. determinar e escrever o melhor tempo realizado por cada um dos atletas nas 12 sessões.

Exercício 1.7 Escreva um programa que, dada uma matriz quadrada de ordem n, $(0 < n \leq 100)$ de elementos inteiros, e dados dois inteiros k e l, devolva a matriz após troca entre si das colunas k e l.

Exercício 1.8 Numa matriz $M(35 \times 12)$ de elementos inteiros, encontram-se registadas as notas dos 35 alunos de cada uma das 12 turmas de Métodos de Programação. Sabendo que todos os elementos de M são valores entre 0 e 20, escreva um programa que:

1. determine e escreva o número de alunos aprovados (nota ≥ 10);
2. determine e escreva a melhor nota em cada uma das turmas;
3. identifique a turma com maior número de alunos aprovados.

Exercício 1.9

1. Dada uma matriz de inteiros, de tipo $m \times n$ $(m, n \leq 100)$, elabore um sub-programa para fazer uma permutação para a esquerda das colunas da matriz.
2. Escreva o correspondente programa de chamada.

Exercício 1.10 Considere o seguinte procedimento:

```
procedure trocar (i, j: integer);
var aux: real;
begin
    if j>i then begin
        aux:=x[i];
        x[i]:=x[j];
        x[j]:=aux;
        trocar(i+1,j-1)
    end
end;
```

1. Sendo **x** um vector de **n** elementos reais, diga o que sucede ao vector **x** se o programa principal contiver a instrução: `trocar(1,n)`.

2. Elabore um procedimento não recursivo para o mesmo efeito.

Exercício 1.11 Escreva, usando um algoritmo recursivo, um sub-programa que, dados um elemento **k** e um vector de **n** ($0 < \mathbf{n} \leq 100$) elementos inteiros, determine o número máximo de ocorrências de **k** no vector.

Exercício 1.12 Dado um vector de n inteiros e um inteiro k fazer um sub-programa que procure k nas n primeiras posições do vector. O resultado deste sub-programa deve ser a posição onde o elemento está, ou uma indicação de falha. (No caso de haver repetições o sub-programa pode fornecer como resultado qualquer uma das posições).

No pior caso quantos elementos do vector é que este sub-programa tem de examinar ?

Exercício 1.13 Faça um sub-programa que encontre o maior dos n primeiros elementos de um vector de inteiros.

Exercício 1.14 Utilizando o sub-programa anterior ordene os primeiros n elementos do vector usando o seguinte algoritmo:

1. encontre o maior dos n elementos;

2. coloque esse elemento na sua posição final trocando-o com o enésimo elemento;

3. neste momento para ordenar todo o vector basta ordenar as $n-1$ primeiras posições do vector.

Note-se que a recursão implícita na descrição anterior pode ser facilmente eliminada.
(Este algoritmo de ordenação chama-se **selecção linear**.)

Exercício 1.15 Outro algoritmo de ordenação é a adaptação de um usado pelos jogadores de cartas.

Consideramos o vector como a concatenação de duas sequências: uma ordenada, e a segunda não ordenada. No início a primeira sequência contém apenas um elemento.

O primeiro elemento da sequência não ordenada é inserido na posição correcta da sequência ordenada (movendo os elementos maiores uma posição para a esquerda).

Esta operação aumentou a sequência ordenada de um elemento (e retira esse elemento à não ordenada). Repetimos até todos os elementos estarem na sequência ordenada.

Note-se que a inserção pode ser simplificada usando uma sentinela.
(Este algoritmo de ordenação chama-se **inserção linear**.)

Exercício 1.16 Dado um vector de n inteiros e um inteiro x existente no vector pretende-se particionar esse vector em duas partes: os elementos inferiores a x e os elementos superiores a x.

Uma maneira de fazer isto é:

1. Começando do lado esquerdo do vector procurar um elemento que seja maior ou igual a x.

2. Começando do lado direito do vector procurar um elemento que seja menor ou igual a x.

3. Trocar estes dois elementos.

4. Continuar até as duas procuras se cruzarem.