

Aula 04 - Introdução à Linguagem C

Programação: Estr. Dados Vetores

OPL e IC1

Prof: Anilton Joaquim da Silva

Anilton.ufu@outlook.com

Introdução

- Imagine o seguinte problema:
Ler um conjunto de N notas do tipo float, calcular sua média e imprimir a quantidade de notas acima da média.
- Com o que sabemos até aqui isto não é possível porque quando lemos um novo valor para nota o valor anterior será substituído.
- Se fossem somente 5 notas uma saída seria definir cinco variáveis diferentes ($N_1, N_2 \dots N_5$).
- De qualquer forma este programa seria limitado a somente 5 notas, é um programa difícil de manter, será necessário um código grande para um problema simples.....
- A solução para esta situação é o uso de vetores (ou variáveis indexadas, ou arranjo, ou array)

Vetores

- Um vetor é uma variável capaz de armazenar vários dados de mesmo tipo, com o mesmo identificador (mesmo nome) e alocadas seqüencialmente na memória. Uma vez que as variáveis têm o mesmo nome, o que as distingue é um índice, que referencia sua localização (posição) dentro da estrutura.

- Declaração de um Vetor

Tipo_de_dado nome_vetor [tamanho_vetor];

- Ex: `int vetor1 [4];`
`float notas[100];`
`char nome [30];`

Vetores

- Seja o exemplo de números inteiros:

```
int vetor1 [4];
```

vetor1	1	5	9	4
Índice	0	1	2	3

	Memória
1000	1
1001	
1002	5
1003	
1004	9
1005	
1006	4
1007	
1008	

Obs.: o índice do vetor, em C, sempre começa em 0. Logo o vetor1 pode armazenar 4 inteiros, com controle de índice começando em 0 até 3.

Vetores

- Seja o exemplo de números float:
`float vetor2 [3];`

vetor2	3.14	1.8	9.5
Índice	0	1	2

Obs.: O vetor2 armazena 3 floats, com controle de índice começando em 0 até 2.

	Memória
1000	
1001	
1002	3.14
1003	
1004	
1005	
1006	1.8
1007	
1008	
1009	9.5
1010	
1011	
1012	
1013	

Vetores

- Seja o exemplo de números inteiros:

```
int vetor1 [4];
```

vetor1	1	5	9	4	?
Índice	0	1	2	3	

- Posição (índice):
 - `vetor1[2]` → 9
 - `vetor1[1]` → 5
 - `vetor1[4]` → ? ---> ERRO

Atribuindo valores a um vetor

- Para atribuir diretamente um valor a uma determinada posição do vetor, pode-se usar a seguinte expressão:

nome_vetor[posição] = valor;

Ex.: int vetor1 [6];

vetor1[5]=2;

vetor1[1]=1;

vetor1		1				2	
Índice	0	1	2	3	4	5	

Lendo um vetor

- Para ler um vetor e ter seus valores alocados em posições específicas, pode-se utilizar o comando **for**.

```
for (int i = 0; i < N; i++)  
{  
    printf("Digite o elemento %d do vetor" , i );  
    scanf("%d", &vetor [i]);  
}
```

OBS: o valor da variável N deve ser informado (lido) antes do for, sendo definido como o tamanho do vetor em uso na execução, porém não pode exceder o valor do tamanho do vetor declarado. Isto é, por exemplo se declarar `int vetor[10]` ou `float vetor[10]`, então N não pode ser maior que 10, pois a minha reserva é para no máximo 10 valores.

Imprimindo um vetor

- Para mostrar um vetor já carregado e ter seus valores mostrados na tela, pode-se utilizar o comando **for**.

```
printf ("Vetor digitado: ");  
for (int i = 0; i < N; i++)  
{  
    printf (" %d" , vetor [i]);  
}  
printf("\n"); //posicionar na próxima linha
```

Exemplo completo

Programa que lê um vetor de n ($n < 100$) posições e imprime estes valores na tela:

```
Int main()
{
    int n;
    int vetor [100];
    printf ("Digite tamanho, efetivo, do vetor (n<100): ");
    scanf ("%d", &n);
    for (int i = 0; i < n; i++)
    {
        printf("Digite o elemento %d do vetor: " , i );
        scanf ("%d", &vetor [i]);
    }
    printf ("Vetor digitado: ");
    for (int i = 0; i < n; i++)
    {
        printf (" %d" , vetor [i]);
    }
    printf("\n"); //posicionar na próxima linha
    return 0;
}
```