

## PROTÓTIPOS DE FUNÇÕES/PROCEDIMENTOS

Uma função (ou procedimento) deve ser declarada sempre antes da sua primeira utilização. É por essa razão que devemos escrever todas as funções antes da função `main()`. Existe uma maneira de utilizar uma função em C escrita após a função `main()`. Basta que utilizemos Protótipos de função.

O protótipo de função (ou procedimento) nada mais é do que a própria declaração da função, antes da função `main()`, porém sem a sua descrição e seguida de um ponto e vírgula. Dessa forma, a declaração da função é feita, mas a função não é escrita no momento. Veja no exemplo:

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

void doisbeeps(); // protótipo de função

int main()
{
    printf("Vamos ouvir dois beeps: tecla qualquer tecla para ouvir:");
    getche();
    doisbeeps();
    return 0;
}

void doisbeeps()
{
    int k;
    printf("\x7");
    for(k=0;k<5000;k++);
    printf("\x7");
}
```

Observando esse programa acima, percebemos que o procedimento `doisbeeps()` foi escrito após a função `main()`. Mas, se olharmos acima do `main()` encontraremos o que chamamos de **Protótipo de função**, que é a declaração do procedimento seguido de um ponto e vírgulas. Se o procedimento (ou função) receber qualquer parâmetro, eles deverão ser especificados também. O protótipo de função é uma cópia idêntica à declaração da função / procedimento.

## UTILIZANDO MAIS DE UMA FUNÇÃO NO MESMO PROGRAMA

Em um mesmo programa podemos utilizar mais de uma função ou procedimento, com ou sem utilização de protótipos de função. Observe o programa abaixo.

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

void linha(int x);

void espacos(int y)
{
    int i;
    for(i=0;i<y;i++)
        printf(" ");
}

int main()
{
    printf("\n\n\n\n\n\n\n\n\n\n");
    espacos(30);
    linha(20);
    espacos(30);
    printf("\xDB UM PROGRAMA EM C \xDB\n");
    espacos(30);
    linha(20);
    printf("\n\n\n\n\n\n\n\n\n\n");
    system("PAUSE");
    return 0;
}

void linha(int x)
{
    int i;
    for(i=0;i<x;i++)
        printf("\xDB");
    printf("\n");
}
```

## UTILIZANDO UMA FUNÇÃO COMO PARÂMETRO DE OUTRA FUNÇÃO

Como uma função retorna um valor, ela pode ser utilizada para chamar outra função como se fosse um valor. Veja o exemplo:

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

float somasqr(float m, float n);
float sqr(float z);
float soma(float x, float y);

int main()
{
    float a,b;
    printf("Calcula a soma do quadrado de dois numeros");
    printf("\n\nDigite o primeiro numero: ");
    scanf("%f",&a);
    printf("\n\nDigite o segundo numero: ");
    scanf("%f",&b);
    printf("\n\nA soma dos quadrados dos numeros digitados e
%f\n\n",somasqr(a,b));
    system("PAUSE");
    return 0;
}

float somasqr(float m, float n)
{
    return soma(sqr(m),sqr(n));
}

float sqr(float z)
{
    return z*z;
}

float soma(float x, float y)
{
    return x+y;
}
```

## UTILIZANDO UMA FUNÇÃO RECURSIVA

Uma função é chamada recursiva quando ela chama ela mesma. Veja no exemplo abaixo uma função recursiva:

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

float fatorial(float num)
{
    if(num==1)
        return 1;
    else
        return num*fatorial(num - 1);
}

int main()
{
    float n;
    printf("Calcular o fatorial de um numero com funcao recursiva");
    printf("\n\n\nDigite o numero a ser calculado: ");
    scanf("%f",&n);
    printf("\n\nO fatorial do numero e %f \n\n",fatorial(n));

    system("PAUSE");
    return 0;
}
```

# Exercícios de Funções Recursivas

1. Escreva uma função recursiva **potencia(base,expoente)** que, quando chamada, retorna  $\text{base}^{\text{expoente}}$ . Por exemplo,  $\text{potencia}(3,4) = 3*3*3*3$ . Assuma que expoente é um inteiro maior ou igual a zero.

Dica: o passo de recursão deve utilizar o relacionamento  $\text{base}^{\text{expoente}} = \text{base} * \text{base}^{\text{expoente}-1}$

e a condição de terminação ocorre quando expoente é igual a 0, porque

$$\text{base}^0 = 1$$

Incorpore essa função em um programa que permita que o usuário informe a base e o expoente da potenciação e receba o resultado desejado.

2. Faça um programa que calcule a soma dos números naturais de 0 até um valor x digitado pelo usuário, utilizando função recursiva e protótipo de função. A função deverá obedecer aos critérios:

função: `int soma(int num)`

$\text{soma}(n) = n + \text{soma}(n-1)$

$\text{soma}(0) = 0$

O usuário deverá informar o número desejado e o programa retorna a soma dos números. Por exemplo: usuário informa 6, resultado=21.

3. Escreva um programa que calcule o valor de um número da sequência Fibonacci, utilizando recursividade e escrevendo a função depois do programa principal.

Nota: A sequência Fibonacci (1,1,2,3,5,8,13,21,...,n) é calculada da seguinte forma:

$\text{Fib}(1) = 1$

$\text{Fib}(2) = 1$

$\text{Fib}(3) = \text{Fib}(2) + \text{Fib}(1)$

$\text{Fib}(4) = \text{Fib}(3) + \text{Fib}(2)$

$\text{Fib}(5) = \text{Fib}(4) + \text{Fib}(3)$

...

$\text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2)$

4. Escreva uma função em C que calcule o número de divisores entre 1 e um número informado pelo usuário. Esta função deverá utilizar o seguinte protótipo:

`int divisores(int num);`

5. Escreva uma função que leia um valor  $x$  qualquer e retorne o valor de  $S$ , utilizando protótipo de função.

$$S = 1/2 + 1/3 + 1/4 + 1/5 + \dots + 1/x$$