

Algoritmos e Técnicas de Programação



Aula? - Vetores

Prof. Guilherme Guerino



Introdução

 Com as estruturas básicas de controle de fluxo já vistas, sequência, seleção e iteração, é possível resolver praticamente qualquer problema de forma algorítmica;

 Contudo, os recursos de armazenamento até agora utilizados não são adequados para soluções que envolvam grandes volumes de dados.





- Se um professor tem 30 alunos em sua turma e deseja calcular a média aritmética e ainda listar as médias dos alunos que forem iguais ou superiores à média da turma, quantas variáveis ele precisará para armazenar os 30 valores?
 - A resposta é: precisará de 30 variáveis.

Dessa forma, o uso de *variáveis simples* para armazenar grandes volumes de dados claramente não parece ser a melhor opção de armazenamento.



Introdução

 Esta aula discute um tipo de variável estruturada denominado arranjo;

 Que agrupa dados do mesmo tipo e possibilita trabalhar com grandes volumes de dados de forma eficiente;

Arranjos de uma dimensão (ou seja, unidimensionais)





- Um *arranjo* é uma variável estruturada, formada pelo agrupamento de variáveis de mesmo tipo (inteiro, real, etc.);
- As variáveis que integram um arranjo compartilham um único nome;
- São identificadas individualmente por um *índice*, que é o valor que indica sua posição no agrupamento;
- Os í*ndices dos arranjos são de tipos ordinais*, como inteiro e podem ser tanto variáveis como constantes.





- A qualquer momento pode-se acessar um determinado *elemento* de um arranjo, sem a necessidade de acessar antes os elementos que o precedem;
- O *índice* utilizado define o elemento sendo acessado:
- Um *elemento* de um arranjo é equivalente a uma *variável simples*;
- Sobre qualquer elemento de um arranjo podem ser efetuadas *todas* as operações até aqui estudadas para variáveis simples.



Arranjos

- Podem ser *unidimensionais* (de uma única dimensão, quando precisam de um só índice para identificar seus elementos) ou;
- Multidimensionais (de duas ou mais dimensões);
- Os arranjos de uma só dimensão também são chamados de VETORES:

• E os de duas ou mais dimensões, de MATRIZES.





 Vetor é conhecido como variável composta homogênea unidimensional;

 Que trata de um conjunto de variáveis de mesmo tipo, que possuem o mesmo identificador (nome) e são alocadas sequencialmente na memória;

 Como as variáveis têm o mesmo nome, o que as distingue é um índice que referencia sua localização dentro da estrutura.





Declaração de vetor em C

Sintaxe:

Tipo NomeDoVetor[quantidade_de_itens];

Onde:

- Tipo é o tipo de variável que o vetor vai armazenar (int, float, char, etc...)
- NomeDoVetor é o nome da variável do tipo vetor
- quantidade_de_itens é um número inteiro que demonstra o tamanho do vetor.



- float V[10];
 - Declaração do vetor do tipo float com 10 números:

- int numeros[5];
 - Declaração do vetor do tipo int com 5 números:





 É importante notar que em linguagem C, o vetor é indexado a partir da posição zero.

- Podemos dizer que em C:
 - A primeira posição de um vetor tem índice zero.
 - A última posição de um vetor tem índice = número de posições – 1.





float V[10];

٧	1.5	9.8	174.7	10.0	- 8.5	49.6	35.0	41.2	1.0	8.9	
	0	1	2	3	4	5	6	7	8	9	

int numeros[5];

numeros	2	7	10	-1	89
	0	1	2	3	4





- Podemos declarar e inicializar um vetor com um tamanho constante, como abaixo:
 - int numeros[5] = {10, 20, 30, 40, 50};

numeros	10	20	30	40	50
	0	1	2	3	4





- Iniciando apenas alguns elementos do vetor:
 - int valores[5] = {2,4,6};
- será equivalente a
 - int valores[5] = {2,4,6,0,0};
- Isto ocorre porque apenas alguns itens do vetor foram inicializados.
- Neste caso, quando o número de itens inicializados é menor que o número total de itens do vetor, os itens não inicializados são automaticamente zerados.





- Inicializando um vetor sem especificar a quantidade de elementos
 - o int valores[] = {3,5,7};

 Neste exemplo, não foi especificado o tamanho do vetor, porém ao inicializar os elementos o compilador faz a contagem dos itens e determina o tamanho do vetor automaticamente.





- É possível criar um vetor e manipular um elemento dele como se fosse uma variável simples.
- Exemplo:
 - int valores[3] = {80, 90, 10};
 - valores[0] = 70:

No exemplo acima, o vetor *valores* na *posição O* foi inicializado com o valor 80. Na próxima linha, a *posição 0* do vetor *valores* recebeu o valor 70. Portanto, o vetor valores na posição O deixou de ser 80 e passou a ser 70.





Operações em vetores

 Diferentes operações podem ser realizadas em um vetor.

- Entre as principais operações estão
 - Atribuição de valores
 - Apresentação de valores





Operações em vetores

 Diferentes operações podem ser realizadas em um vetor.

- Entre as principais operações estão
 - Atribuição de valores
 - Apresentação de valores





Atribuições de valores

A atribuição de valores significa preencher uma ou mais posições do vetor.

• Para isso, deve-se controlar a variável de índice.

 Para tal controle, estruturas de repetição devem ser utilizadas.





Preenchendo um vetor

Atribuição por Leitura

```
for (i = 0; i < 10; i++) {
    scanf ("%d", &vetor[i]);
}</pre>
```





Preenchendo um vetor

Atribuição por inicialização

```
for (i = 0; i < 10; i++) {
  vetor[i] = i;
}</pre>
```





```
for (i = 0; i < 5; i++) {
    scanf("%d", &valor[i] );
}

valor</pre>
```

Iteração 1

Posição = 0 1 2 3 4





```
for (i = 0; i < 5; i++) {
    scanf("%d", &valor[i] );
}</pre>
```

valor

Iteração 2

i = 1 valor[i] = 45

Posição = 0 1 2 3 4





```
for (i = 0; i < 5; i++) {
    scanf("%d", &valor[i] );
}</pre>
```

valor

10 | 45 | 17 |

Posição = 0 1 2 3 4

Iteração 3

i = 2 valor[i] = 17





```
for (i = 0; i < 5; i++) {
    scanf("%d", &valor[i] );
}</pre>
```

valor

Iteração 4

```
10 | 45 | 17 | 3
```

Posição =	0	1	2	3	4
-----------	---	---	---	---	---



SS .

Exemplo

```
for (i = 0; i < 5; i++) {
    scanf("%d", &valor[i] );
}</pre>
```

Iteração 5

| valor[i] = 45

valor

10	45	17	3	45

Posição = 0 1 2 3 4



S.

Exemplo

```
for (i = 0; i < 5; i++) {
    scanf("%d", &valor[i] );
}</pre>
```

Iteração 6

$$i = 5$$

Fora do laço





Operações em vetores

 Diferentes operações podem ser realizadas em um vetor.

- Entre as principais operações estão
 - Atribuição de valores
 - Apresentação de valores





```
for (i = 0; i < 5; i++) {
                                     Iteração 1
     printf("%d", valor[i]);
                         valor
10
                           10
                                45
                                       17
                                                   45
```

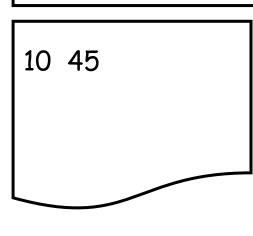
Posição =

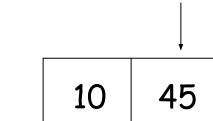
3 4 29





```
for (i = 0; i < 5; i++) {
    printf("%d", valor[i]);
}</pre>
valor
```





17	

45

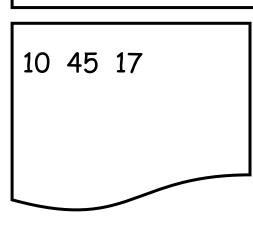
- Posição =
 - 0
- 1 2
- 3
- 4 30





```
for (i = 0; i < 5; i++) {
    printf("%d", valor[i]);
}</pre>
valor
10 45 17
```

Posição =



			ţ
10) 4	15	17

3 45

0 1 2 3



45

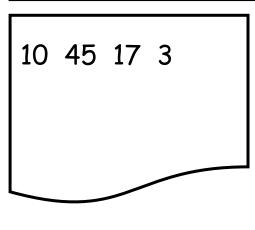


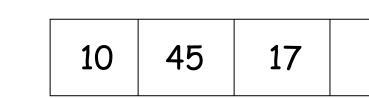
Apresentando vetores

```
for (i = 0; i < 5; i++) {
    printf("%d", valor[i]);
}

valor</pre>
```

Posição =



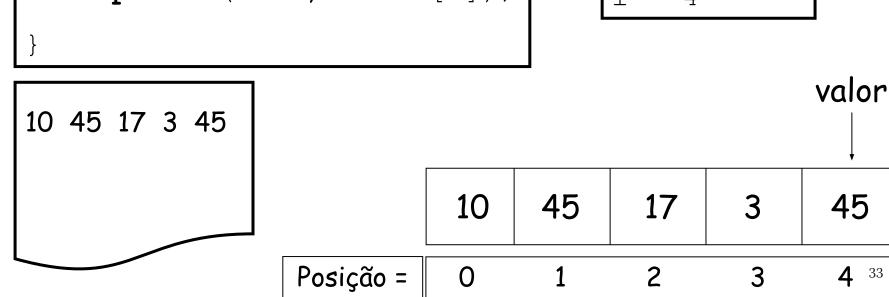


0 1 2 3 4





```
for (i = 0; i < 5; i++) {
                                   Iteração 5
     printf("%d", valor[i]);
```







Considerações Finais

Um vetor apresenta diferentes características.

- Principais:
 - Armazenam dados do mesmo tipo.
 - Apenas um nome é utilizado para referenciar o vetor.
 - O Considera múltiplas posições e/ou índices de memória.

Vetores representam a forma mais simples de estruturar um conjunto de dados.

Dúvidas?

