



Universidade Federal de Uberlândia - Campus Monte Carmelo
Instituto de Geografia
Curso Eng. Agrimensura e Cartográfica



Isadora Ribeiro de Souza

Pesquisa sobre Vetor

Monte Carmelo – MG

09/2018

Vetor

Um vetor é uma sequência de vários valores do mesmo tipo, armazenados sequencialmente na memória, e fazendo uso de um mesmo nome de variável para acessar esses valores.

Os dados armazenados em um vetor são chamados de itens do vetor.

Para localizar a posição de um item em um vetor usamos um número inteiro denominado índice do vetor.

A vantagem de utilizar o vetor é pela facilidade de manipular um grande conjunto de dados do mesmo tipo declarando-se apenas uma variável.

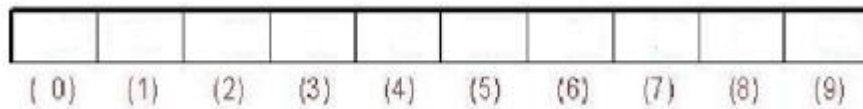
Pseudocódigo: Para declarar um vetor procedemos como indicado a seguir.

NomeDoVetor [inicio..<tamanho>]: TipoDeVariavel

Exemplo:

Vetor[0..9]: Real; (Vetor com 10 posições para guardar números reais)

Representação gráfica de um vetor.



Representação de vetor com 10 elementos.

Vetor em linguagem C

Sintaxe:

Tipo NomeDoVetor[quantidade_de_itens];

Exemplo:

Declaração do vetor do tipo de variável real com 10 números:

float V[10];

É importante notar que em linguagem C, o vetor é indexado (organizado em forma de índice) 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.

Uso de Vetores

- São usados índices para acessar uma casa de um vetor.
- Um índice é um número natural.
- O índice da primeira casa é sempre zero.

- Exemplo 1:

```

1 #include <stdio.h>
2
3 int main () {
4     int v[80], i;
5
6     v[3] = 4; /* casa de índice 3 do vetor v recebe o inteiro 4 */
7     i = 2;
8     v[i] = 3; /* casa de índice 2 do vetor v recebe o inteiro 3 */
9     v[v[i]] = 10; /* vc saberia dizer qual casa do vetor v
10                  * recebe o inteiro 10?
11                  */
12
13     return 0;
14 }

```

Na Linha 4, o vetor `v` com 80 casas é declarado:

[illegible]

Na Linha 6, casa de índice 3 do vetor v recebe o inteiro 4:

[illegible]

Na Linha 8, casa de índice 2 do vetor v recebe o inteiro 3:

[illegible]

Na Linha 9, temos: $i=2$, $v[i]=3$ e $v[v[i]]=v[3]=4$. Desta forma, no comando da Linha 9, a casa de índice 4 do vetor v recebe o inteiro 10:

[illegible]

Declarando e inicializando vetores

Exemplo 1:

➔ Podemos declarar e inicializar um vetor com um tamanho constante, como abaixo:

```
int numeros[5] = { 10, 20, 30, 40, 50};
```

Exemplo 2:

➔ 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.

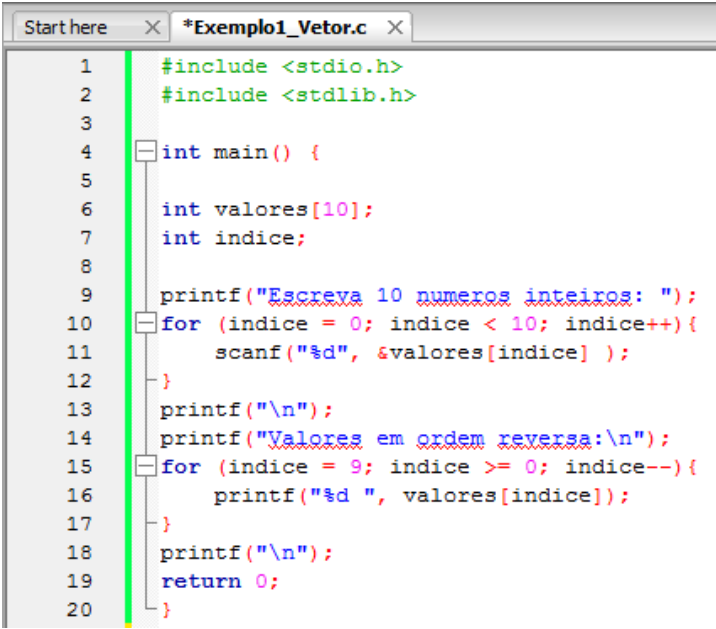
Exemplo 3:

➔ Inicializando um vetor sem especificar a quantidade de elementos:

```
int valores[] = {3,5,7};
```

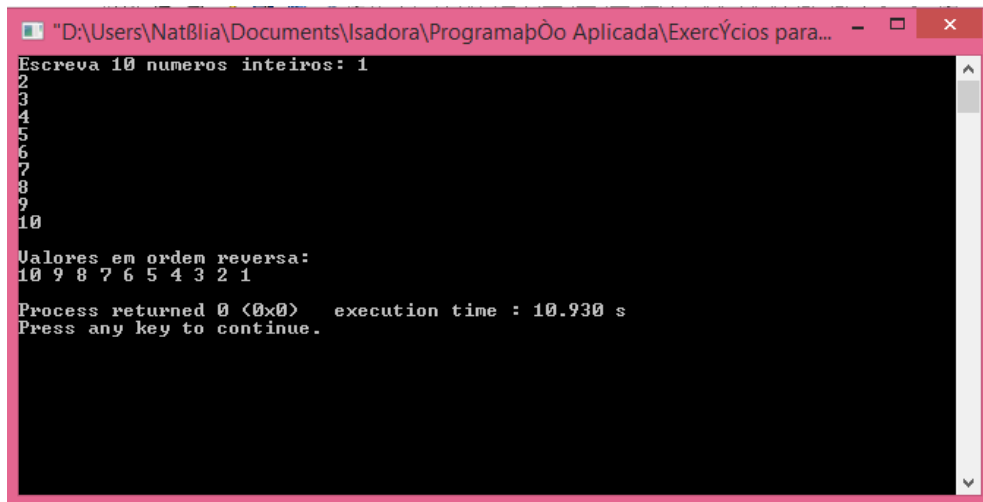
Neste exemplo, não foi especificado o tamanho do vetor, porém ao inicializar os elementos o compilador faz a contagem dos itens e determina o tamanho do vetor automaticamente.

Exercício 1: Um programa que lê dez números e os imprime em ordem inversa.



```
Start here X *Exemplo1_Vetor.c X
1      #include <stdio.h>
2      #include <stdlib.h>
3
4      int main() {
5
6          int valores[10];
7          int indice;
8
9          printf("Escreva 10 numeros inteiros: ");
10         for (indice = 0; indice < 10; indice++){
11             scanf("%d", &valores[indice] );
12         }
13         printf("\n");
14         printf("Valores em ordem reversa:\n");
15         for (indice = 9; indice >= 0; indice--){
16             printf("%d ", valores[indice]);
17         }
18         printf("\n");
19         return 0;
20     }
```

Figura 1: Código fonte (O autor).



```
"D:\Users\Nat8lia\Documents\Isadora\Programa\Oo Aplicada\Exercícios para..."
Escreva 10 numeros inteiros: 1
2
3
4
5
6
7
8
9
10

Valores em ordem reversa:
10 9 8 7 6 5 4 3 2 1

Process returned 0 (0x0)   execution time : 10.930 s
Press any key to continue.
```

Figura 2: Programa executado (O autor).

```
6   int valores[10];
7   int indice;
```

A primeira variável, valores, é um vetor de números inteiros, com 10 elementos. Eles são numerados sequencialmente de 0 até 9. Este vetor armazenará os valores digitados pelo usuário. A segunda declaração cria uma variável contadora para o for de leitura e o for de escrita de valores.

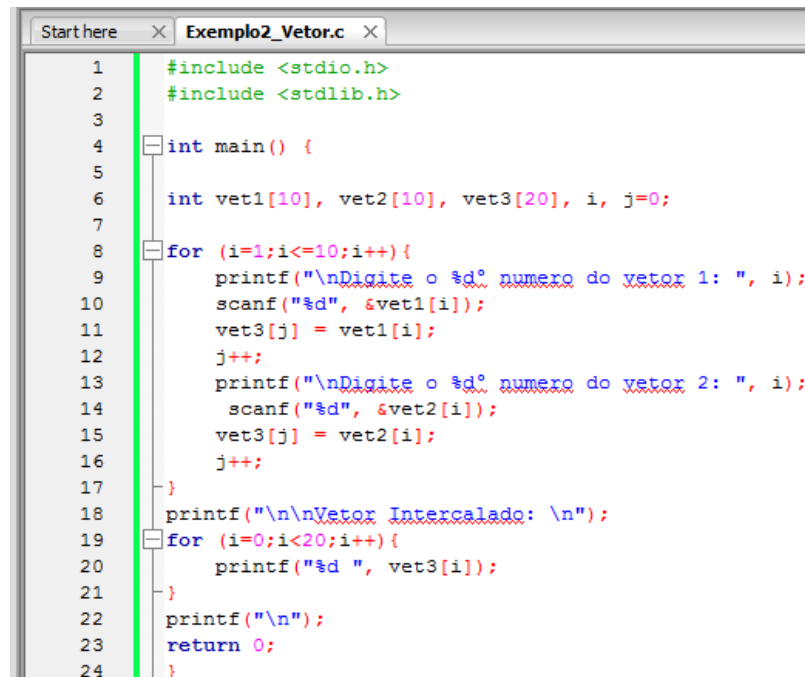
```
9   printf("Escreva 10 numeros inteiros: ");
10  for (indice = 0; indice < 10; indice++){
11      scanf("%d", &valores[indice] );
12  }
```

A estrutura de repetição for executa 10 vezes, variando o valor de indice desde 0 até 9. A cada repetição, o comando scanf lê um número inteiro e o armazena no índice-ésimo elemento do vetor valores. Note que tal como no scanf para variáveis comuns, a referência valores[indice] se comporta como o nome de uma variável comum de tipo int e é necessário precedê-la pelo símbolo &.

```
14  printf("Valores em ordem reversa:\n");
15  for (indice = 9; indice >= 0; indice--){
16      printf("%d ", valores[indice]);
17  }
```

O vetor contém os 10 números lidos no for anterior. Agora, vamos usar novamente a estrutura de repetição for para imprimir o vetor de trás para frente. Por este motivo, fazemos o índice variar de 9 para 0.

Exercício 2: Faça um programa que preencha dois vetores de dez elementos numéricos cada um e mostre o vetor resultante da intercalação deles.

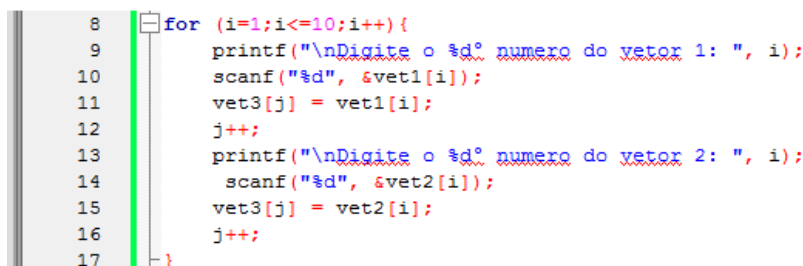


```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5
6      int vet1[10], vet2[10], vet3[20], i, j=0;
7
8      for (i=1;i<=10;i++){
9          printf("\nDigite o %dº numero do vetor 1: ", i);
10         scanf("%d", &vet1[i]);
11         vet3[j] = vet1[i];
12         j++;
13         printf("\nDigite o %dº numero do vetor 2: ", i);
14         scanf("%d", &vet2[i]);
15         vet3[j] = vet2[i];
16         j++;
17     }
18     printf("\n\nVetor Intercalado: \n");
19     for (i=0;i<20;i++){
20         printf("%d ", vet3[i]);
21     }
22     printf("\n");
23     return 0;
24 }
```

Figura 3: Código fonte (O autor).

```
6  int vet1[10], vet2[10], vet3[20], i, j=0;
```

As variáveis `vet1` e `vet2` são vetores de números inteiros, com 10 elementos. Eles são numerados sequencialmente de 1 até 10. Estes vetores armazenarão os valores digitados pelo usuário. A variável `vet3` será a variável auxiliar onde os valores do primeiro e segundo vetor serão “colocados” no `vet3` e assim formar um novo vetor com 20 elementos intercalados. E as demais variáveis serão contadoras para o `for` de leitura e o `for` de escrita de valores.



```
8  for (i=1;i<=10;i++){
9      printf("\nDigite o %dº numero do vetor 1: ", i);
10     scanf("%d", &vet1[i]);
11     vet3[j] = vet1[i];
12     j++;
13     printf("\nDigite o %dº numero do vetor 2: ", i);
14     scanf("%d", &vet2[i]);
15     vet3[j] = vet2[i];
16     j++;
17 }
```

A estrutura de repetição `for` executa 10 vezes o vetor 1 e o vetor 2, variando o valor de índice desde 1 até 10. A cada repetição, o comando `scanf` lê um número inteiro e o armazena no índice-ésimo elemento do vetor 1 e do vetor 2. Em seguida o vetor 3 irá receber os valores do vetor 1 e vetor 2. O `j` nada mais é que o contador para ir acrescentando os valores recebidos pelo vetor 3.

```
18 printf("\n\nVetor Intercalado: \n");
19 for (i=0;i<20;i++){
20     printf("%d ", vet3[i]);
21 }
```

O último for utilizado serve para mostrar o vetor 3 que contém 20 elementos lidos no for anterior que é a união do vetor 1 e do vetor 2, apresentados de forma intercalada.