

# Algoritmos - AULA 2

---

## Sumário

1	O que são funções? .....	1
2	Vantagens .....	1
3	Definindo uma função .....	1
4	Funções sem passagem de parâmetro.....	2
4.1	Exemplo.....	2
5	Funções com passagem de parâmetro .....	2
5.1	Exemplo.....	3
6	Funções com retorno .....	3
6.1	Exemplo.....	4
7	Exercícios.....	4

---

## 1 O que são funções?

Funções são as estruturas que permitem ao usuário separar seus programas em blocos. Se não as tivéssemos, os programas teriam que ser curtos e de pequena complexidade. Um ponto chave na resolução de um problema complexo é conseguir “quebrá-lo” em subproblemas menores.

Já utilizamos diversas funções, como o `sqrt( )`, `printf( )` e `scanf( )`. Entretanto, não existe um repositório com todas as funções que precisaremos em um determinado sistema. Precisamos criar as nossas próprias funções.

## 2 Vantagens

- Organiza melhor o código:
  - Cada função é um algoritmo;
  - Dividir um programa complicado em várias funções simples.
- Evita repetição de código semelhante:
  - Escrever o código apenas uma vez;
  - Usar a mesma função várias vezes para variáveis diferentes.
- Simplifica e agiliza a programação:
  - Permite o reaproveitamento de código já construído (por você ou por outros programadores).

## 3 Definindo uma função

Podemos ter funções com passagem de parâmetro e sem passagem de parâmetro, com retorno ou sem retorno. A forma mais genérica de definição de uma função em C é a seguinte:

```
tipo nome (tipo parâmetro1, tipo parâmetro2,...)
{
```

```

    comandos;
    return valor;
}

```

Onde:

- tipo: determina o tipo do valor de retorno;
- nome: é o nome da função;
- tipo parâmetro: são variáveis, que recebem os valores passados quando chamamos a função.
- return: é o valor a ser retornado depois que executamos a função.

## 4 Funções sem passagem de parâmetro

Para chamar uma função, basta colocar o nome da referida função seguida de parênteses. Se não tem passagem de parâmetros, você deve deixar os parênteses em branco.

Se não tem retorno, você deve utilizar void, que em inglês quer dizer vazio, permitindo construir funções que não retornam.

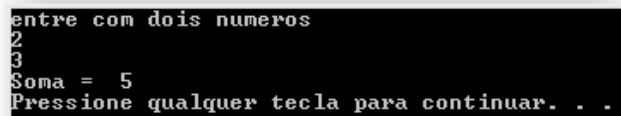
### 4.1 Exemplo

Vamos elaborar um algoritmo que lê dois valores, calcula a soma dos mesmos e mostra o resultado dessa soma.

```

#include <stdlib.h>
#include <stdio.h>
void calcula(){
    int soma,a,b;
    printf("entre com dois numeros\n");
    scanf("%d%d",&a,&b);
    soma = a + b;
    printf("Soma = %d\n",soma);
}
int main ( ) {
    calcula();
    system("pause");
    return 0;
}

```



```

entre com dois numeros
2
3
Soma = 5
Pressione qualquer tecla para continuar. . .

```

## 5 Funções com passagem de parâmetro

Se você for passar parâmetros para uma função, deve definir o tipo e quantidade de parâmetros na definição da função.

E quando for chamar esta função, deverá passar parâmetros na quantidade, na ordem e do tipo definido para aquela função, separados por vírgula.

Ao chamar uma função passando variáveis como parâmetros, estamos usando apenas os seus valores que serão copiados para as variáveis parâmetros da função.

### 5.1 Exemplo

```
#include <stdlib.h>
#include <stdio.h>
void calcula(int a, int b){
    int soma;
    soma = a + b;
    printf("Soma = %d\n",soma);
}
int main ( ) {
    int a,b;
    printf("entre com dois numeros\n");
    scanf("%d%d",&a,&b);
    calcula(a,b);
    system("pause");
    return 0;
}
```

#### Atenção:

- Os parâmetros formais definidos em uma função têm endereço de memória diferentes das variáveis da função que chama. Ou seja, as variáveis "a" e "b" da função calcula são diferentes das variáveis "a" e "b" da função main.
- Funções só podem ser definidas fora de outras funções.

Os nomes das variáveis dos parâmetros não precisam ser necessariamente iguais aos nomes das variáveis que foram passadas como argumentos. Poderia ter:

```
#include <stdlib.h>
#include <stdio.h>
void calcula(int a, int b){
    int soma;
    soma = a + b;
    printf("Soma = %d\n",soma);
}
int main ( ) {
    int valor1,valor2;
    printf("entre com dois numeros\n");
    scanf("%d%d",&valor1,&valor2);
    calcula(valor1,valor2);
    system("pause");
    return 0;
}
```

## 6 Funções com retorno

Ao chamar uma função a mesma pode retornar algum valor. Neste caso, precisamos definir na função um tipo de retorno diferente de void. Podemos atribuir o retorno da função a uma variável ou podemos utilizar este retorno em qualquer lugar que aceite uma expressão.

## 6.1 Exemplo

<pre>#include &lt;stdlib.h&gt; #include &lt;stdio.h&gt; int calcula(int a , int b){     return a + b; } int main ( ) {     int soma,a,b;     printf("entre com dois numeros\n");     scanf("%d%d",&amp;a,&amp;b);     soma = calcula(a,b);     printf("Soma = %d\n",soma);     system("pause");     return 0; }</pre>	<pre>#include &lt;stdlib.h&gt; #include &lt;stdio.h&gt; int calcula(int a , int b){     return a + b; } int main ( ) {     int soma,a,b;     printf("entre com dois numeros\n");     scanf("%d%d",&amp;a,&amp;b);     printf("Soma = %d\n",calcula(a,b));     system("pause");     return 0; }</pre>
---	--

### Atenção:

- Digamos que uma função está sendo executada. Quando se chega a uma declaração return a função é encerrada imediatamente e, se o valor de retorno é informado, a função retorna este valor. É importante lembrar que o valor de retorno fornecido tem que ser compatível com o tipo de retorno declarado para a função.
- Uma função pode ter mais de uma declaração return. Isto se torna claro quando pensamos que a função é terminada quando o programa chega à primeira declaração return.

## 7 Exercícios

- Desenvolva um programa que lê três notas de um aluno e chama uma segunda função que recebe estas notas, calcula e mostra a média aritmética dessas notas.
- Desenvolva um programa que lê três notas de um aluno. A função main então deve chamar uma segunda função que calcula e mostra a média aritmética dessas notas. Depois crie uma terceira função que recebe o valor da média e informa se o aluno foi aprovado ou reprovado (nota mínima para aprovação: 6,0).
- Desenvolva um programa que lê um valor do teclado.
  - Se o valor lido for 1, chamará uma função que lê o valor do raio de uma, calcula e mostra o volume (V) da referida esfera usando a seguinte fórmula:

$$V = \frac{4}{3}\pi r^3$$

- Se o valor lido for 2, chamará outra função que lê o valor do aresta de um cubo, calcula e mostra o volume (V) do referido cubo usando a seguinte fórmula:  

$$V = \text{aresta} * \text{aresta} * \text{aresta}$$
- Crie um programa que lê um valor de ângulo e pergunta se o mesmo está em graus ou radianos. Se estiver em graus chama uma função que converte para radianos e mostra o resultado. Se estiver em radianos, chama outra função que converte para graus e mostra o resultado.
- Faça um programa que lê dois valores diferentes e chama uma função para saber qual dos dois é o maior.