

**BCC701 – Programação de Computadores I**  
Universidade Federal de Ouro Preto  
Departamento de Ciência da Computação

**[www.decom.ufop.br/bcc701](http://www.decom.ufop.br/bcc701)**  
**2015/1**



# Funções.

Material Didático Unificado.

## Propósitos do Uso de Funções

- Modularizar um programa em partes menores;
- Executar uma tarefa que é frequentemente solicitada;
- Aumentar a legibilidade e manutenibilidade do programa;
- Implementar as chamadas UDF (**U**ser **D**efined **F**unctions), para complementar as necessidades do programador na execução de tarefas não suportadas pelo ambiente de programação.

## Exemplo de Uso de Funções

- Cálculo do número de combinações de  $n$  tomados  $k$  a  $k$ ;
- Observe que o cálculo do fatorial é repetido três vezes.

$$\binom{n}{k} = \frac{n!}{(n - k)! k!}$$

## Exemplo de Uso de Funções

- Para calcular o fatorial de um número inteiro  $n$  pode-se usar o seguinte trecho de programa:

```
fat = 1;  
for i = 1:n  
    fat = fat * i;  
end
```

- Entretanto é necessário adaptar este código para obter o cálculo do número de combinações:

## Introdução

### Exemplo de Uso de Funções

```
n = input("n="); k = input("k=");
```

```
fat_n = 1;  
for i = 2:n  
    fat_n = fat_n * i  
end
```

```
fat_n-k = 1;  
for i = 2:(n - k)  
    fat_n-k = fat_n-k * i  
end
```

```
fat_k = 1;  
for i = 2:k  
    fat_k = fat_k * i  
end
```

```
nComb = fat_n / (fat_n-k * fat_k);
```

- Agora o programa anterior será dividido em duas partes: o programa principal e a função;
- O programa principal será codificado da seguinte forma:

```
n = input("n=") ; k = input("k=") ;  
nComb = fatorial(n) / ...  
         fatorial(n - k) * fatorial(k) ;
```

A função será codificada da seguinte forma:

```
function fat = fatorial(n)
    fat = 1;
    for i = 1:n
        fat = fat * i;
    end
endfunction
```

- Um programa é designado principal quando ele faz chamadas as funções.
- A execução de um programa com funções se inicia pelo programa principal.
- A execução de uma chamada transfere o controle de execução para a função.
- Ao término da execução da função, o controle é devolvido ao ponto de chamada, em uma operação chamada de retorno da função.

# Sintaxe de Função

**Parâmetro de Saída:**  
calculado pela função

```
function fat = fatorial(n)
    fat = 1;
    for i = 1:n
        fat = fat * i;
    end
endfunction
```

**Parâmetro de Entrada:** fornecido  
na chamada da função

# Sintaxe de Função: Vários Parâmetros

```
function [x1, x2] = eq2g(a, b, c)
    delta = b^2 - 4 * a * c;
    x1 = (-b + sqrt(delta)) / (2 * a);
    x2 = (-b - sqrt(delta)) / (2 * a)
endfunction
```

```
// Programa Principal;
```

```
x = 2; y = 4; z = 6;
```

```
[raiz_1, raiz_2] = eq2g(x, y, z);
```

# Observações: Funções

- Uma função cria um espaço novo para as variáveis, que podem ter nomes iguais aos de variáveis já definidas no programa principal.
- As variáveis definidas por uma função são denominadas variáveis locais.
- As variáveis definidas no programa principal são denominadas variáveis globais.
- Mais sobre funções: Introdução à Organização e à Programação de Computadores – Prof. Oswaldo Carvalho.

# Exemplo 1

Codifique um programa que faça a leitura de  $n$  valores através do teclado.

Cada valor lido no teclado deve ser aplicado á função  $f(x) = x - \text{sqrt}(x)$ . O resultado da aplicação da função deve ser acumulado em um somatório.

O cálculo de  $f(x)$  deve ser codificado em uma função definida pelo usuário.

Ao final o programa imprime o valor do somatório calculado.

# Exemplo 1

```
function f = minhaF(x)
    f = x - sqrt(x);
endfunction
```

```
n = input("QUANTIDADE DE LEITURAS: ");
soma = 0;
for i = 1:n
    x = input("DIGITE UM VALOR: ");
    soma = soma + minhaF(x);
end
printf("\nSOMATÓRIO CALCULADO: %7.3f",
    soma);
```

## Exemplo 2

Codifique um programa que calcule a série a seguir, onde  $n$  é o número de parcelas.

Cada parcela contém um numerador e um denominador. O Cálculo de ambos deve ser feito por funções definidas pelo usuário.

Ao final o programa imprime o valor da série.

$$\sum_{i=1}^n \frac{i - \text{sen}(i)}{i^3 - \cos(2i)}$$

## Exemplo 2

```
function resposta = numerador(x)
```

```
    resposta = x - sin(x);
```

```
endfunction
```

```
// -----
```

```
function resposta = denominador(x)
```

```
    resposta = x^3 - cos(2 * x);
```

```
endfunction
```

```
// -----
```

```
n = input("QUANTIDADE DE PARCELAS: ");
```

```
soma = 0;
```

```
for i = 1:n
```

```
    soma = soma + numerador(i) / ...  
                denominador(i);
```

```
end
```

```
printf("\nSOMATÓRIO CALCULADO: %7.3f",  
    soma);
```