Vetores

Módulo 7 Aula 0

Linguagem C, o Curso Definitivo WR Kits

Autor: Dr. Eng. Wagner Rambo

Em C podemos utilizar vetores (também chamados de *arrays* ou matrizes unidimensionais) para armazenar um conjunto de dados do mesmo tipo. Em outras palavras, quando existem muitas variáveis que devem armazenar dados semelhantes e tenham o mesmo tipo de dado, podemos utilizar um vetor. Para declarar um vetor, utilize a sintaxe apresentada no Box 1.

```
tipo nome[número de elementos];
```

Box 1 - Sintaxe da declaração de um vetor.

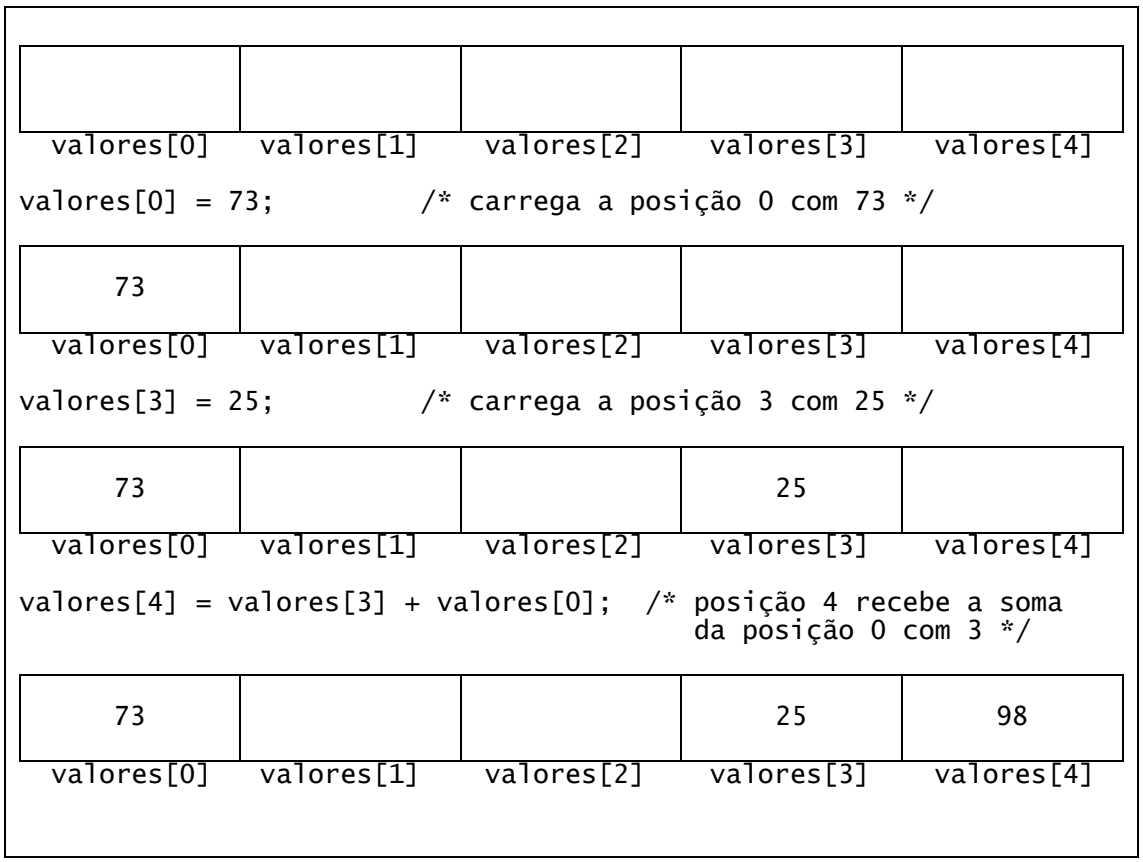
O tipo consiste em qualquer tipo de dado da Linguagem C, tal como utilizamos para a declaração de variáveis. O nome do vetor segue as mesmas regras para os nomes de variáveis e o número de elementos indica quantos elementos o vetor conterá (tamanho do vetor). No Box 2 apresentamos dois exemplos de vetores.

```
int   valores[5];       /* vetor com 5 números inteiros */
float results[40];      /* vetor com 40 números reais */
char  abcd[4];          /* vetor com 4 caracteres */
```

Box 2 - Exemplo de vetor inteiro e vetor real.

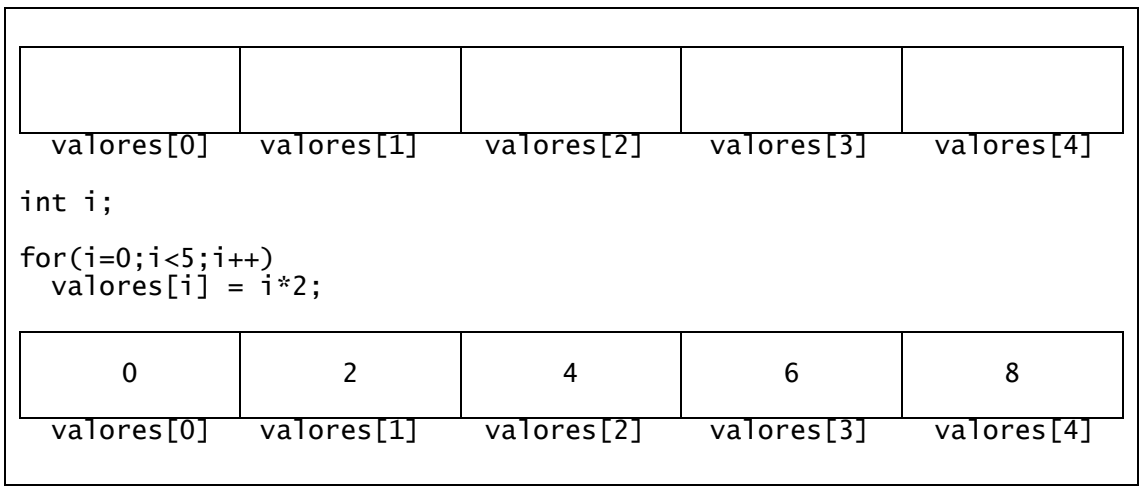
O índice da primeira posição do vetor sempre será 0 e o índice da última posição de um vetor de tamanho n sempre será n-1. Portanto valores[0] é a primeira posição do vetor valores e valores[4] é a última posição. Para o vetor results, a primeira posição é results[0] e a última é results[39]. Para o vetor abcd, a primeira posição é abcd[0] e a última é abdc[3].

Para acessar os elementos de um vetor individualmente, recorremos ao índice, que sempre inicia em zero. O vetor valor tem o aspecto conforme o Box 3, onde também já demonstramos alguns exemplos de atribuição.



Box 3 - Exemplo de atribuição de conteúdo em cada posição de um vetor.

O mesmo vetor poderá ser carregado totalmente a partir de um laço *for* por exemplo, onde você poderá utilizar para diversos tipos de cálculo.



Box 4 - Carga de vetor utilizando laço for.

Um vetor pode ser inicializado posição por posição e também com carga inicial entre chaves e separação por vírgulas, conforme Box 5.

```
int vet[3];                /* declara vetor com 3 inteiros */

vet[0] =  2;               /* inicializa posição 0 */
vet[1] = 75;               /* inicializa posição 1 */
vet[2] = 81;               /* inicializa posição 2 */

/* ========================================================= */

int vet2[3] = {52, 27, 33};  /* declara o vetor e inicializa
                                as posições */
```

Box 5 - Inicialização de vetores.

Assim como nas variáveis, um vetor não inicializado conterá lixo de memória. No entanto, se inicializarmos um vetor parcialmente, os demais elementos serão automaticamente preenchidos com 0, como pode ser visto no Box 6.

```
int vet[6] = { 4, 8, 12 }

/* vet[6] conterá os seguintes valores em cada posição: */
vet[0] =  4;
vet[1] =  8;
vet[2] = 12;
vet[3] =  0;
vet[4] =  0;
vet[5] =  0;
```

Box 6 - Inicialização parcial de vetor.

O programa a seguir calcula a média aritmética simples de 4 números reais, utilizando um vetor.

```
main()
{
  float media[4];
  float med;
  int i;

  for(i=0;i<4;i++)
  {
    printf("Valor %d = ",i);
    scanf("%f",&media[i]);

  } /* end for */

  med = (media[0]+media[1]+media[2]+media[3])/4.0;

  printf("M\x82 \bdia= %.2f\n",med);


} /* end main */
```

Box 7 - Cálculo de média aritmética simples usando vetor.

Constantes

Existem situações onde a ocorrência de um determinado número se repete ao longo do código e poderemos no futuro querer alterar o valor dessa ocorrência. Tomando como exemplo o programa do Box 7, o que aconteceria se precisarmos de um programa que calcule a média de N valores e não de um número fixo? Podemos recorrer ao uso de constantes em C.

Para isso, a estrutura do programa deverá ser alterada um pouco, e veremos logo mais como fazer isso. Primeiro vamos aprender a declarar nossas constantes, que poderão ser utilizadas não apenas com vetores, mas com situações diversas onde tivermos valores imutáveis em nossos códigos. Normalmente declaramos nossas constantes logo após os #includes em nosso projeto. Uma das formas de fazer isso é utilizando a diretiva #define, a mesma que já vínhamos utilizando para a criação de macros. Veja um exemplo no Box 8.

```
#define   pi   3.141592     /* cria uma constante para o π */
```

Box 8 - Declarando uma constante com #define.

Assim, utilizando a palavra "pi" em nossos códigos, o pré-processador irá substituir essa ocorrência pelo valor 3.141592. Suponhamos que em uma atualização futura desejemos aumentar a precisão do pi em mais casas. Ao invés de substituir pi em cada linha de código onde o mesmo é utilizando, simplesmente mudamos isso na linha do #define.

Outra forma de declarar uma constante é por meio da palavra reservada do C *const*, que é um prefixo na declaração de variáveis que impedirá que a mesma tenha o seu conteúdo alterado ao longo do código. Para declarar uma constante com valor 30 por exemplo, utilize a linha do Box 9, lembrando de que a mesma deve ser declara fora do escopo de funções.

```
const int valor = 30;     /* declara uma constante com valor 30 */
```

Box 9 - Declarando uma constante com valor 30.

Portanto, uma constante declarada com *const* faz parte da Linguagem C e existirá fisicamente em determinada posição de memória. Uma constante com a diretiva #define não consiste em Linguagem C e sim uma diretiva de pré-processamento, onde suas ocorrências serão substituídas ao longo do código, de forma semelhante como acontece com as macros. Chamamos essas constantes com #define de constantes simbólicas.

O programa do Box 10 resolve a média aritmética para o número constante que o usuário definir.

```
#define    N    4                /* constante para o número de
                                    Valores da média */

main()
{
  float media[N];                 /* declara vetor de tamanho N */
  float med = 0.0;
  int i;

  for(i=0;i<N;i++)
  {
    printf("Valor %d = ",i);
    scanf("%f",&media[i]);

  } /* end for */

  for(i=0;i<N;i++)                /* faz a somatória de todos os */
    med += media[i];              /* elementos do vetor */

  med/=N;                         /* divide a soma pelo número de
                                    valores, conforme constante */

  printf("M\x82 \bdia= %.2f\n",med);


} /* end main */
```

Box 10 - Programa para fazer a média aritmética de N valores reais.

Exercício proposto: desenvolva um projeto em C onde o usuário define um valor máximo de aporte para a bolsa de valores. Após, ele entra com o valor de 5 ações (que devem ser armazenados em um vetor) e a quantidade para cada ativo. O sistema imprime o valor total em dinheiro e soa um alerta caso o aporte seja menor que o total. Desenvolva o projeto de modo a ser alterado para N ações.

Exemplo: o usuário entre com um aporte de $ 2500,00.

Ativos hipotéticos

```
EMPA3F: $ 12,95 Qtd  30
EMPB4F: $ 19,45 Qtd  14
EMPC3F: $ 22,12 Qtd  50
EMPD3:  $ 11,23 Qtd 100
EMPE4:  $ 32,40 Qtd 200
```

Multiplicando cada valor pela quantidade e somando tudo, o resultado é $ 9369,80. Portanto, o alerta será emitido, pois o aporte é menor.

Bibliografia: DAMAS, Luís; Linguagem C, décima edição.

Disponível em: https://amzn.to/3nGdIbN