

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.



Introdução

- **Exemplo**

- 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***)₄



Arranjos

- 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 **índices dos arranjos são de tipos ordinais**, como inteiro e podem ser tanto variáveis como constantes.



Arranjos

- 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

- 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.⁸



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.



Exemplos

- **float V[10];**
 - Declaração do vetor do tipo float com 10 números:
- **int numeros[5];**
 - Declaração do vetor do tipo int com 5 números:



Vetor em C

- 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.



Vetor em C

float V[10];

V	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



Declarando e iniciando vetores

- 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



Declarando e iniciando vetores

- 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.



Declarando e iniciando vetores

- Inicializando um vetor sem especificar a quantidade de elementos
 - `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.



Declarando e iniciando vetores

- 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 0** 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 0** 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]);  
}
```



Preenchendo um vetor

- Atribuição por inicialização

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



Exemplo

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

valor



10				
----	--	--	--	--

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

Iteração 1

i = 0

valor[i] = 10



Exemplo

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

valor



10	45			
----	----	--	--	--

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

Iteração 2

i = 1

valor[i] = 45



Exemplo

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

valor



10	45	17		
----	----	----	--	--

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

Iteração 3

i = 2

valor[i] = 17



Exemplo

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

valor



10	45	17	3	
----	----	----	---	--

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

Iteração 4

i = 3

valor[i] = 3



Exemplo

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

Iteração 5

$i = 4$

$\text{valor}[i] = 45$

valor



10	45	17	3	45
----	----	----	---	----

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



Exemplo

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

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**



Apresentando vetores

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

Iteração 1

i = 0

10

valor



10	45	17	3	45
----	----	----	---	----

Posição =

0

1

2

3

4



Apresentando vetores

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

Iteração 2

i = 1

10 45

valor



10	45	17	3	45
----	----	----	---	----

Posição =

0

1

2

3

4

30



Apresentando vetores

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

10 45 17

Iteração 3

i = 2

valor



10	45	17	3	45
----	----	----	---	----

Posição =

0	1	2	3	4
---	---	---	---	---



Apresentando vetores

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

Iteração 4

i = 3

10 45 17 3

valor



10	45	17	3	45
----	----	----	---	----

Posição =

0	1	2	3	4
---	---	---	---	---



Apresentando vetores

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

Iteração 5

i = 4

10 45 17 3 45

valor



10	45	17	3	45
----	----	----	---	----

Posição =

0	1	2	3	4
---	---	---	---	---



Considerações Finais

- Um vetor apresenta diferentes características.
- Principais:
 - Armazenam dados do mesmo tipo.
 - Apenas um nome é utilizado para referenciar o vetor.
 - Considera múltiplas posições e/ou índices de memória.
- Vetores representam a forma mais simples de estruturar um conjunto de dados.

Dúvidas?

