parte6 _ VETORES

Marcos Silvano / DACOM

BCC32A-Algoritmos 1

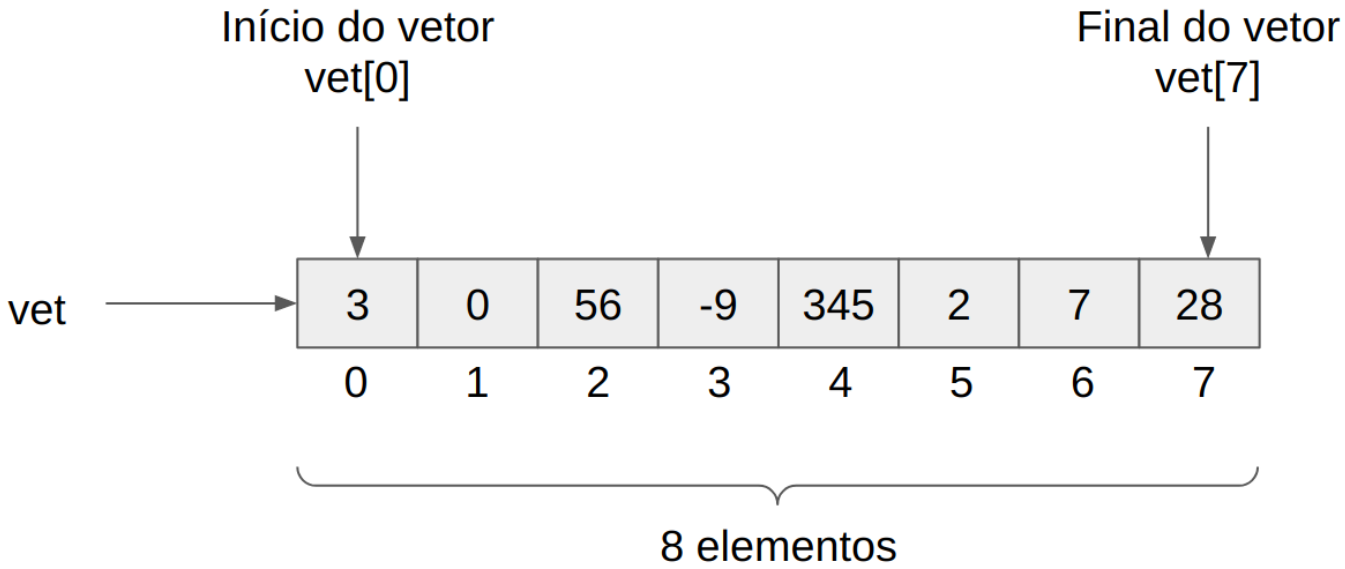
Roteiro

- Visão geral
- Declaração de vetores
- Acesso a vetores
- Vetores multidimensionais
- Passando vetores para funções

Vetores: visão geral

- Um **vetor/array** é uma estrutura que armazena dados de forma sequencial
  - Todos os elementos são do mesmo tipo
  - Vetor possui tamanho fixo
  - O primeiro elemento está na posição 0 (zero)
  - O último elemento está na posição tamanho-1

```
int vet[ ] = {3, 0, 56, -9, 345, 2, 7, 28};
```



Vetores: declaração

- Na declaração do vetor informamos:
  - tipo
  - nome da variável que apontará para o vetor
  - tamanho e/ou elementos

```c
#include <stdio.h>
void main() {
    // DECLARAÇÃO 1: por tamanho
    // SINTAXE: tipo nome_variavel[tamanho]

    int vet1[10]; // <- armazena 10 inteiros em sequência
    float vet2[5]; // <- armazena 5 floats em sequência

    // DECLARAÇÃO 2: por inicialização
    // SINTAXE: tipo nome_variavel[] = {elementos}

    int vet3[] = {1,2,3,4,5,6,7,8}; // <- armazena 8 inteiros em sequência

    // tamanho é opcional quando há inicialização
    int vet4[5] = {-10,21,3,-47,50}; // <- armazena 5 inteiros em sequência
}
```

Vetores: acesso

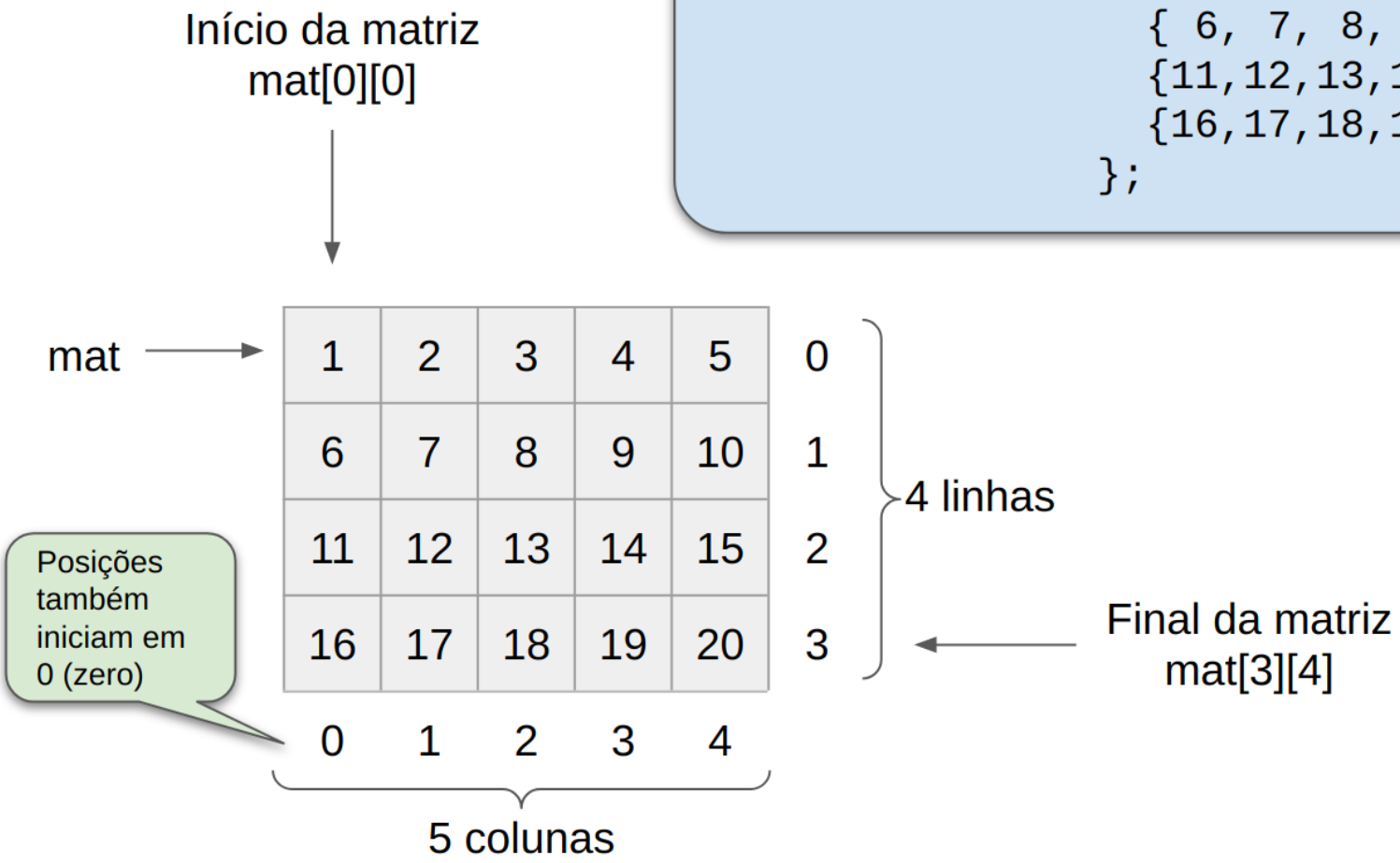
- O tamano de um vetor não é armazenado automaticamente
  - Devemos **SEMPRE** guardar o tamanho do vetor (**é por nossa conta!**)
- Os elementos do vetor são acessados por posição/índice:
  - **vet[pos]** : acessando vetor na posição 'pos'

```c
#include <stdio.h>
void main() {
    printf("VETORES E ELEMENTOS\n");

    int n = 6;
    int vet[] = {2,4,6,8,10,12}; // <- 6 elementos

    printf("primeiro elemento: %d\n", vet[0]);
    printf("ultimo elemento: %d\n", vet[n-1]);

    printf("Percorrendo vetor: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", vet[i]);
    }
    printf("\n");
}
```

Vetores: multidimensões

- Podemos definir vetores de múltiplas dimensões
- Caso mais comum: vetor bidimensional -> matriz

```
int mat[4][5] = { { 1, 2, 3, 4, 5},
                  { 6, 7, 8, 9,10},
                  {11,12,13,14,15},
                  {16,17,18,19,20}
                };
```



Vetores: multidimensões

- Para inicialização da matriz
  - A primeira dimensão sempre deve ter tamanho explícito
  - As demais podem ser omitidas

```
#include <stdio.h>
void main() {
    printf("MATRIZES E ELEMENTOS\n");

    int mat[4][5] = { { 1, 2, 3, 4, 5}, // <- 4 x 5 elementos
                      { 6, 7, 8, 9,10},
                      {11,12,13,14,15},
                      {16,17,18,19,20}
                    };
    int lin = 4;
    int col = 5;

    printf("primeiro elemento: %d\n", mat[0][0]);
    printf("ultimo elemento: %d\n", mat[lin-1][col-1]);

    printf("Percorrendo matriz:\n");
    // laço para percorrer as linhas
    for (int i = 0; i < lin; i++) {
        // laço para percorrer cada elemento de uma linha (colunas)
        for (int j = 0; j < col; j++) {
```

Vetores: passando vetores para funções