SI100

Algoritmos e Programação de Computadores I

1º Semestre - 2018

Tópico 6 – Matrizes

Unidade 10

Prof. Guilherme Palermo Coelho

guilherme@ft.unicamp.br

Roteiro

- Introdução;
- Matrizes Bidimensionais;
- Matrizes N-dimensionais;
- Linearização de Matrizes;
- Exercícios;
- Referências.

INTRODUÇÃO

Introdução

- Foi visto nas aulas anteriores que a linguagem C é capaz de trabalhar com estruturas de dados chamadas ***vetores***;
  - **Vetor**: é um conjunto de posições consecutivas de memória, identificadas por um mesmo nome, individualizadas por índices e cujo conteúdo é do mesmo tipo;
  - **Exemplo**: vetor de inteiros

```
...
int notas[10]; /* declara vetor de inteiros
                *  com 10 posições */
notas[2] = 7;  /* atribui o valor 7 à
                *  posição 3 do vetor */
...
```

  - A linguagem C permite vetores de qualquer tipo → até mesmo ***vetores de vetores*** (que serão chamados aqui de ***matrizes***).

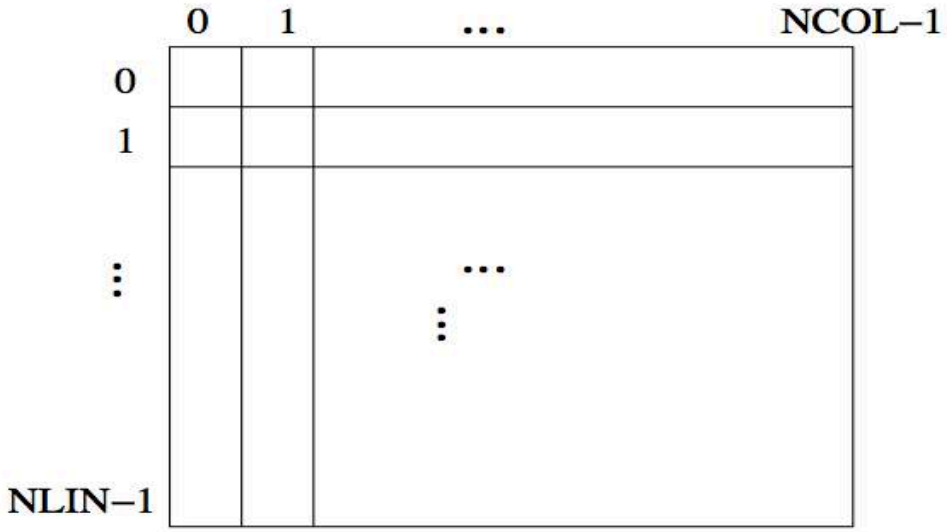
Introdução

- A capacidade de representação e manipulação de matrizes em um programa é muito útil em diversas aplicações, tais como:
  - Armazenamento de imagens, onde cada imagem pode ser considerada uma *matriz de pixels*;
  - Realização de cálculos envolvendo álgebra linear;
  - Armazenamento de dados de clientes;
  - ...
- A forma mais simples de uma matriz é a ***bidimensional***, que pode ser vista como uma organização de dados em uma "tabela" com um dado número de linhas e de colunas.
- Também existem matrizes com mais dimensões.

MATRIZES BIDIMENSIONAIS

Matrizes Bidimensionais

- Correspondem às matrizes que estamos acostumados a ver no ensino básico;
  - **Exemplo**: matriz bidimensional com NLIN linhas e NCOL colunas




```
#define NLIN 80
#define NCOL 100

int main() {
  int m[NLIN][NCOL];

}
```

- Neste exemplo, **m** é um vetor de elementos que, por sua vez, são vetores de valores do tipo inteiro.

Matrizes Bidimensionais

- Exemplo de aplicação: **soma de matrizes quadradas**

```
#include <stdio.h>

#define N 20

int main() {
        int m1[N][N], m2[N][N], m3[N][N];
        int l, c, nlin, ncol;

        printf("Entre com os num de linhas e colunas\n");
        scanf("%d %d",&nlin,&ncol); /* nlin e ncol < 20*/

        printf("Entre com os elementos da matriz 1\n");
        for (l=0; l < nlin; l++) //percorre linhas
                for (c=0; c < ncol; c++) //percorre colunas
                        scanf("%d",&m1[l][c]);
// continua
```

Matrizes Bidimensionais

- Exemplo de aplicação: **soma de matrizes**

```
// continuação:
        printf("Entre com os elementos da matriz 2\n");
        for (l=0; l < nlin; l++) //percorre linhas
                for (c=0; c < ncol; c++) //percorre colunas
                        scanf("%d",&m2[l][c]);

        /* soma as matrizes */
        for (l=0; l < nlin; l++)
                for (c=0; c < ncol; c++)
                        m3[l][c] = m1[l][c] + m2[l][c];
// continua
```

Note que a utilização de matrizes bidimensionais é análoga ao uso de vetores visto nas aulas anteriores, exceto pelo índice extra correspondente à segunda coordenada.

Matrizes Bidimensionais

- Exemplo de aplicação: **soma de matrizes**

```
// continuação:
        /* imprime o resultado */
        printf("Resultado: \n");

        for (l=0; l < nlin; l++) {
                for (c=0; c < ncol; c++)
                        printf("%2d ",m3[l][c]);
                printf("\n");
        }

        return 0;
}
```

Matrizes Bidimensionais

- Da mesma forma que vetores unidimensionais, matrizes também podem ser **inicializadas** na própria declaração, com os valores colocados entre "{" "}" e separados por vírgulas;
  - No entanto, é importante notar que cada elemento deste vetor **também será um vetor**, e deve ser inicializado como tal;

```
#include <stdio.h>

#define N 3

int main() {
        int m1[N][N] = { {0, 1, 2},
                         {3, 4, 5},
                         {6, 7, 8} };

        ...
        return 0;
}
```

Valores correspondentes à primeira linha da matriz

Valores correspondentes à segunda linha da matriz

Valores correspondentes à terceira linha da matriz

MATRIZES N-DIMENSIONAIS

Matrizes N-dimensionais

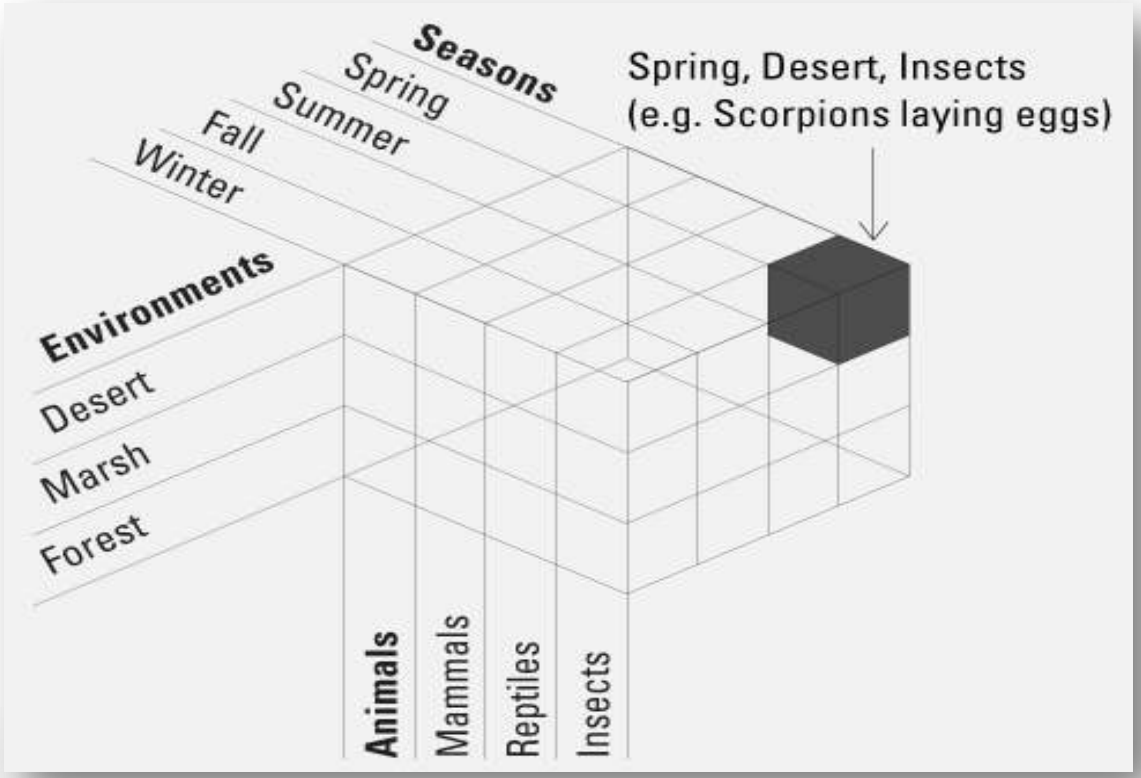
- Pela definição, vimos que uma matriz bidimensional é um vetor cujos elementos são vetores de dados (inteiros, caracteres etc.);
- Podemos estender esta definição para ***matrizes N-dimensionais***:
  - **Tridimensionais**: vetores cujos elementos são matrizes bidimensionais (vetores de vetores de dados);

```
int main() {
  int m[2][2][3] = {{{1, 2, 3}, {3, 2, 1}},
                    {{5, 6, 7}, {8, 9, 9}}};
}
```

  - Quadridimensionais ...

Matrizes N-dimensionais

- Ilustração de uma matriz tridimensional:

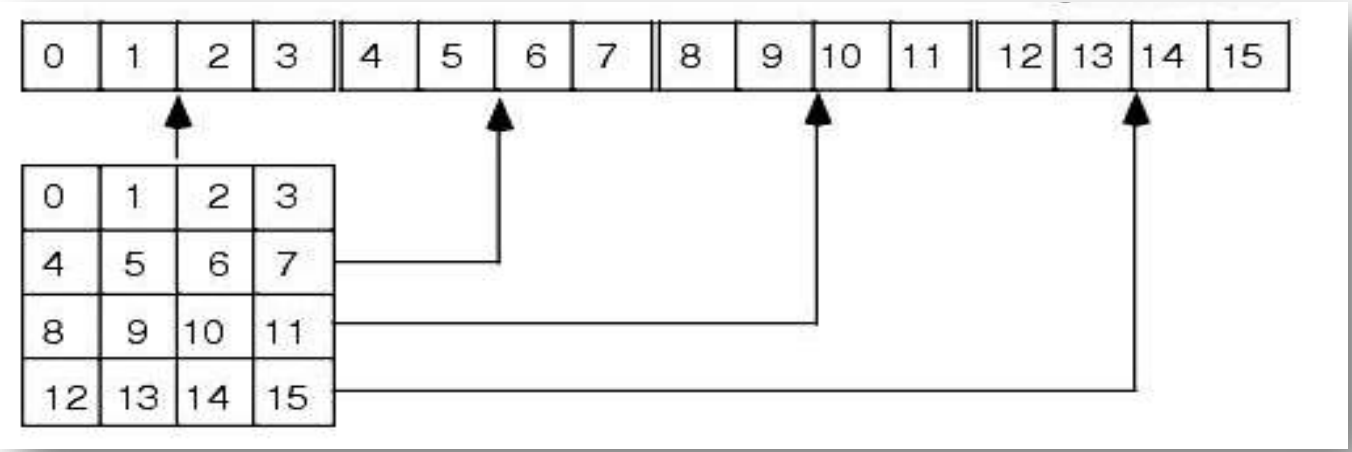


Fonte: http://goo.gl/RQwkQ

Linearização de Matrizes

Linearização de Matrizes

- Algumas vezes desejamos armazenar matrizes de dados em vetores unidimensionais:
  - Esta é a forma como geralmente matrizes são armazenadas em memória;

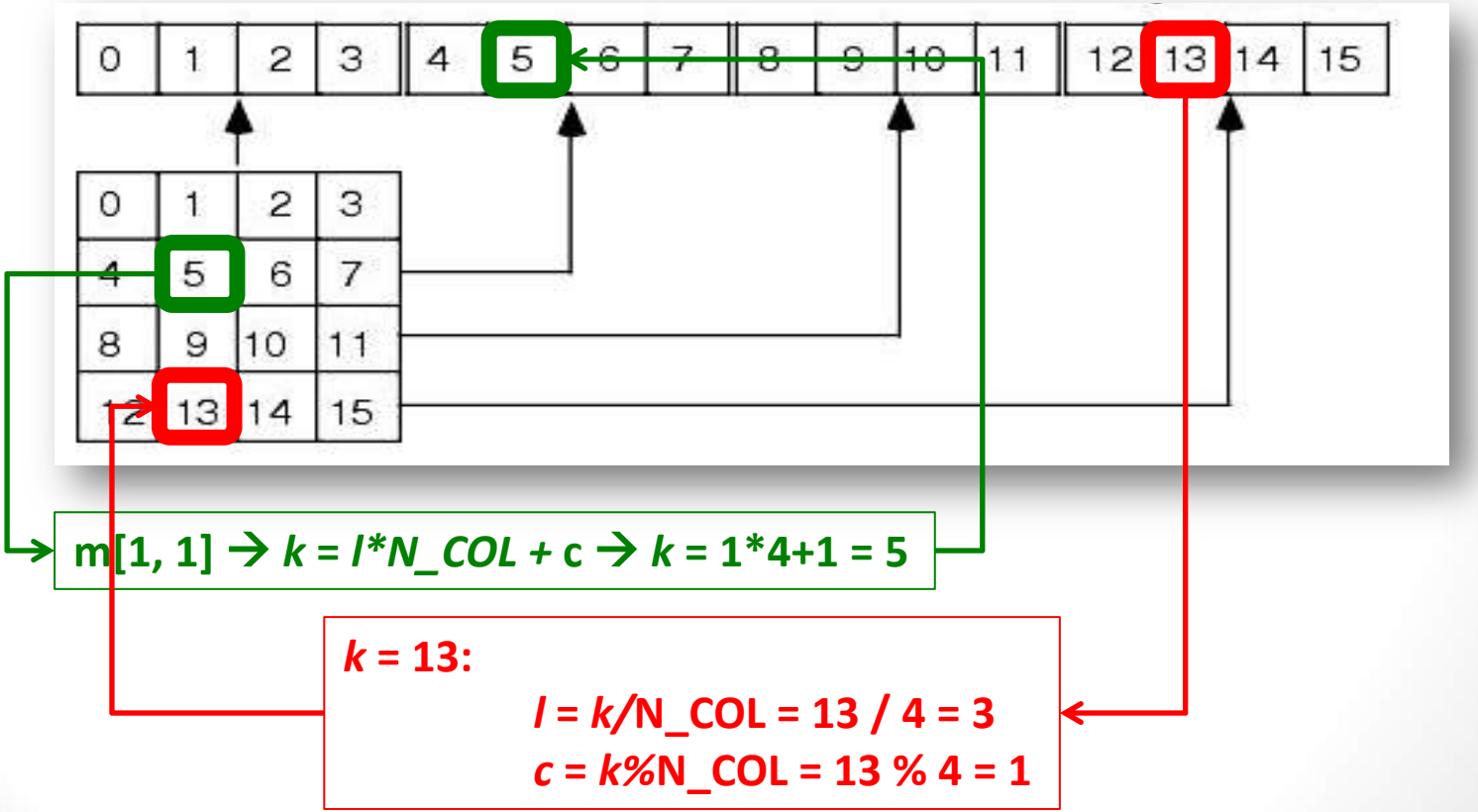


Linearização de Matrizes

- Sendo assim, como converter as coordenadas de uma matriz para a posição de um vetor?
- Suponha uma matriz *m* bidimensional:
  - Com N_LIN linhas e N_COL colunas;
  - Dados são armazenados linha a linha (da linha 0 até a linha N_LIN-1) sequencialmente no vetor (como figura no slide anterior);
- O elemento *m*[*l*, *c*] da matriz será armazenado na posição:
  - *k* = *l**N_COL + *c*;
- O elemento *k* do vetor corresponderá ao elemento *m*[*l*, *c*] da matriz, onde:
  - *l* = k / N_COL
  - *c* = k % N_COL

Linearização de Matrizes

- **Exemplo**: N_LIN = 4, N_COL = 4



EXERCÍCIOS

Exercícios

1. Escreva um programa que receba do usuário as dimensões NUM_LIN e NUM_COL (máximo 20) e os dados de uma matriz bidimensional, e imprima a matriz transposta na tela.
2. Escreva um programa que receba do usuário as dimensões NUM_LIN e NUM_COL (não maiores que 20) e os dados de uma matriz bidimensional, converta-a em um vetor e imprima na tela os valores armazenados neste vetor.
3. Escreva um programa que leia as dimensões e os dados de duas matrizes bidimensionais (máximo 20 linhas e 20 colunas) e imprima na tela o resultado da multiplicação destas duas matrizes. Caso as dimensões das duas matrizes não permitam a multiplicação, o usuário deverá ser notificado.

Exercícios

- **Observação**:
  - Os seguintes exercícios devem ser entregues via SuSy.
    - Exercício 1;
    - Exercício 2;
    - Exercício 3.
  - Veja os enunciados atualizados no site do sistema:
    - https://susy.ic.unicamp.br:9999/si100a (Turma A);
    - https://susy.ic.unicamp.br:9999/si100b (Turma B).

REFERÊNCIAS

Referências

- MIZRAHI, V. V., *Treinamento em Linguagem C – Curso Completo*. 2a Edição, Pearson Makron Books, 2005.
- MIZRAHI, V. V., *Treinamento em Linguagem C – Curso Completo – Módulo 2*. 2a Edição, Pearson Makron Books, 2005.
- ASCENCIO, A. F. G. & DE CAMPOS, E. A. V., *Fundamentos da Programação de Computadores – Algoritmos, Pascal e C/C++*. Pearson Prentice Hall, 2003.