
# Prática Profissional II – Linguagem de Programação Estruturada

Curso: Análise e Desenvolvimento de Sistemas
Modalidade: Presencial
Professor Esp. Wesley Tschiedel
Email: wesley.tschiedel@ucb.br

# Unidade 2
# Estrutura de dados fundamentais: Declarações e tipos.
# Unidade 3
# Vetores e Matrizes.

# Declarando vetores e matrizes

Um vetor é uma sequência de valores do mesmo tipo.

Que podem ser acessados a partir da sua posição.

Declaração:

**tipo nomeDaVariavel**[quantidade de elementos];

# Indexador

Elemento responsável por identificar a posição do vetor.

**Lembre-se: o endereço inicial do vetor é sempre índice 0 (zero).**

**Exemplo:** declaração de um vetor de inteiros, com 10 elementos:

```
...
int v[10];
...
```

Os elementos de um vetor podem ser acessados a partir da sua posição, ou índice, e usados no programa como qualquer outra variável.

Os índices de um vetor de **N** elementos variam de **0** a **N-1** (**0** corresponde ao primeiro elemento e **N-1** corresponde ao último).

```
...
v[0] = 1;
v[1] = 2;
v[2] = v[0] + v[1];
...
```

O índice para acesso ao elemento de um vetor pode ser uma variável ou uma expressão inteira.

**Exemplos:**

```
...
for(i=0; i < N; i++) v[i] = 0;
...
v[n-k+1] = v[n-k+i];
...
```

# EXEMPLO 1

```
main(){
 int t[10];
 int i, max, min;

 for(i=0; i<10; i++)
 scanf("%d", &t[i]);
 max = t[0]; min = max;

    for(i=1; i<10; i++){
        if(t[i] > max)
        max = t[i];

        if(t[i] < min)
        min = t[i];
        }
 printf("maior valor: %d e menor valor: %d \n", max, min);
 system("PAUSE");
}
```

## Valor Inicial de um Vetor

Ao declarar uma variável é possível definir um valor inicial para a mesma. **Exemplo:**

```
int k = 21;
```

Isso é possível também no caso de um vetor e nesse caso, a sequência de valores aparece entre '{' e '}', separados por vírgula. **Exemplo:**

```
int dias[12]={31,28,31,30,31,30,31,31,30,31,30,31};
```

Um vetor é uma série de variáveis do mesmo tipo referenciadas por um único nome, onde cada variável é diferenciada através de um número chamado "**subscrito**" ou "**índice**". Colchetes são usados para conter o subscrito. A declaração abaixo:

**int notas[5];**

Aloca memória para armazenar 5 inteiros e anuncia que notas é um vetor de 5 membros ou "**elementos**".

# EXEMPLO 2

```
main(){
    int notas[5];
    int i, soma;
    for(i=0; i<5; i++) {
        printf("Digite a nota do aluno %d: ", i);
        scanf("%d", &notas[i]);
        }
    soma=0;
    for (i=0; i<5; i++)
       soma=soma+notas[i];
    printf("Media das notas: %d. \n", soma/5);
    system("PAUSE");
}
```

# ARMAZENANDO DADOS NO VETOR

A seção do código que coloca dados no vetor é:

```
for(i=0; i<5; i++) {
    printf("Digite a nota do aluno %d: ", i);
    scanf("%d", &notas[i]);
}
```

# LENDO DADOS DO VETOR

Código que lê os dados já armazenados no vetor:

```
soma=0;
    for (i=0; i<5; i++)
        soma=soma+notas[i];
printf("Media das notas: %d. \n", soma/5);
```

## DIRETIVA #define

Pode ser usada para definir constantes simbólicas com nomes apropriados.

**#define LIM 40**

# MATRIZ

Uma matriz é um tipo de dado usado para representar uma certa quantidade de variáveis de valores homogêneos.

Uma matriz é uma série de variáveis do mesmo tipo referenciadas por um único nome, onde cada variável é diferenciada através de um número chamado “**subscrito**” ou “**índice**”.

# EXEMPLO 03

```
main(){
   int tab[5][10];
   int i,j;
   for(i = 0; i < 5; i++)
          for(j = 0; j < 10; j++)
                    scanf(“%d”, &tab[i][j]);
   for(i = 0; i < 5; i++){
        for(j = 0; j < 10; j++)
        printf(" %d ",tab[i][j]);
        printf("\n");
      }
  system(“pause”);
}
```

# ARMAZENANDO DADOS NA MATRIZ

A seção do código que coloca dados na matriz é:

```
for(i = 0; i < 5; i++)
        for(j = 0; j < 10; j++)
                scanf("%d", &tab[i][j]);
```

# LENDO DADOS DA MATRIZ

Código que lê os dados já armazenados na matriz:

```
for(i = 0; i < 5; i++){
        for(j = 0; j < 10; j++)
                printf(" %d ",tab[i][j]);
                printf("\n");
      }
```

# EXEMPLO 04

```
#define L 5
#define C 10
main()
{
   int tab[L][C];
   int i,j;
   for(i = 0; i < L; i++)
          for(j = 0; j < C; j++)
                 scanf(“%d”, &tab[i][j]);
   for(i = 0; i < L; i++){
        for(j = 0; j < C; j++)
        printf(" %d ",tab[i][j]);
        printf("\n");
       }
  system(“pause”);
}
```

# Atividade Prática

# Referências Bibliográficas

**Básica:**


- EVARISTO, J.,**Aprendendo a programar programando em C**, Book Express, 2001, 205p.
- MIZRAHI, V. V., **Treinamento em Linguagem C**, Módulo 1 e 2, Makron Books do Brasil Editora Ltda, 1990, 273 p.
- SCHILDT, H., **C Completo e Total**, Editora Makron Books doBrasil Editora Ltda, 1997, 827p.

# Referências Bibliográficas

**Complementar:**


- DEITEL, H. M. e Deitel, P. J.,**C++ Como Programar**, 3. ed. Porto Alegre: Artmed Editora S.A, 2001. 1098 p.

- MANZANO, J. A. N. G. **Estudo Dirigido: Linguagem C.** 6. ed. São Paulo: Érica, 2002.

- SOFFNER, Renato. **Algoritmos e programação em linguagem C**. São Paulo Saraiva 2013.

- TENENBAUM, A. M.; LANGSAM, Y.; AUGENSTEIN, M. J. **Estruturas de Dados Usando C.** São Paulo: Makron Books, 1995.

- ZIVIANI, Nivio. **Projeto de algoritmos: com implementações em Pascal e C.** 3. ed., rev. e ampl. São Paulo, SP: Cencage Learning, 2015. xx, 639 p.