

INF01202

Algoritmos e Programação

Modalidade Ensino a Distância

Linguagem C

Tópico 11: Arranjos multidimensionais

Material desenvolvido pelas professoras:

- 2009/I

Maria Aparecida Livi, Maria Aparecida Souto
e Magda Bercht

Arranjos Multidimensionais

- A linguagem C nos permite trabalhar com arranjos de várias dimensões:
- Uma dimensão → vetores
 - Duas dimensões → tabelas ou matrizes
 - Três dimensões
 - ou mais ...

Vetor

10	15	5	...	18
0	1	2	...	n

Matriz bidimensional

Alunos	Notas →			
	Id	Nota1	Nota2	Nota3
↓	001/08	8.5	7.0	10.0
	002/08	6.0	8.0	9.5
	003/08	7.5	6.0	9.8

Matriz tridimensional

Alunos	Turmas	Notas →			
		Id	Nota1	Nota2	Nota3
↓	1
	2

Até agora ...

- Você trabalhou com **vetores**, os quais constituem um caso particular de arranjo, e que têm muitas aplicações importantes na Ciência da Computação.
- Os arranjos de duas dimensões, ou **matrizes**, ou **tabelas**, também constituem um caso particular de arranjos, e também têm aplicações importantes na área.
- A seguir, você irá estudar os arranjos bidimensionais.

Matrizes

- São arranjos bidimensionais, nos quais as informações são organizadas em **linhas** e **colunas**.

	Coluna 0	Coluna 1	Coluna 2	Coluna 3
Linha 0	M [0] [0]	M [0] [1]	M [0] [2]	M [0] [3]
Linha 1	M [1] [0]	M [1] [1]	M [1] [2]	M [1] [3]
Linha 2	M [2] [0]	M [2] [1]	M [2] [2]	M [2] [3]

Índice ou subscrito da **coluna**

Índice ou subscrito da **linha**

nome da matriz

Uma matriz requer 2 subscritos para identificar um elemento particular.

Observe que os elementos de uma linha todos têm o primeiro subscrito iguais.

Observe que os elementos de uma coluna todos têm o segundo subscrito iguais.


Declaração de Matrizes em C

int **m1** [3] [2];

- Declara uma matriz de inteiros de 3 linhas e 2 colunas.


int **m2** [2] [2] = {{1, 2}, {3, 4}};

- Declara uma matriz quadrada 2 x 2 de inteiros e a inicializa da seguinte forma:


$$\mathbf{m2} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$


int **m3** [2] [2] = { {1}, {3, 4} };

- Declara uma matriz quadrada 2 x 2 de inteiros e a inicializa da seguinte forma:


$$\mathbf{m3} = \begin{bmatrix} 1 & 0 \\ 3 & 4 \end{bmatrix}$$

float **m4** [2] [3] = {{1.0, 2.0, 3.0},
 {4.0, 5.0, 6.0}};

- Declara uma matriz 2 x 3 de valores tipo float e a inicializa da seguinte forma:


$$\mathbf{m4} = \begin{bmatrix} 1.0 & 2.0 & 3.0 \\ 4.0 & 5.0 & 6.0 \end{bmatrix}$$

Operações básicas com matrizes: *leitura e impressão*

➤ Seja a matriz:

```
int m [4] [4];
```

$$m = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{pmatrix}$$

➤ Quais os procedimentos para:

- Ler a matriz **m** por linhas?
- Ler a matriz **m** por colunas?
- Imprimir a matriz **m** por linhas?
- Imprimir a matriz **m** por colunas?

Código em C: *leitura* de matriz

➤ Por linhas

```
for (i = 0; i < 4; i++)  
    for (j = 0; j < 4; j++)  
        scanf ("%d", &m[i] [j]);
```

Neste caso, para cada valor do índice *i* de linha, o índice *j* de coluna varia em toda sua extensão.

➤ Por colunas

```
for (j = 0; j < 4; j++)  
    for (i = 0; i < 4; i++)  
        scanf ("%d", &m[i] [j]);
```

Neste caso, para cada valor do índice *j* de coluna, o índice *i* de linha varia em toda sua extensão.

**Vejamos a animação a seguir sobre
como acontece a leitura de arranjos
multidimensionais
(ou matrizes, tabelas, quadros)**

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
  for (j=0;j <MAXCOL;j++)  
  {  
    scanf("%d", &notas[i] [j]);  
  }  
}
```

i --> 0

j --> 0

notas[0] [0] <-- 9

	0	1	2
0	9		
1			
2			

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
  for (j=0;j <MAXCOL;j++)  
  {  
    scanf("%d", &notas[i][j]);  
  }  
}
```

i --> 0

j --> 1

notas[0][1] <-- 5

	0	1	2
0	9	5	
1			
2			

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)
```

```
{
```

```
  for (j=0;j <MAXCOL;j++)
```

```
  {
```

```
    scanf("%d", &notas[i] [j]);
```

```
  }
```

```
}
```

i --> 0

j --> 2

notas[0] [2] <-- 7

	0	1	2
0	9	5	7
1			
2			

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
    for (j=0;j <MAXCOL;j++)  
    {  
        scanf("%d", &notas[i] [j]);  
    }  
}
```

i --> 0

j --> 3

(fim do for j)

	0	1	2
0	9	5	7
1			
2			

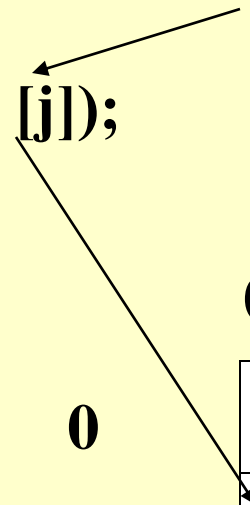
Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
  for (j=0;j <MAXCOL;j++)  
  {  
    scanf("%d", &notas[i][j]);  
  }  
}
```

i --> 1

j --> 0

notas[1][0] <-- 11



	0	1	2
0	9	5	7
1	11		
2			

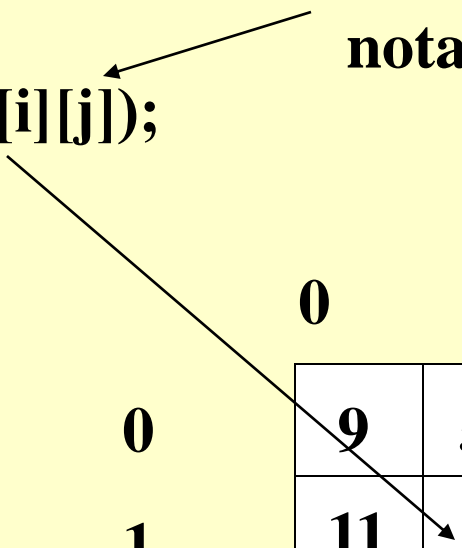
Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
  for (j=0;j <MAXCOL;j++)  
  {  
    scanf("%d", &notas[i][j]);  
  }  
}
```

i --> 1

j --> 1

notas[1][1] <-- 4



	0	1	2
0	9	5	7
1	11	4	
2			

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
  for (j=0;j <MAXCOL;j++)  
  {  
    scanf("%d", &notas[i][j]);  
  }  
}
```

i --> 1

j --> 2

notas[1][2] <-- 1

	0	1	2
0	9	5	7
1	11	4	1
2			

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
    for (j=0;j <MAXCOL;j++)  
    {  
        scanf("%d", &notas[i] [j]);  
    }  
}
```

i --> 1
j --> 3
(fim do for j)

	0	1	2
0	9	5	7
1	11	4	1
2			

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
  for (j=0;j <MAXCOL;j++)  
  {  
    scanf("%d", &notas[i][j]);  
  }  
}
```

i --> 2

j --> 0

notas[2][0] <-- 6

	0	1	2
0	9	5	7
1	11	4	1
2	6		

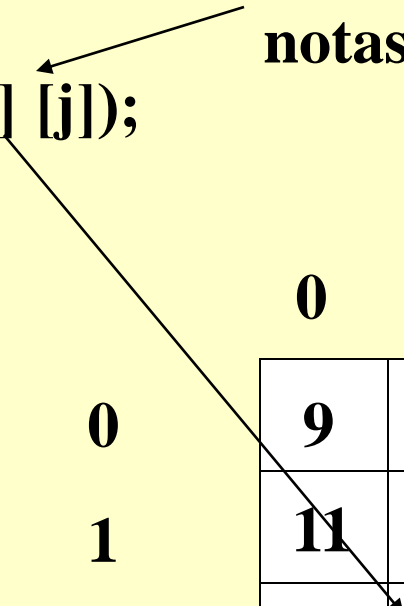
Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
  for (j=0;j <MAXCOL;j++)  
  {  
    scanf("%d", &notas[i][j]);  
  }  
}
```

i --> 2

j --> 1

notas[2][1] <-- 3



	0	1	2
0	9	5	7
1	11	4	1
2	6	3	

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
  for (j=0;j <MAXCOL;j++)  
  {  
    scanf("%d", &notas[i][j]);  
  }  
}
```

i --> 2

j --> 2

notas[2][2] <-- 2

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
    for (j=0;j <MAXCOL;j++)  
    {  
        scanf("%d", &notas[i][j]);  
    }  
}
```

i --> 2
j --> 3
(fim do for j)

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Leitura da matriz notas 3 X 3

```
for (i=0;i<MAXLIN; i++)  
{  
    for (j=0;j <MAXCOL;j++)  
    {  
        scanf("%d", &notas[i] [j]);  
    }  
}
```

i --> 3
j --> 0
(fim do for i)

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Código em C: *impressão de matriz*

➤ Por linhas

```
for (i = 0; i < 4; i++)  
{  
    for (j = 0; j < 4; j++)  
        printf ("%d", m[i] [j]);  
    printf ("\n\n");  
}
```

Neste caso, para cada valor do índice *i* de linha, o índice *j* de coluna varia em toda sua extensão.

➤ Por colunas

```
for (j = 0; j < 4; j++)  
{  
    for (i = 0; i < 4; i++)  
        printf ("%d", m[i] [j]);  
    printf ("\n\n");  
}
```

Neste caso, para cada valor do índice *j* de coluna, o índice *i* de linha varia em toda sua extensão.

**Veja a seguir a animação
Sobre a escrita da
matriz notas 3 X 3**

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)
```

```
{
```

```
    printf("\nLinha %d" , i);
```

```
    for (j=0;j <MAXCOL;j++)
```

```
    {
```

```
        printf("\n%d", notas[i] [j]);
```

```
    }
```

```
}
```

i --> 0

j --> 0

notas[0] [0] --> 9

Linha 1
9

0

1

2

0

1

2

	9	5	7
	11	4	1
	6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)
```

```
{
```

```
    printf("\nLinha %d", i);
```

```
    for (j=0;j <MAXCOL;j++)
```

```
    {
```

```
        printf("\n%d", notas[i] [j]);
```

```
    }
```

```
}  
Linha 1  
9  
5
```

i --> 0

j --> 1

notas[0] [1] --> 5

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)
```

```
{
```

```
    printf("\nLinha %d", i);
```

```
    for (j=0;j <MAXCOL;j++)
```

```
    {
```

```
        printf("\n%d", notas[i] [j]);
```

```
    }
```

5
7

i --> 0

j --> 2

notas[0] [2] --> 7

0

1

2

0	1	2
9	5	7
11	4	1
6	3	2

Escrita da matriz
notas

```
for (i=0;i<MAXLIN; i++)
{
    printf("\nLinha %d" , i);
    for (j=0;j <MAXCOL;j++)
    {
        printf("\n%d", notas[i] [j]);
    }
}
```

i --> 0

j --> 3

(fim do for j)

5
7

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)
```

```
{
```

```
    printf("\nLinha %d" , i);
```

```
    for (j=0;j <MAXCOL;j++)
```

```
    {
```

```
        printf("\n%d", notas[i] [j]);
```

```
    }
```

```
} 7  
  Linha 2  
  11
```

i --> 1

j --> 0

notas[1] [0] --> 11

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)
```

```
{
```

```
    printf("\nLinha %d", i);
```

```
    for (j=0;j <MAXCOL;j++)
```

```
    {
```

```
        printf("\n%d", notas[i] [j]);
```

```
    }
```

```
}  
Linha 2  
11  
4
```

i --> 1

j --> 1

notas[1] [1] --> 4

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Escrita da matriz
notas

```
for (i=0;i<MAXLIN; i++)
```

{

```
printf("\nLinha %d" , i);
```

```
for (j=0;j <MAXCOL;j++)
```

{

```
printf("\n%d", notas[i] [j]);
```

}

11
4
1

i --> 1

j --> 2

notas[1] [2] --> 1

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)  
{  
    printf("\nLinha %d", i);  
    for (j=0;j <MAXCOL;j++)  
    {  
        printf("\n%d", notas[i] [j]);  
    }  
}
```

i --> 1

j --> 3

(fim do for j)

11
4
1

0

1

2

0	1	2
9	5	7
11	4	1
6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)
```

```
{
```

```
    printf("\nLinha %d", i);
```

```
    for (j=0;j <MAXCOL;j++)
```

```
    {
```

```
        printf("\n%d", notas[i] [j]);
```

```
    }
```

```
} 1  
  Linha 3  
  6
```

i --> 2

j --> 0

notas[2] [0] --> 6

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)  
{  
    printf("\nLinha %d" , i);  
    for (j=0;j <MAXCOL;j++)  
    {  
        printf("\n%d", notas[i] [j]);  
    }  
}
```

i --> 2

j --> 1

notas[2] [1] --> 3

Linha 3
6
3

0

1

2

	0	1	2
0	9	5	7
1	11	4	1
2	6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)  
{  
    printf("\nLinha %d", i);  
    for (j=0;j <MAXCOL;j++)  
    {  
        printf("\n%d", notas[i] [j]);  
    }  
}
```

notas[2][2] --> 2

6
3
2

0
1
2

0	1	2
9	5	7
11	4	1
6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)  
{  
    printf("\nLinha %d", i);  
    for (j=0;j <MAXCOL;j++)  
    {  
        printf("\n%d", notas[i] [j]);  
    }  
}
```

i --> 2

j --> 3

(fim do for j)

6	
3	
2	

0

1

2

0	1	2
9	5	7
11	4	1
6	3	2

Escrita da matriz notas

```
for (i=0;i<MAXLIN; i++)  
{  
    printf("\nLinha %d", i);  
    for (j=0;j <MAXCOL;j++)  
    {  
        printf("\n%d", notas[i] [j]);  
    }  
}
```

i --> 3

j --> 0

(fim do for i)

6	
3	
2	

0

1

2

0	1	2
9	5	7
11	4	1
6	3	2

Álgebra Matricial: Soma e Multiplicação

- Seja o trecho de código que declara as matrizes A, B e C:

```
...  
#define MAX 100  
int A [MAX] [MAX];  
int B [MAX] [MAX];  
int C [MAX] [MAX];  
...
```

- Como efetuar?

- A Soma: $C = A_{mn} + B_{mn}$
- A Multiplicação: $C = A_{mp} \times B_{pn}$

Código C: Soma de Matrizes

➤ Soma:

```
for (i = 0; i < MAX; i++)  
    for (j = 0; j < MAX; j++)  
        C [i] [j] = A [i] [j] + B [i] [j];
```

Código C: Multiplicação de Matrizes

➤ Multiplicação:

$$C_{ij} = \sum_{k=0}^{n-1} A_{ik} * B_{kj}$$

```
for (i = 0; i < MAX; i++)  
    for (j = 0; j < MAX; j++)  
    {  
        s = 0;  
        for (k = 0; k < MAX; k++)  
            s += A [i] [k] * B [k] [j];  
        C [i] [j] = s;  
    }
```

Aplicação de Matriz:

- Exemplo Comentado -

- Dada uma matriz M ($MAXLIN$, $MAXCOL$), preenchê-la por leitura e imprimir:
 - o maior elemento de cada linha da matriz;
 - a média dos elementos de cada coluna;
 - o produto de todos os elementos diferentes de zero;
 - quantos elementos são negativos;
 - posição ocupada (linha-coluna) por um elemento cujo valor será lido pelo programa.

- Exemplo Comentado -

Declarações, inicialização e impressão da matriz

```
// Inserir cabeçalho ...
#include <stdio.h>
#include <stdlib.h>
#define MAXLIN 10
#define MAXCOL 3
int main ()
{ int m [MAXLIN] [MAXCOL];    //declaracao da matriz
  int lin, col; //indices de linha e coluna
  int maior, somacol, produto, negativos, valor, achou;
  float mediacol;
  for (lin = 0; lin < 10; lin++)    //inicializacao da matriz
    for (col = 0; col < 3; col++)
    {   printf ("Forneca valor inteiro m [%d] [%d]: \n", lin, col);
        scanf ("%d", &m[lin] [col]);
    }
  for (lin = 0; lin < 10; lin++)    //impressão da matriz
  {
    printf ("Linha %d\t", lin);
    for (col = 0; col < 3; col++)
      printf ("\t%d", m[lin] [col]);
    printf ("\n\n");
  }

  //continua ...
```

- Exemplo Comentado - Procurando o maior elemento de cada linha...

```
for (lin = 0; lin < 10; lin++) //procurando o maior valor de cada linha
{
    maior = m [lin] [0]; //supõe que o 1º. Eh o maior
    for (col = 1; col < 3; col++)
        if (maior < m[lin] [col]) //compara maior com os outros
            maior = m[lin] [col]; //se existir algum > que maior, troca
    printf ("\n\nMaior valor da linha %d eh %d", lin, maior);
} //continua ...
```

- Exemplo Comentado - calculando a média dos valores de cada coluna...

```
for (col = 0; col < 3; col++) //calculando a media da coluna
{
    somacol = 0; //a cada coluna, inicializa somacol
    for (lin = 1; lin < 10; lin++)
        somacol += m[lin] [col]; //acumula valores da coluna
    printf ("\n\nA media da coluna %d eh %.2f", col, (float)somacol/MAXLIN);
}
//continua ...
```

- Exemplo Comentado - calculando o produto dos valores != 0 ...

```
produto = 1; //inicializa em 1, pois eh produto!  
for (lin = 0; lin < 10; lin++) //calculando o produto dos valores != 0  
    for (col = 0; col < 3; col++)  
        if ( m[lin] [col] != 0 )  
            produto *= m[lin] [col];  
printf ("\n\nO produto dos valores != 0 eh: %d", produto);
```

//continua ...

- Exemplo comentado - calculando o total de valores negativos ...

```
negativos = 0; //inicializa em 0, pois eh contador!  
for (lin = 0; lin < 10; lin++) //calculando o total de negativos  
    for (col = 0; col < 3; col++)  
        if ( m[lin] [col] < 0 )  
            negativos += m[lin] [col];  
printf ("\n\nO total de valores < 0 eh: %d", negativos);  
//continua ...
```

- Exemplo Comentado -

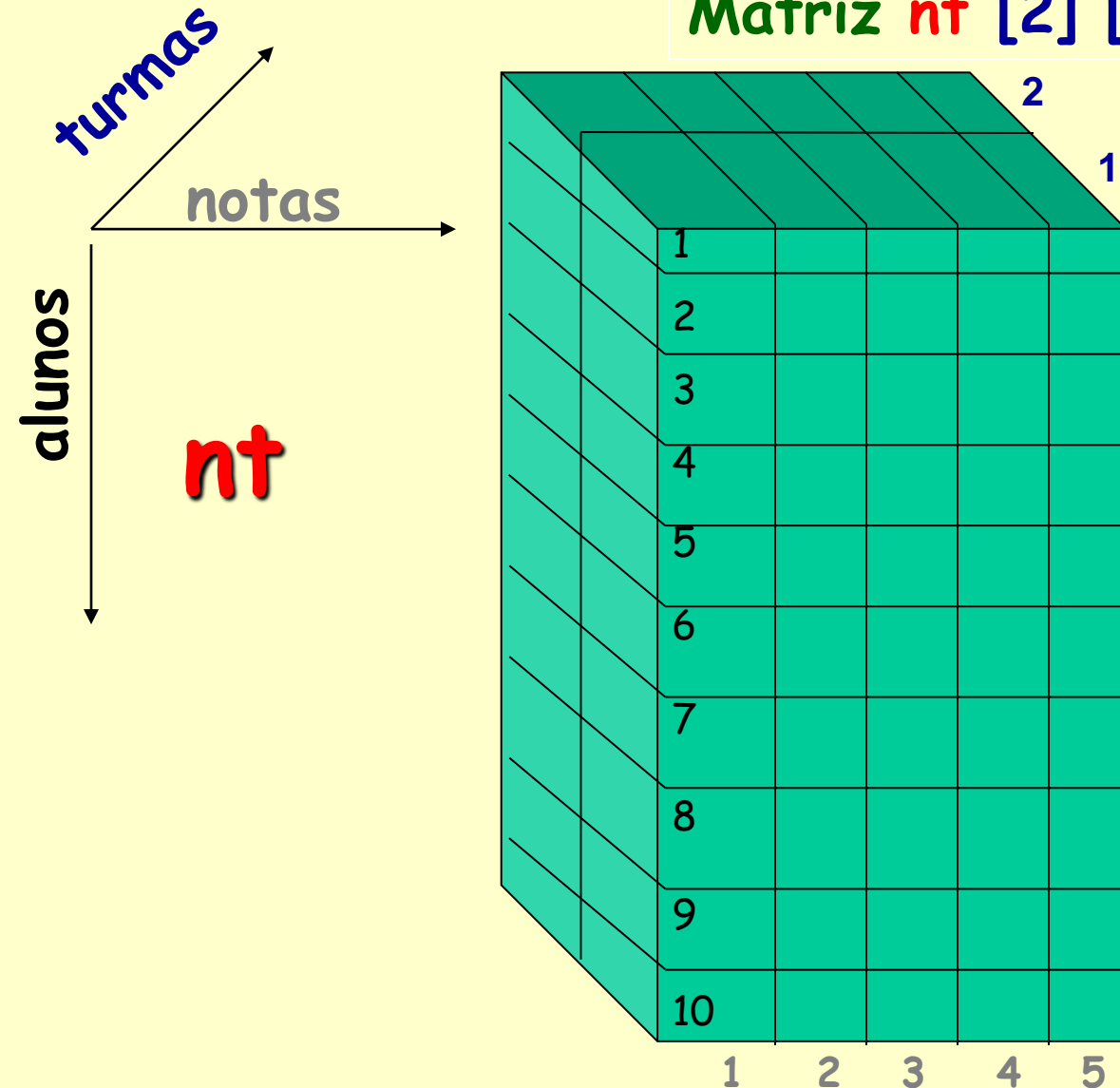
procurando a posição ocupada por um elemento
cujo valor será lido pelo programa

```
printf ("\n\nForneca valor a ser procurado: ");
scanf ("%d", &valor);
achou = 0;
for (lin = 0; lin < 10; lin++) //procurando o valor fornecido ...
    for (col = 0; col < 3; col++)
        if ( m[lin] [col] == valor )
        {
            achou = 1;
            printf ("\n\nValor estah em: [%d] [%d]",lin, col);
        }
if (!achou)
    printf ("\n\nValor não encontrado!");
printf ("\n\n");
system ("pause");
} //fim de main
```

Arranjos com mais de 2 dimensões ...

Ex: Duas turmas
10 alunos em cada turma
5 notas para cada aluno
Ler e armazenar as notas de cada aluno.

Matriz **nt** [2] [10] [5]



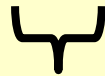
Arranjo 3 dimensões em C

Declaração

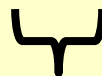
tipo nome do arranjo [dim] [dim] [dim];

Ex:

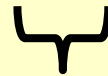
float nt [2] [10] [5];



1ª dimensão



2ª dimensão



3ª dimensão

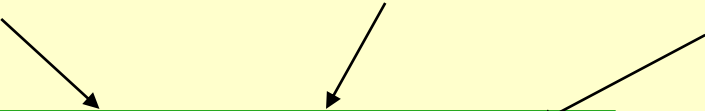
Arranjo 3 dimensões em C

Utilização

- um índice para cada dimensão
- respeitando a ORDEM da declaração

Ex:

turmas alunos notas



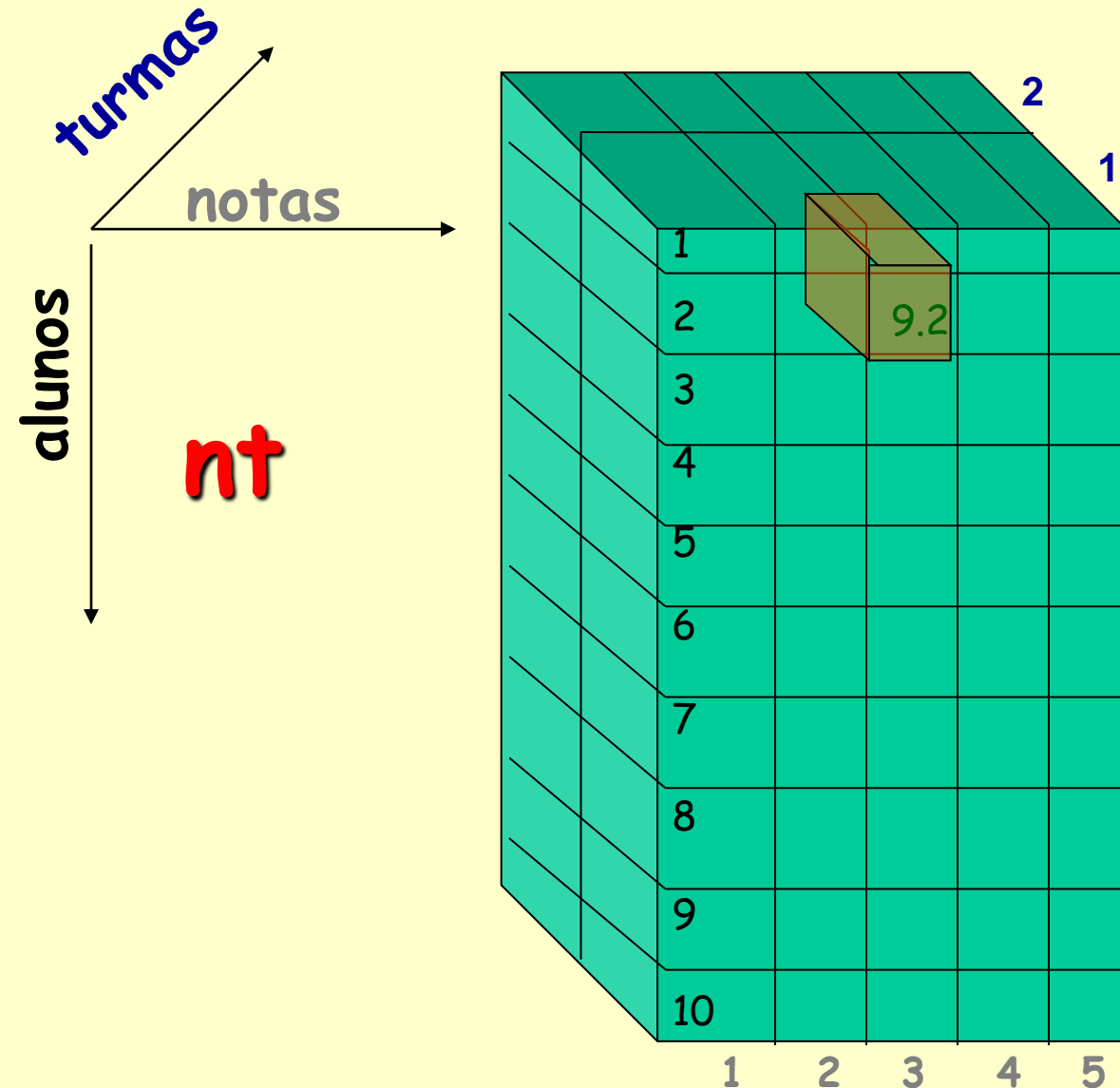
```
float nt [ 2 ] [ 10 ] [ 5 ];
```

referência



```
nt [turma] [aluno] [nota]
```

Ex: Duas turmas
10 alunos em cada turma
5 notas para cada aluno
Ler e armazenar as notas de cada aluno.



`nt [1] [2] [3] = 9.2;`

Ex: 10 lojas
5 setores por loja
30 produtos por setor
12 meses

4 dimensões

O arranjo armazena quantidade de itens de cada produto vendido em cada mês, por setor e por loja.

```
float quantvend [ 11 ] [ 6 ] [ 31 ] [ 13 ];
```

loja setor produto mês



ATENÇÃO: para utilizar os códigos de loja, setor, produto e mês, diretamente como índices da matriz, todas as dimensões devem ser declaradas com um elemento a mais, uma vez que os índices de arranjos na Linguagem C partem de 0!

...

```
// listar quantidade vendida do produto 17, em janeiro,  
// em todos os setores da loja 6
```

```
for (setor = 1; setor <= 5; setor++)  
    printf ("%4d", quant_vend [6] [setor] [17] [1] );
```

...

- Exemplo Comentado -

Ex: Rede de 10 lojas, 40 produtos, armazenar a quantidade de itens de cada produto vendidos nos últimos 12 meses. Códigos dos produtos armazenados em arranjo separado.

Algoritmo Lojas:

1. Preencher por leitura todos os dados (códigos dos produtos e matriz de vendidos).
2. Informar quantas unidades de cada produto foram vendidas em uma determinada loja (fornecida) no último mês.
3. Informar quantas unidades de um determinado produto (informado seu código) foram vendidas em cada loja nos últimos 6 meses.
4. Informar qual a loja que vendeu a maior quantidade de um determinado produto (fornecido seu código) em um determinado mês (fornecido).
5. Informar os códigos de todos os produtos dos quais não foi vendida nenhuma unidade nos últimos 2 meses.
6. Entrar com os dados de vendas de um novo mês, que será o último. Com isso, todos os dados relativos aos meses anteriores 'avançam' uma posição, e os dados do primeiro mês (o mais antigo) são perdidos
7. ... (determine novas operações sobre estes dados)

Ex: Rede de 10 lojas, 40 produtos, armazenar a quantidade de itens de cada produto vendidos nos últimos 12 meses. Códigos dos produtos armazenados em arranjo separado.

1. Preencher por leitura todos os dados (códigos dos produtos e matriz de vendidos)

```
#include <stdio.h>
int main ( )
{
    int codprod [41];
    int vendido [11] [41] [13]; //loja/prod/mes
    int lj, p, m;           // indices para as 3 dimensoes
    int lojalida;           // para armazenar numero relativo a loja
    int codlido; // para armazenar codigo de produto lido

    for (lj = 1; lj < 11; lj++) //carga do arranjo: para cada loja
        for (p = 1; p < 41; p++) //todos os produtos
        {
            scanf("%d", &codprod [p]);
            for (m = 1; m<13; m++) //em todos os meses
                scanf("%d", &vendido[lj] [p] [m]);
        }

    ...
    for (p = 1; p < 41; p++) //carga do arranjo: para cada produto
    {
        scanf("%d", &codprod [p]);
        for (lj = 1; lj < 11; lj++) //todas as lojas
            for (m = 1; m<13; m++) //em todos os meses
                scanf("%d", &vendido[lj] [p] [m]);
    }

    ...
}
```

Alternativa **1** de carga do arranjo

ou

Alternativa **2** de carga do arranjo

2. Informar quantas unidades de cada produto foram vendidas em uma determinada loja (fornecida) no último mês
3. Informar quantas unidades de um determinado produto (informado seu código) foram vendidas em cada loja nos últimos 6 meses.

```
#include <stdio.h>
int main ( )
{
    int codprod [41];
    int vendido [11] [41] [13]; //loja/prod/mes
    int lj, p, m;           // indices para as 3 dimensoes
    int lojalida;           // para armazenar numero relativo a loja
    int codlido; // para armazenar codigo de produto lido
    int quant, buscap;      // contador e variável auxiliar
    scanf( "%d", &lojalida ); // 2 - Obtem loja a ser analisada
    for (p=1; p<41; p++)
        printf ("Produto %d - loja %6d", codprod[p], vendido[lojalida] [p] [12]);

    scanf( "%d", &codlido ); // 3 - obtem o produto a ser analisado e busca o indice p
    for (buscap =1; buscap < 41; buscap++) // nao preve que codlido seja inexistente
        if (codprod[p] == codlido)
            p = buscap;
    for (lj = 1; lj <11; lj++) // para cada loja
    {
        quant = 0; // inicializa o contador para cada loja
        for (m= 7; m < 13 ; m++)
            quant = quant + vendido[lj] [p] [m];
        printf( "Loja %d - vendidos %5d", lj, quant)
    }
} // fim de main
```