Algoritmia e

Programação

Estruturas de dados indexadas.



Matrizes: variáveis indexadas

* Alguns tipos de problemas têm dados que se representam naturalmente na forma de variáveis indexadas, por exemplo:

•

•

•

contar número de parafusos pelo seu diâmetro,

contar respostas correctas por questão de um teste,

contar número de automóveis que passaram numa ponte, por hora do dia, etc…



**APROG - LETI**

Exemplo: automóveis por hora, numa ponte…

* Do ponto de vista lógico, estes 24 valores representam partes de um conceito maior.
  + Faz sentido estarem agrupados numa "super-variável"!



**veiculos**

15  2  1

**0 h 1 h 2 h**

523  201 324  298  502  389

**10 h 11 h 12 h 13 h 14 h 15 h**

138  107  32

**21 h 22 h 23 h**



**APROG - LETI**

Declarar uma matriz

•

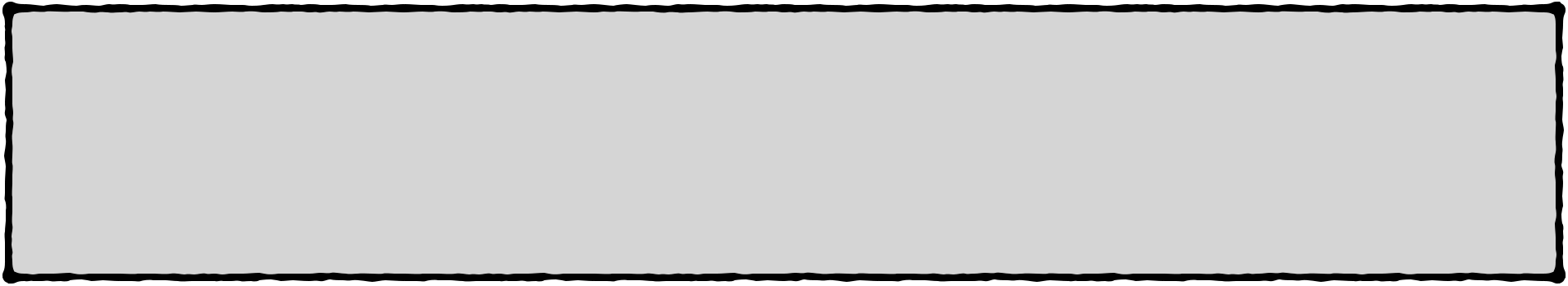
•

•

Uma matriz armazena valores do mesmo tipo.

Uma matriz tem dimensão fixa.

A sua declaração reflecte estas duas características:



**ESTRUTURAS DE DADOS**

**INTEIRO:** matriz\_inteiros[20]

**REAL:** matriz\_reais[3]



**APROG - LETI**

Acesso aos elementos

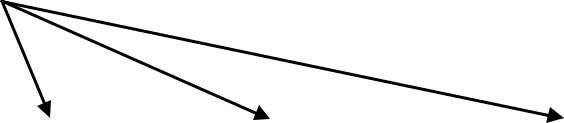
* O acesso a cada **elemento** da variável é realizado através do seu **índice**.
  + Cada índice é **único**.
  + Os índices começam em **zero** e são **sequenciais**.



**APROG - LETI**

Exemplo: acesso aos elementos…

**Índices**



|  |  |  |  |
| --- | --- | --- | --- |
| **[0]** | | **[1]** | **[2]** |
|  |  |  |  |
| **veiculos** | 15 | 2 | 1 |
|  |  |  |  |
|  | **0 h** | **1 h** | **2 h** |

**[10][11][12][13][14][15]**



523  201 324  298  502  389

**10 h 11 h 12 h 13 h 14 h 15 h**

**[21][22][23]**

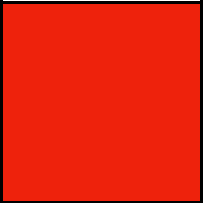
138  107  32

**21 h 22 h 23 h**



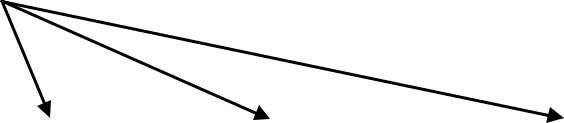
**APROG - LETI**

Exemplo: acesso aos elementos…



**veiculos[11]** 201

**Índices**



**veiculos**

**[0] [1] [2]**

15  2  1

**0 h 1 h 2 h**

**[10][11][12][13][14][15]**



523  201 324  298  502  389

**10 h 11 h 12 h 13 h 14 h 15 h**

**[21][22][23]**

138  107  32

**21 h 22 h 23 h**



**APROG - LETI**

Vantagens das matrizes

•

•

Permitem agrupar sob um único nome um conjunto de elementos do mesmo tipo, relacionados entre si.

Permitem definir de forma concisa código que executa repetitivamente a mesma operação sobre cada um dos elementos (uso de ciclos).



**APROG - LETI**

Exemplo: calcular a média de automoveis/hora.

**ESTRUTURAS DE DADOS**

**INTEIRO:** i, total, veiculos[24]

**REAL:** media\_veiculos



**APROG - LETI**

Exemplo: calcular a média de automoveis/hora.

**ALGORITMO**

(...)

total = 0

**PARA** i = 0 **ATÉ** 23 **FAZER**

total = total + veiculos[i]

**FIM PARA**

media\_veiculos = total / 24

**ESCREVER**(media\_veiculos)

**FIM**



**APROG - LETI**

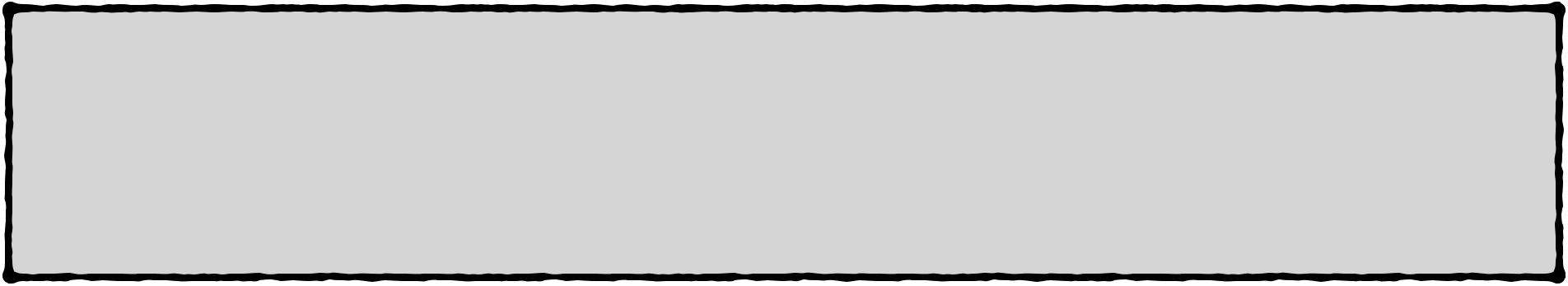
Matrizes bidimensionais

•

•

É possível também utilizar matrizes bidimensionais.

A forma de indexar os elementos de uma matriz bidimensional é através de um par de índices:



**ESTRUTURAS DE DADOS**

**INTEIRO:** matriz\_plana[4][10]

* Neste exemplo, a matriz tem 4 linhas e 10 colunas. Tem capacidade para 40 inteiros.



**APROG - LETI**

Implementação em C



**APROG - LETI**

Declaração de matrizes

* A declaração de uma matriz é semelhante à de uma variável, sendo necessário acrescentar as dimensões.



**int main ()**

**{**

**int** veiculos[24], matriz\_plana[4][10]; **float** matriz\_reais[3];

/\* (...) \*/

}



**APROG - LETI**

Acesso aos elementos da matriz

* O acesso aos elementos da matriz faz-se através do(s) seu(s) índice(s).



**int main ()**

**{**

**int** veiculos[24], i, total;

**float** media;

/\* (...) \*/

**for**(i = 0; i < 24; i++) {

total += veiculos[i];

}

media = (**float**) total / 24;

**printf**("Média de veículos por hora: %.1f\n", media); **return** 0;

}

**APROG - LETI**

Cuidados a ter!!!

* A linguagem C não impõe limites no acesso aos elementos de uma matriz.
  + É possível pedir para aceder a um índice superior à capacidade da matriz!

•

•

Esta situação provoca erros, por permitir o acesso a dados que NÃO SÃO correctos.

* dever do programador assegurar-se que o programa não ultrapassa os limites das matrizes (*buﬀer overflow*).



**APROG - LETI**

Exercício



**APROG - LETI**

Exercício

* Elabore um algoritmo que pede ao utilizador 10 valores inteiros.

•

•

1. O algoritmo deve apresentar o menor valor (*mínimo*) e o maior valor (*máximo*) inseridos.
2. O algoritmo deve apresentar para cada valor a sua razão com o menor valor, i.e.:

*valori / mínimo*



**APROG - LETI**