

# 1 Abstração: funções

**ATENÇÃO:** este texto é só um resumo! Você pode aprender mais sobre funções no livro-texto: leia o capítulo 6 (ele chama funções de “módulos”).

**Função:** Um nome dado a uma sequência de comandos que, após serem executados, produzem um resultado. Pode-se passar argumentos para a função. Por exemplo, o “fatorial de n” é uma função com o argumento “n” e que retorna um valor (o fatorial de n).

**Como:** Escrevemos “funcao”, e em seguida:

- Uma lista de argumentos entre parênteses:
- Dois pontos
- Declaração de variáveis locais (que só são usadas dentro da função);
- O código da função
- “retorne X” é usado para dizer que o valor da função é X.

Por exemplo, se definirmos a função fatorial:

```
funcao fatorial (n):  
  int i, f  
  f <- 1  
  para i em (1..n):  
    f <- f * i  
  retorne f
```

Note que declaramos os tipos das variáveis i e f, que são usadas dentro da função. A declaração ficou dentro da função. Estas variáveis *não* podem ser usadas fora desta função.

Veja também a última linha: ela diz qual deve ser o *valor de retorno* da função: sempre que, no programa, usarmos “fatorial(x)”, u

Podemos usá-la depois no programa:

```
// Este programa calcula combinacoes  
leia n  
leia k  
c <- fatorial(n) / fatorial(k) * fatorial (n-k)
```

Note que se não tivéssemos definido a função “fatorial”, teríamos que escrever o código “para i ...” três vezes (porque usamos o fatorial três vezes no programa).

Esta é uma finalidade da abstração: reusar código

Outro efeito da abstração é a simplificação do programa. A linha

```
c <- fatorial(n) / fatorial(k) * fatorial (n-k)
```

é fácil de ler e lembra a fórmula que usaríamos ( $C_{n,k} = \frac{n!}{k!(n-k)!}$ ).

Uma função pode não retornar valor algum. Por exemplo,

```
funcao pontos (n):  
  int i  
  para i em (1..n):  
    mostre "."
```

O que importa é que a função `pontos` mostra n pontos na tela; ela não precisa retornar valor algum. Podemos usá-la assim:

```
mostre nome  
pontos (10)  
mostre telefone
```

O trecho de programa acima produzirá a saída:

Joao.....3322-4595

## 1.1 Um exemplo completo

O exemplo a seguir lê uma seqüência de  $n$  strings, depois lê um número  $k$  e diz ao usuário quantas combinações podem ser feitas com as  $n$  strings, tomadas  $k$  de cada vez.

```
int q, c, i
string s

funcao fatorial (int n):
  int i, f
  f <- 1
  para i em (1..n):
    f <- f * i
  retorne f

funcao combinacao (int n, int k):
  retorne fatorial(n) / fatorial(k) * fatorial (n-k)

mostre "Digite a lista de strings. Quando quiser parar, digite
      uma string vazia"
i <- 0
leia s
enquanto (s != ""):
  i <- i + 1
  leia s

mostre "Quantas strings de cada vez?"
leia q

c <- combinacao (i, q)
mostre "Ha' ", c, " combinacoes."
```