



**Profº Luiz Paulo Zanetti**

**E-mail: [luizpaulozanetti@hotmail.com](mailto:luizpaulozanetti@hotmail.com)**



**Curso Superior de Tecnologia em  
Análise e Desenvolvimento de Sistemas**

**Disciplina  
Linguagem de Programação**

# Vetor / Arrays

# Vetor / Arrays

**Os vetores, também conhecidos por arrays.**

**O que é um vetor em C... E para quê serve ?**

**Imagine que você foi contratado para criar um programa em C para uma escola.**

**Nesse programa você tem que armazenar as notas dos alunos, nomes, médias, nome dos pais, faltas e tudo mais.**

**E aí? Vai declarar quantos inteiros pra armazenar as notas? Centenas? Milhares?**

**E quantos caracteres para armazenar esses nomes?**

**E quantos floats para armazenar as notas e médias, de cada matéria, para cada aluno?**

# Vetor / Arrays

Como declarar um vetor em C ?

A sintaxe é a seguinte:

**tipo nome[numero\_de\_elementos];**

Ou seja, a sintaxe é a mesma de declarar uma variável normal, mas não vamos declarar somente uma, vamos declarar várias. E o par de colchetes ao lado do nome da variável serve para isso: especificar quantas daquelas variáveis estamos declarando.

# Vetor / Arrays

**Como declarar um vetor em C**

**Por exemplo, vamos declarar 10 inteiros que vão representar a idade de 10 pessoas:**

**int idade[10];**

**Agora 50 floats que vão representar a nota de 50 alunos:**

**float notas[50];**

# Vetor / Arrays

**A contagem dos índices começa sempre do 0  
Embora tenhamos declarado as variáveis com um nome, elas não podem ter um mesmo nome. Por isso, um número é associado ao seu nome.**

**No caso da `idade[10]`, as variáveis inteira são:  
`idade[0]`, `idade[1]`, `idade[2]`, ..., `idade[9]`**

# Vetor / Arrays

**Isso mesmo, o primeiro elemento é sempre o zero.**

**No caso das `notas[50]`, as variáveis do tipo float são: `notas[0]`, `notas[1]`, `notas[2]`, ... , `notas[48]` e `notas[49]`**

**Então, se uma variável tem ‘n’ elementos, seus índices variam, sempre, de 0 até n-1, totalizando ‘n’ elementos.**



# Vetor / Arrays

**Como usar acessar os elementos de um vetor em C**

**Declaramos várias variáveis com o mesmo nome, mas como se referir, individualmente, a cada uma delas?**

**A resposta é simples: usando números, ou índices.**

**'notas' é um vetor de floats.**

**Se quiser usar um tipo float, use a seguinte sintaxe: nome[índice]**

# Vetor / Arrays

Então, suas variáveis, de forma independente, são chamadas de: **notas[0]**, **notas[1]**, **notas[10]** etc.

Esses serão seus nomes. Você pode usar como usaria as variáveis (na verdade elas são variáveis, como se tivessem sido declaradas manualmente), por exemplo:

Armazenar a nota de um aluno que tirou 10

**nota[10] = 10.0** //esse programa em C

Somar a nota de dois alunos:

**float soma = notas[3] + notas[4];**

Incrementar: **nota[5]++;**

# Vetor / Arrays

Enfim, pode fazer tudo. São variáveis do tipo float normais.

A diferença é que **os nomes das variáveis têm números**, que são **chamados**, em programação, de **índice**, que são **criados automaticamente** quando você declara um bloco de vários **elementos chamados de vetores ou arrays**.

# Vetor / Arrays

Como de costume, para fixar melhor, vamos aos exemplos de código!

**Exemplo 1:** Faça um programa que peça 3 números inteiros ao usuário, armazene em um vetor, depois mostre o valor de cada elemento do vetor, assim como seu índice.

Primeiro declaramos um vetor de inteiros, contendo 3 elementos:

```
int numbers[3];
```

# Vetor / Arrays

Agora vamos pedir pro usuário preencher esses três números.

Lembre-se que você é programador e sabe que os **índices vão de 0 até 2**.

Mas o usuário não. **Pro leigo, é número 1, número 2 e número 3, não inicia no 0.**

No laço for, o nosso **'índice' vai de 0 até 2**.

Porém, ao recebermos o valor de índice **'índice'**, estamos pedindo ao usuário o valor do número **'índice+1'**.

# Vetor / Arrays

Por exemplo, para armazenar um valor no **'number[0]'**, vamos pedir o número **'0+1'** ao usuário.

Para armazenar um valor no **'number[1]'**, vamos pedir o número **'1+1'** ao usuário.

Para armazenar um valor no **'number[2]'**, vamos pedir o número **'2+1'** ao usuário.

Usaremos **outro laço for** para exibir o valor dos números, através dos índices, que **variam de 0 até 2**.

Porém, novamente, temos que **mostrar 1 ao 3 pro cliente, pois pra ele não faz sentido**

**'número 0 -> valor 10 '** e sim **'número 1 -> valor 10'**.

# Vetor / Arrays

```
#include <stdio.h>
#include <conio.h>
int main()
{
    int number[3],
        indice;
    clrscr();
    for(indice=0 ; indice <= 2 ; indice++)
    {
        printf("Entre com o numero %d: ", indice+1);
        scanf("%d", &number[indice]);
    }
    for(indice=0 ; indice <= 2 ; indice++)
        printf("Numero %d = %d\n", indice+1, number[indice]);
    getch();
    return(0);
}
```

```
Entre com o numero 1: 123
Entre com o numero 2: 124
Entre com o numero 3: 125
Numero 1 = 123
Numero 2 = 124
Numero 3 = 125
```

# Vetor / Arrays

**Exemplo 2:** Faça um programa em C que peça ao usuário duas notas que ele tirou e mostre a média. Use vetores! Aliás, use somente um vetor para essas três variáveis.

Vamos declarar um vetor de float de três elementos. Nas duas primeiras posições armazenamos as notas do usuário (**nota[0]** e **nota[1]**), e na terceira posição (**nota[2]**) armazenaremos a média (**nota[0]** + **nota[1]**)/2.



# Vetor / Arrays

```
#include <stdio.h>
#include <conio.h>
int main()
{
    float notas[3];
    clrscr();
    printf("Insira sua primeira nota: ");
    scanf("%f", &notas[0]);
    printf("Insira sua segunda nota: ");
    scanf("%f", &notas[1]);
    notas[2] = (notas[0] + notas[1])/2;
    printf("Sua media e: %.2f\n", notas[2]);
    getch();
    return(0);
}
```

```
Insira sua primeira nota: 8
Insira sua segunda nota: 9
Sua media e: 8.50
```

# Vetor / Arrays

São simplesmente **variáveis normais**, a diferença é que **são várias delas**.

É até **mais fácil** trabalhar com **vetores** do que com **variáveis declaradas manualmente**, por **conta dos índices**.

Inicializando vetores

Assim como nas variáveis, podemos **inicializar os vetores assim que declaramos**.

Como são vários valores, temos que colocar todos esses valores **entre chaves {}**.

# Vetor / Arrays

## Inicializando vetores

Assim como nas variáveis, podemos inicializar os vetores assim que declaramos.

Como são vários valores, temos que colocar todos esses valores entre chaves {}.

Veja, pra inicializar um vetor de 3 inteiros:

```
int numeros[3] = { 1, 2, 3};
```

Ou seja:

```
numeros[0] = 1;
```

```
numeros[1] = 2;
```

```
numeros[2] = 3;
```

# Vetor / Arrays

**E para caracteres? Ora, vale o mesmo.  
Vamos armazenar a frase:**

**FATEC Taubaté !!!**

**Com o caractere de new line \n, temos que declarar 17 caracteres (incluindo o espaço em branco, que também é um char em C).**

# Vetor / Arrays

```
#include <stdio.h>
#include <conio.h>
```

```
void main()
{
```

```
    int indice;
```

```
    char nome[8] = {'Z', 'a', 'n', 'e', 't', 't', 'i', '\n'};
```

```
    clrscr();
```

```
    for(indice=0 ; nome[indice] != '\n' ; indice++)
```

```
        printf(" %c", nome[indice]);
```

```
    getch();
```

```
}
```



Zanetti

# Vetor / Arrays

## *Funções para manipular strings - gets*

*Lê uma string do teclado.*

*Forma geral:*

*gets(nome\_string);*

*Exemplo:*

```
main()
{
    char str[80];
    gets(str);
    printf("%s", str);
}
```

# Vetor / Arrays

## *Funções para manipular strings - puts*

*Mostra uma string na tela.*

*Forma geral:*

```
puts(nome_string);
```


*Exemplo:*

```
main()  
{  
    puts("Esta é uma mensagem");  
}
```

# Vetor / Arrays

## *Funções para manipular strings - strcpy*

*Copia uma string em outra.*

*Forma geral:*   
*strcpy(para, de);*

*Lembre-se que a string para deve ser grande o suficiente para conter de.*

```
main()
{
char str[80];
strcpy(str, "alo");
}
```



# Vetor / Arrays

## *Funções para manipular strings - strcat*

*Adiciona uma string em outra.*

*Forma geral:*

*strcat(s1, s2);*

*s2 será anexada (concatenada) ao final de s1.*

```
main()
```

```
{ char primeiro[20], segundo[10];
```

```
    strcpy(primeiro, "bom");
```

```
    strcpy(segundo, " dia");
```

```
    strcat(primeiro, segundo);
```

```
    printf("%s\n", primeiro);
```

```
}
```

# Vetor / Arrays

## *Funções para manipular strings – strcmp*

*Compara 2 strings.*

*Forma geral:*

*strcmp(s1, s2)*

*Compara a string 1 com a string 2. Se as duas forem idênticas a função retorna zero. Se elas forem diferentes a função retorna não-zero.*

*Exemplo:*

*if (strcmp(str1, str2))*

*printf ("\n\nAs duas strings sao diferentes.");*

*else printf ("\n\nAs duas strings sao iguais.");*

# Vetor / Arrays

## *Funções para manipular strings - strlen*

*Retorna o tamanho da string*

*Forma geral:*

*strlen(str);*

*Exemplo:*

*main()*

*{*

*char str[80];*

*printf("Digite uma string: ");*

*gets(str);*

*printf("Tamanho: %d\n", strlen(str));*

*}*