Algoritmos e Programação de Computadores

Matrizes

Prof. Daniel Sundfeld Lima
daniel.sundfeld@unb.br

Introdução

- Muitas vezes, vetores são utilizados para agrupar dados relevantes
- É possível declarar matrizes, que são vetores com duas dimensões (ou mais)
- Para isso, podemos usamos a notação com colchetes

Introdução

- Declara um vetor de 10 inteiros:
- int vetor[10];
- Declara uma matriz de 10x10:
- int matriz[10][10];
- Declara uma matriz 10x10x10:
- Int matriz[10][10][10];

Introdução

- Para acessar os dados de uma matriz, devemos utilizar os colchetes para cada dimensão
- int i = 0;
- int cub[5][5][5];
- cub[3][1][9] = i;
- Lembrando: assim como vetores não há verificação do limite de vetores. Mesmo para matrizes.

Matrizes

- Vetores e matrizes não são utilizados para agrupar dados diferentes.
- Ex: crie uma matriz que armazene a idade (inteiro) e nota (float) dos alunos. Não é possível fazer essa declaração
- Nesse caso, não usamos matrizes, mas tipos de dados definidos pelo usuário

Matrizes

- Apesar de utilizarmos 2 dimensões, a memória do computador é linear e possui apenas uma dimensão
- Por isso, as matrizes são uma abstração na linguagem de programação, o uso de múltiplas variáveis cria a ilusão que trabalhamos com múltiplas dimensões

Matrizes

(0,0)

1	2	3
4	5	6
7	8	9

(2,2)

(0,0) (1,0) (2,0)

1	2	3	4	5	6	7	8	9

Matrizes

- A linguagem de programação C é **row-major order**
- Isso quer dizer, que as células são adjacentes na memória: (0, 0) e (0, 1) são adjacentes. Mas (0, 0) e (1, 0) não serão em uma matriz 3x3.

Matrizes

Row-major order

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Column-major order

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

- Fonte: Wikipedia.

Matrizes

- Vetores e matrizes podem ser inicializados utilizando loops e atribuições
- for (int i = 0; i < 10; i++)
  vet[i] = i;
- for (int i = 0; i < 10; i++)
  for (int j = 0; j < 10; j++)
  mat[i][j] = i + j;

Exercício

- Faça um programa que leia uma dimensão N, e preencha os dados de uma matrix NxN
- Depois, imprima essa matriz

Matrizes

- A inicialização pode ser feita durante a declaração da varíavel
- int vet[4] = {0, 1, 2, 3};
- Atenção! Isso só pode ser feito durante a inicialização!
- int vet[4];
  vet = {0, 1, 2, 3}; //ERRADO!!

Matrizes

- As matrizes seguem a mesma lógica que os vetores, para inicializar. No entanto, é muito recomendado (mas não obrigatório) que se utilize {} adicionais para separar as dimensões
- int m[3][3] = {10, 11, 12, 20, 21, 22, 30, 31, 32};
- CONFUSO! Mas compila...
- int m[3][3] = {{10, 11, 12}, {20, 21, 22}, {30, 31, 32}};

Matrizes

- As matrizes seguem a mesma lógica que os vetores, para inicializar. No entanto, é muito recomendado (mas não obrigatório) que se utilize {} adicionais para separar as dimensões
- int m[3][3] = {10, 11, 12, 20, 21, 22, 30, 31, 32};
- CONFUSO! Mas compila...
- int m[3][3] = {{10, 11, 12}, {20, 21, 22}, {30, 31, 32}};

Matrizes

- Também é possível utilizar a inicialização sem tamanho em vetores e matrizes
- int vet[] = {0, 1, 2, 3};
- int mat[][2] = {{0, 1}, {1, 2}, {1,3}};
- Mas para matrizes, é obrigatório informar a última dimensão.
- Qual o resultado de printf("%d\n", mat[1][1]); ?

Exercício

- Faça um programa que possui uma matriz 3 x 3 inicializada com colchete e imprima o maior número da matriz

Exercício

- Faça um programa que leia uma matriz 3 x 3 do teclado, some a matriz identidade (células que estão anti-diagonal = 1) e imprima na tela

- $$\begin{matrix}[4,\ 3,\ 9] \\ [9,\ 3,\ 8] \\ [2,\ 7,\ 3]\end{matrix} + \begin{matrix}[1,\ 0,\ 0] \\ [0,\ 1,\ 0] \\ [0,\ 0,\ 1]\end{matrix} = \begin{matrix}[5,\ 3,\ 9] \\ [9,\ 4,\ 8] \\ [2,\ 7,\ 4]\end{matrix}$$

Exercício

- Desafio: imagine que, por algum motivo, você não possa declarar matrizes, apenas vetores.
- Como seria possível utilizar duas variáveis para acessar diferentes colunas?