Linguagem de Programação S Fundamentos e Aplicações em Recursos Florestais



5. Noções de Programação

5.2. Funções Simples

5.2.1. A Estrutura Básica de uma Função

Toda manipulação de dados e análises gráficas ou estatísticas no R são realizadas através de funções. Entretanto, você não precisa ser um programador experimentado para construir algumas funções simples para facilitar a atividade de manipulação de dados.

A estrutura básica de uma função é:

```
#########################################################
########### 5 NOÇÕES DE PROGRAMAÇÃO
########### 5.2 Funções Simples
#########################################################
#  5.2.1. A Estrutura Básica de uma Função

minha.funcao <- function( argumento1, argumento2, argumento3, . . .)
     {
           comando 1
           comando 2
           comando 3
           . . .
           comando n
     }

```

Os elementos dessa expressão são:

- **minha.funcao** é o nome que a nova função receberá;
- **function** é a expressão no R que cria uma nova função;
- entre os parênteses "**()**" são listados (separados por vírgula) os argumentos necessários a função;
- entre as chaves "**{}**" são listados os comandos da função, sempre com um comando por linha.

Vejamos um exemplo simpes:

```
16 sincos <- function(x)
17  {
18        sin(x) * cos(x)
19  }
20 sincos(10)
21 sin(10) * cos(10)
22 sincos(pi)
23 sin(pi) * cos(pi)
24
```

5.2.2. Definindo Argumentos

Todos argumentos de uma função tem seu respectivo nome. Ao evocar a função podemos fazê-lo de duas formas:

- utilizando o nome dos argumentos em qualquer ordem;
- utilizando a ordem dos argumentos, mas omitindo os nomes.

Vejamos um exemplo com os dados do arquivo "egrandis.csv".

```
25 # 10.2.2. Definindo Argumentos
26
27 egr = read.csv("egrandis.csv")
28 plot( col="red", pch=2, y=egr$ht, x=egr$dap )
29 plot( egr$dap, egr$ht )
30
```

Para qualquer argumento podemos definir um valor default apresentando esse valor junto com argumento na definição da função:

```
31 myplot <- function(..., cor="red") { plot(..., col=cor) }
32 myplot( egr$dap, egr$ht )
33 myplot( ht ~  dap, data=egr )
34
```

O exemplo acima também mostra a função do argumento "`. . .`". Esse argumento representa qualquer argumento adicional que desconhecemos, mas que desejamos que seja passado para as funções dentro da função que estamos construindo.

5.2.3. Um Aspecto Prático na Edição de Funções

É natural se definir funções **particulares a um dado projeto de análise de dados** no próprio "script" em que se programa a análise.

Contudo, quando se definem diversas funções que serão utilizadas em diferentes **projetos de análise de dados**, é interessante definir as funções num arquivo exclusivo para as funções (p.ex., arquivo "**`minhas-funcoes.R`**"). Esse arquivo é como um script (mas apenas com as definições das funções) e pode ser editado utilizando a janela de edição de scripts da interface utilizada (p.ex. RStudio). Para ler o arquivo em qualquer sessão do R utiliza-se o comando **`source`**:

```
35 # 5.2.3. Um Aspecto Prático na Edição de Funções
36
37 source( "minhas-funcoes.R" )
38
```

From:
http://insilvaarbores.com.br/dokuwiki/ - **In Silva, Arbores ...**

Permanent link:
http://insilvaarbores.com.br/dokuwiki/doku.php?id=pt:cursos_online:s_linguagem:5-nocoes-programacao:5-2-funcoes-simples

Last update: **2026/08/26 21:08**