INSTITUTO FEDERAL
São Paulo
Câmpus São Carlos

Algoritmos e Programação II

Variáveis compostas homogêneas: vetores e matrizes

Profa. Dra. Eloize Seno

Variáveis compostas homogêneas

- Variável composta é formada por um conjunto de variáveis: **Vetor** e **Matriz**
  - Todas com o mesmo nome;
  - Todas do mesmo tipo de dados;
  - Alocadas em sequência na memória;
  - Por ter o mesmo nome, o que faz a distinção são os índices, que permitem acesso ao conteúdo de sua estrutura.

Vetor

- Declaração:

  tipo identificador[tamanho];

  onde:

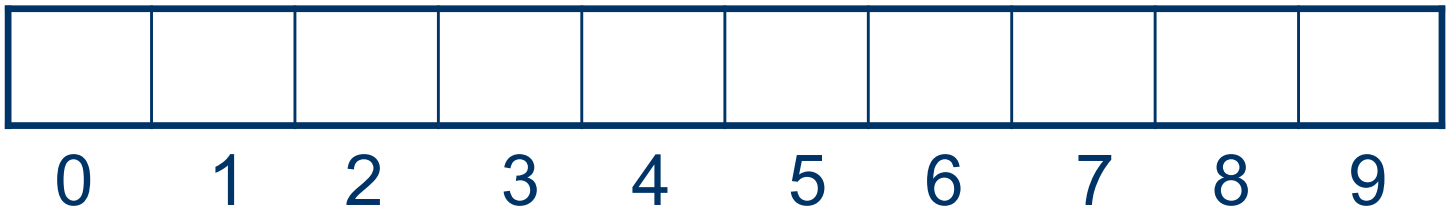
  – **tipo:** é o tipo básico dos dados que serão armazenados no vetor (por ex., int, char, float, etc.);

  – **identificador:** nome da variável do tipo vetor;

  – **tamanho:** é a quantidade de variáveis que compõem o vetor.

Vetor (cont.)

- Exemplo: `int vet[10];`



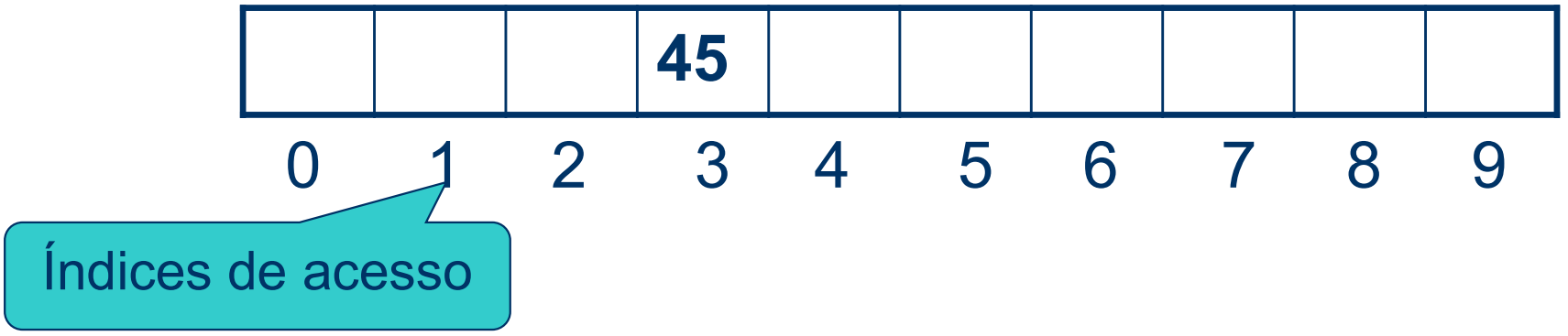
Acessando os elemento de um vetor

- O acesso aos elementos de um vetor é feito por meio de índices;

  Em C, os índices iniciam sempre em **zero** e vão até o tamanho do vetor menos 1 posição.

- **Atribuição**: atribuindo o valor 45 a 4ª posição de vet:

```
vet[3] = 45;
```



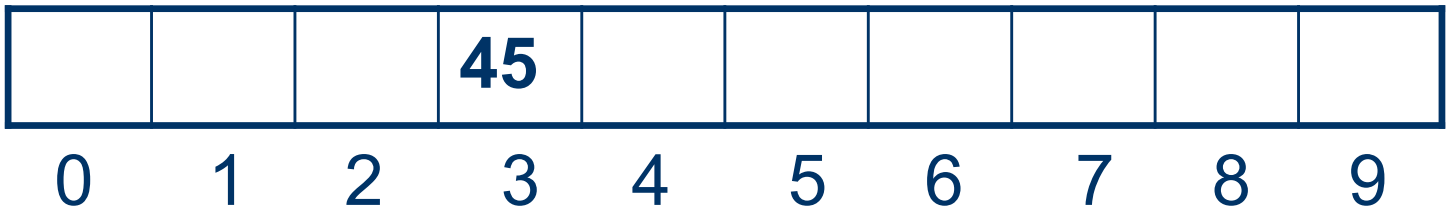
Acessando os elemento de um vetor (cont.)

- **Leitura:**

```
scanf("%d", &vet[3]);
```

- **Impressão**:

```
printf("Valor armazenado na terceira
  posição de vet = %d ", vet[3]);
```



Inicializando vetores

- Inicialização:

```
int z[10];
float f[4] = {4.5, 2.9, -9.0, 5.6};
int a[100] = {0};
int b[ ] = {1,2,3,4};
```

Preenchendo um vetor

- Para preencher um vetor inteiro é necessário usar uma estrutura de repetição.

- Ex.:

```
for (i=0; i<10; i++)
{  printf("Digite o valor da posição
   %d",i);
   scanf("%d", &vet[i]);
 }
```

Imprimindo um vetor

- Para imprimir um vetor inteiro é necessário usar uma estrutura de repetição.

- Ex.:

```
for (i=0; i<10; i++)
   printf("Elemento da linha %d  coluna %d", vet[i]);
```

Vetor de caracteres

- Uma string em C é um vetor de caracteres.

- Exemplo:
  - `char nome[40];` /*armazena um **único** nome.*/
- Inicialização:

```
char s[ ] = “abc”;
char s[ ] = {'a', 'b', 'c', '\0'};
char s[ ] = “Linguagem Programação I";
```

Vetor de caracteres (cont.)

- **Leitura:**
  - `gets(nome); /* lê toda a string até que a tecla Enter seja digitada (inclui espaços em branco). */`

- **Impressão:**
  - `puts(nome); /* imprime toda a string até encontrar o o caracter null ('\0')*/`

Obs: para os comandos **gets** e **puts**, usar a biblioteca **stdio.h**

Matriz

- Variável composta **multidimensional** formada por um conjunto de variáveis;
  - Vetor multidimensional (até 12 dimensões segundo o padrão ANSI)
- Armazena um conjunto de variáveis de um mesmo tipo e identificadas pelo mesmo nome;
- Alocação estática e sequencial na memória;
  - É necessário definir o número de dimensões e o tamanho de cada dimensão a priori;

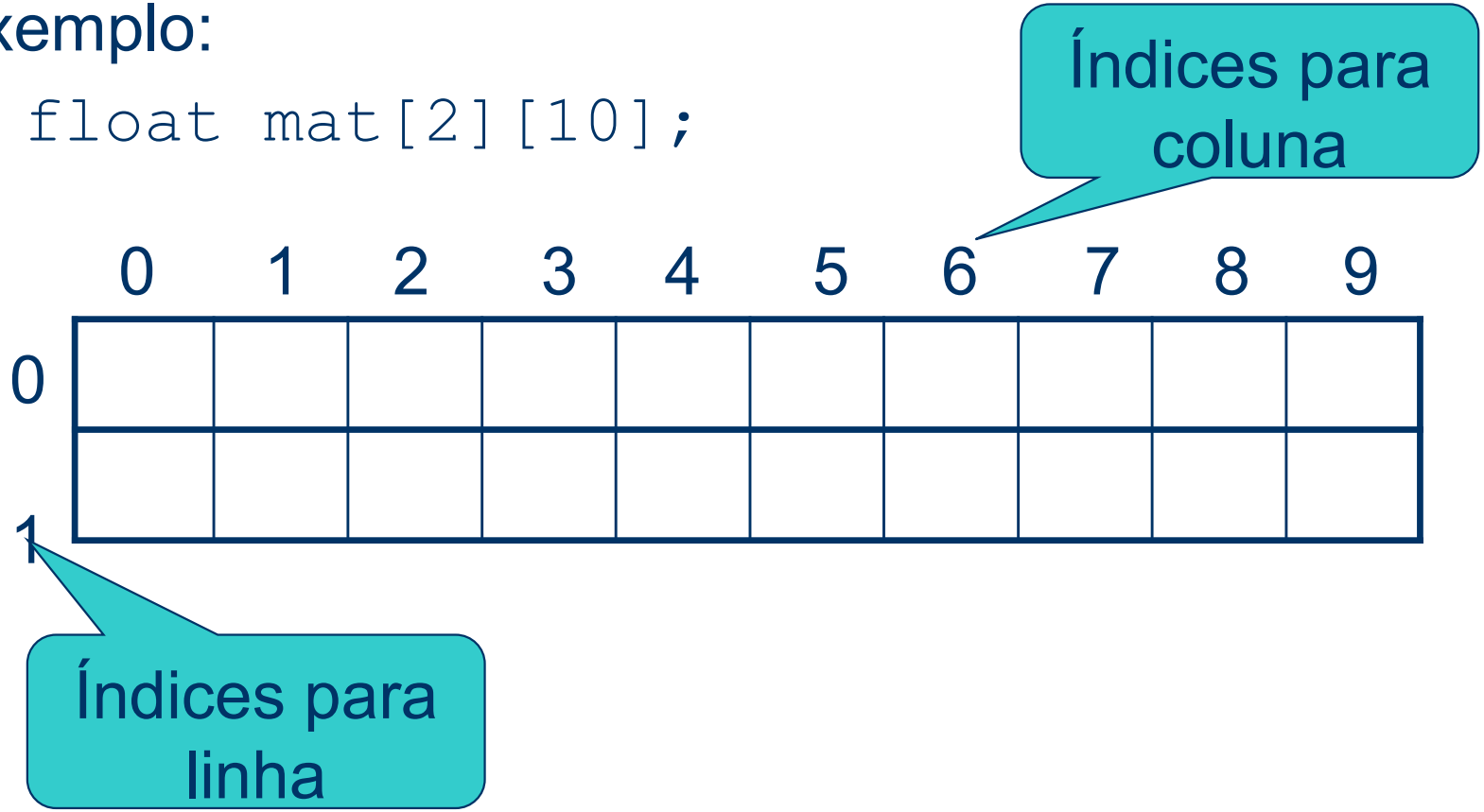
Matriz Bidimensional

- Declaração
  - tipo identificador[dimensão1] [dimensão2];

- Onde:
  - **tipo**: é o tipo básico dos dados que serão armazenados na matriz;
  - **identificador**: nome da variável do tipo matriz;
  - **[dimensão1]**: número de linhas da matriz;
  - **[dimensão2]**: número de colunas da matriz;

Matriz Bidimensional

- Exemplo:
  - `float mat[2][10];`



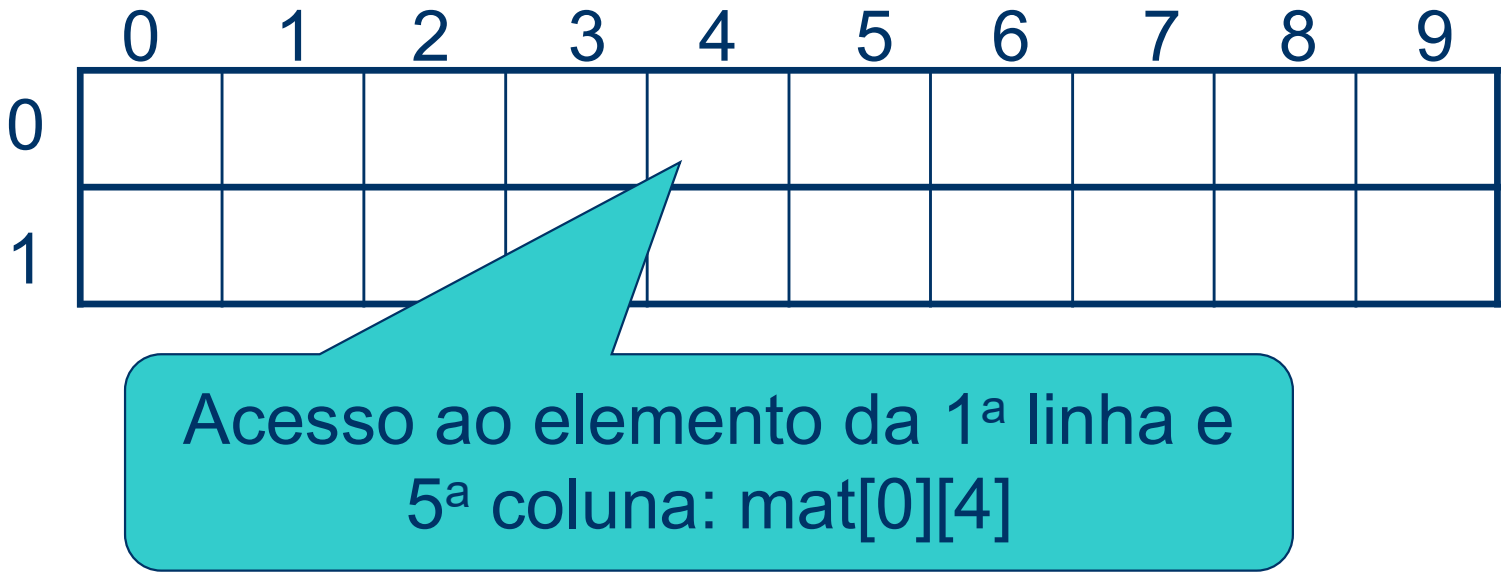
Matriz Bidimensional

- Acesso aos valores da matriz: são necessários dois índices (um para linha e um para coluna);

- Assim como ocorre nos vetores, os índices de uma matriz iniciam sempre em zero;

Matriz Bidimensional

- Exemplo:
  - `float mat[2][10];`



Inicializando matrizes

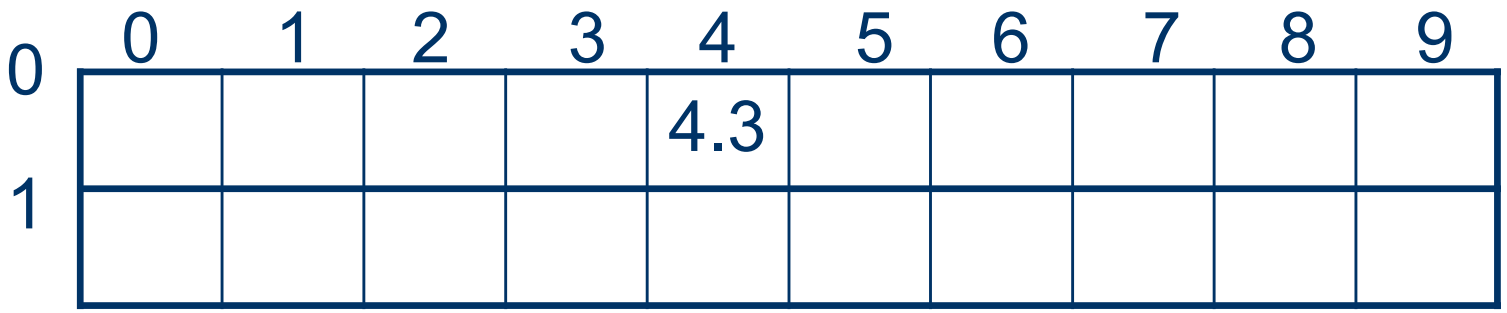
- Ex:

```
int mat1[2][3] = {1,2,3,4,5,6};

int mat2[2][3] = {{1,2,3},{4,5,6}};
```

Acesso aos elementos de uma Matriz

- **Atribuição**: atribuindo o valor 4.3 a 1ª linha e 5ª coluna da matriz:
  - `mat[0][4] = 4.3;`



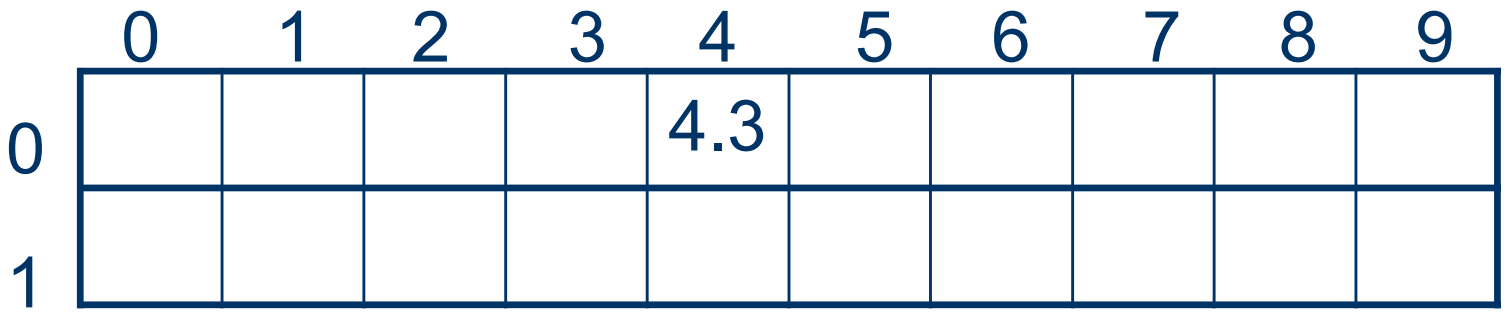
Acesso aos elementos de uma Matriz (cont.)

- **Leitura**: Mostrando o valor armazenado na 1ª linha e 5ª coluna da matriz :

```
scanf("%f", &mat[0][4]);
```

- **Impressão**: Mostrando o valor armazenado na 1ª linha e 5ª coluna da matriz :

```
printf("%2.2f", mat[0][4]);
```



Preenchendo a matriz toda

- Para preencher uma matriz são necessárias **duas** estruturas de repetição.
- Ex.:

```
for (i=0; i<2; i++)
{ for (j=0; j<10; j++)
   { printf("Digite o valor da linha %d
     coluna %d", i, j);
     scanf("%f", &mat[i][j]);
   }
}
```

Imprimindo Todos os Elementos de uma Matriz:

- Para imprimir todos os elementos de uma matriz são necessárias **duas** estruturas de repetição.

```
for (i=0; i<2; i++)
{   for (j=0; j<10; j++)
        printf("%.2f", mat[i][j]);
}
```

Exercícios de Fixação

1- Faça um programa que preencha vetor com oito números inteiros, calcule e mostre dois vetores resultantes. O primeiro vetor deve conter os números positivos e o segundo deve conter os números negativos. **Atenção**: cada vetor terá no máximo oito posições, que podem **não** ser completamente usadas. Não deve haver posições vazias entre dois valores. Por exemplo:

vetor inicial

8	-2	3	1	-5	7	-14	10

positivos	8	3	1	7	10			

negativos	-2	-5	-14					

Exercícios de Fixação (cont.)

2- Faça um programa que preencha dois vetores A e B com dez elementos numéricos cada um calcule e apresente um vetor C resultante da intercalação deles. Por exemplo:

Vetor A

8	3	1	7	10	12	77	100	33	54

Vetor B

-2	-5	-14	5	6	12	17	20	66	16

Vetor C

8	-2	3	-5	1	-14	7	5	10	6	12	12	77	17	100	20	33	66	54	16

Exercícios de Fixação (cont.)

3- Crie um programa que preencha uma matriz 3x3 de números inteiros e verifique se a matriz é simétrica. A matriz será simétrica se e somente se todo elemento A[i,j] = A[j,i]. Segue um exemplo de matriz simétrica:

$$P = \begin{bmatrix} 3 & 2 & 4 \\ 2 & 5 & 8 \\ 4 & 8 & 6 \end{bmatrix}$$

Exercícios de Fixação (cont.)

4- Faça um programa que crie uma matriz A de tamanho n x n de valores inteiros informados pelo usuário. O programa deverá verificar se A é uma matriz identidade e imprimir uma mensagem informando o usuário. Considere que a matriz identidade possui todos os elementos da diagonal principal iguais a 1 e os demais elementos iguais a 0, como no exemplo:

$$I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Exercícios de Fixação (cont.)

5- Faça um programa que crie uma matriz A de dimensão m x n e outra matriz B de dimensão n x p. O programa deverá calcular e apresentar a multiplicação de A por B, como segue no exemplo:

$$A = \begin{bmatrix} 2 & 3 \\ 0 & 1 \\ -1 & 4 \end{bmatrix} \text{ e } B = \begin{bmatrix} 1 & 2 & 3 \\ -2 & 0 & 4 \end{bmatrix}$$

$$A.B = \begin{bmatrix} 2 & 3 \\ 0 & 1 \\ -1 & 4 \end{bmatrix} . \begin{bmatrix} 1 & 2 & 3 \\ -2 & 0 & 4 \end{bmatrix} = \begin{bmatrix} 2.1+3(-2) & 2.2+3.0 & 2.3+3.4 \\ 0.1+1(-2) & 0.2+1.0 & 0.3+1.4 \\ -1.1+4(-2) & -1.2+4.0 & -1.3+4.4 \end{bmatrix} = \begin{bmatrix} -4 & 4 & 18 \\ -2 & 0 & 4 \\ -9 & -2 & 13 \end{bmatrix}$$