Introdução ao programa R

Ronald Targino, DEMA-UFC

Notas de aula

2. Objetos

2.1 Vetores

```
# ------------------------------------------------------------------------
# PARTE 1

# Atenção: evite usar palavras ou letras reservadas do R para o nome dos objetos.
# Exemplo: T, F, c, sd, var, NA ...

# Orientação: todas as instruções (códigos, comentários) abaixo "devem"" ser digitas em
# uma janela de Edição (R script); após cada instrução aperte Ctrl+Enter; os resultados
# (indicados, neste PDF, com o símbolo ## no início da linha) serão mostrados na janela
# Console. Caso digite as instruções no Console, aperte o Enter em vez de Ctrl+Enter.

# Inicialmente, vamos remover todos os objetos (disponíveis/ativos) no R:
# Digite o código (instrução) rm(list=ls()) e aperte Ctrl+Enter
rm(list=ls()) # remover todos os objetos

a = 2  # atribui ao objeto 'a' o valor 2; 'a' é um vetor de comprimento 1
a
```

```
## [1] 2
```

```
b = 6
b
```

```
## [1] 6
```

```
a; b  # use o ponto e vírgula (;) para separar instruções na mesma linha
```

```
## [1] 2
```

```
## [1] 6
```

```
d = 7
e = 8
a + b; a-b; a*b; a/b # operações aritméticas
```

```
## [1] 8
```

```
## [1] -4
```

```
## [1] 12
```

```
## [1] 0.3333333
```

```
k = scan()      # criar o vetor z
w1 = a:b        # criar o vetor z com valores de 'a' a 'b' com incremento 1
w2 = 5.5:10
w3 = -2:-6
```

```r
z = c(3, 2, 4) # criar o vetor z com valores/elementos 3, 2 e 4

ls() # mostrar todos os objetos disponíveis/ativos
```

```
## [1] "a"  "b"  "d"  "e"  "k"  "w1" "w2" "w3" "z"
```

```r
rm("a") # remover o objeto "a"

ls()
```

```
## [1] "b"  "d"  "e"  "k"  "w1" "w2" "w3" "z"
```

```r
rm("b","e") # remover os objetos "d" e "e"
ls()
```

```
## [1] "d"  "k"  "w1" "w2" "w3" "z"
```

```r
rm(list=ls()) # remover todos os objetos
ls()
```

```
## character(0)
```

```r
x = vector(mode = "numeric")   # criar o vetor numérico vazio 'x'
x
```

```
## numeric(0)
```

```r
mode(x)  # tipo de vetor
```

```
## [1] "numeric"
```

```r
x[2] = 4; x # adiciona o valor 4 na posição 2
```

```
## [1] NA  4
```

```r
x[5] = 8
x # NA ("Not Available")
```

```
## [1] NA  4 NA NA  8
```

```r
y = vector(mode = "numeric", length = 3) # especificando tipo e comprimento
y
```

```
## [1] 0 0 0
```

```r
y[1] = 6; y
```

```
## [1] 6 0 0
```

```r
y[4] = 8; y
```

```
## [1] 6 0 0 8
```

```r
y[5:9] = 1; y
```

```
## [1] 6 0 0 8 1 1 1 1 1
```

```r
# As funções 'floor' e  'ceiling' operam um número real para o maior inteiro anterior
# e o menor inteiro seguinte, respectivamente. Em outras palavras,  floor(x) retorna
# o maior inteiro que não seja maior que 'x' e ceiling(x), o menor inteiro que não
# seja menor que 'x'.
```

```r
# ------------------------------------------------------------------------
# PARTE 2
# Operações com vetores

c(1, 2, 3, 4, 5) + c(1, 2) # atenção!
```

```
## Warning in c(1, 2, 3, 4, 5) + c(1, 2): longer object length is not a
## multiple of shorter object length
```

```
## [1] 2 4 4 6 6
```

```r
a = 1:5
b = 1:5
soma = a + b
a; b; soma
```

```
## [1] 1 2 3 4 5
```

```
## [1] 1 2 3 4 5
```

```
## [1]  2  4  6  8 10
```

```r
a = c(4, 3, 2)
b = c(2, 3, 4)
a/b
```

```
## [1] 2.0 1.0 0.5
```

```r
a * b # multiplicação elemento a elemento
```

```
## [1] 8 9 8
```

```r
a%*%b # multiplicação de vetores
```

```
##      [,1]
## [1,]   25
```

```r
(1:6)^2
```

```
## [1]  1  4  9 16 25 36
```

```r
(1:6)^(1:2)
```

```
## [1]  1  4  3 16  5 36
```

```r
x = c(1,4,9)
sqrt(x)
```

```
## [1] 1 2 3
```

```r
sum(x^2)
```

```
## [1] 98
```

```r
sum(x)^2
```

```
## [1] 196
```

```r
x = c(-1,4,9,16)
x
```

```
## [1] -1  4  9 16
```

```r
sqrt(x)# NaN ("Not a Number"): cálculo cujo resultado não é definido
```

```
## Warning in sqrt(x): NaNs produced
```

```
## [1] NaN   2   3   4
```

```r
x = letters[1:10]
x
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
```

```r
a = x[c(2,4)]; a
```

```
## [1] "b" "d"
```

```r
b = x[1:3]; b
```

```
## [1] "a" "b" "c"
```

```r
d = x[-(1:3)]; d
```

```
## [1] "d" "e" "f" "g" "h" "i" "j"
```

```r
e = x[-c(1:3,5,10)]; e
```

```
## [1] "d" "f" "g" "h" "i"
```

```r
x
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
```

```r
indice = match(c("a","k"), letters)
letters[indice[1]:indice[2]]
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k"
```

```r
x[5] = 3
x[6] = "tempo"
x
```

```
##  [1] "a"     "b"     "c"     "d"     "3"     "tempo" "g"     "h"
##  [9] "i"     "j"
```

```r
x[5:6] = letters[5:6]
x
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
```

```r
x = x[-2]
x
```

```
## [1] "a" "c" "d" "e" "f" "g" "h" "i" "j"
```

```r
x = LETTERS[1:10]
x
```

```
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J"
```

```r
x[-c(1,3,4)]
```

```
## [1] "B" "E" "F" "G" "H" "I" "J"
```

```r
x = 1:5
x
```

```
## [1] 1 2 3 4 5
```

```r
x[2] = 20; x
```

```
## [1]  1 20  3  4  5
```

```r
x[3:5] = 100; x
```

```
## [1]   1  20 100 100 100
```

```r
x[3:5] = c(11, 22, 33); x
```

```
## [1]  1 20 11 22 33
```

```r
y = 10:20; y
```

```
##  [1] 10 11 12 13 14 15 16 17 18 19 20
```

```r
y[c(1,3,8:10)] = 999; y
```

```
##  [1] 999  11 999  13  14  15  16 999 999 999  20
```

```r
# ------------------------------------------------------------------------
# PARTE 3
# Vetores a partir da funções 'seq'e 'rep'
# Construção de sequências
# Função: seq(valor inicial, valor final (limite), incremento, comprimento da sequência)
# Função: seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)), length.out = NULL)
#
seq()           # ou seq(from = 1, to = 1)
```

```
## [1] 1
```

```r
seq(2, 5)       # ou seq(from = 2, to = 5)
```

```
## [1] 2 3 4 5
```

```r
seq(2, 5.2)
```

```
## [1] 2 3 4 5
```

```r
seq(2, 5.8)
```

```
## [1] 2 3 4 5
```

```r
seq(1.3, 5)
```

```
## [1] 1.3 2.3 3.3 4.3
```

```r
seq(2.7, 6.6)
```

```
## [1] 2.7 3.7 4.7 5.7
```

```r
seq(5, -2)
```

```
## [1]  5  4  3  2  1  0 -1 -2
```

```r
seq(1, 10, 2)   # ou seq(from = 1, to = 10, by = 2)
```

```
## [1] 1 3 5 7 9
```

```r
seq(1, 10, 4)
```

```
## [1] 1 5 9
```

```r
seq(1, 10, 2.3)
```

```
## [1] 1.0 3.3 5.6 7.9
```

```r
seq(10, -5, -2.3)
```

```
## [1] 10.0  7.7  5.4  3.1  0.8 -1.5 -3.8
```

```r
seq(10, 20, length = 6) # length ou length.out: comprimento da sequência
```

```
## [1] 10 12 14 16 18 20
```

```r
seq(10, 20, length = 5.4)
```

```
## [1] 10 12 14 16 18 20
```

```r
seq(10, 30, length = 10)
```

```
##  [1] 10.00000 12.22222 14.44444 16.66667 18.88889 21.11111 23.33333
##  [8] 25.55556 27.77778 30.00000
```

```r
# ------------------------------------------------------------------------
# PARTE 3 - Exercícios
# Vetores a partir da funções 'seq'e 'rep'
# Função: rep(vetor de elementos a serem replicados, vetor indicando o número de vezes
# que cada elemento será replicado, comprimento desejado do vetor a ser gerado, número
# de vezes que cada elemento é replicado)
# Função: rep(x, times, length.out, each)
# Escreva as funções que geraram os resultados abaixo.
```

```
## [1] 2 2 2 2 2
```

```
##  [1] 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100
## [18] 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100
## [35] 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100
## [52] 100 100 100 100
```

```
##  [1] 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1
```

```
##  [1] 1 2 2 2 2 2 3 3 3 3 3 3 3 3 3
```

```
##  [1]  1  1  2  2  3  3  4  4  5  5  6  6  6  6  7  7  7  7  8  8  8  8  9
## [24]  9  9  9 10 10 10 10
```

```
##  [1]  1  1  2  2  3  3  4  4  5  5  6  6  6  6  7  7  7  7  8  8  8  8  9
## [24]  9  9  9 10 10 10 10
```

```
## [1] 1 1 1 2 2 2 3 3 3
```

```
##  [1] 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
```

```
##  [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4
```

```
##  [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
## [36] 4 4 4 4 4 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4 1 1 1 1 1 2 2 2 2 2
## [71] 3 3 3 3 3 4 4 4 4 4
```

```
##  [1] 1 1 1 2 2 2 3 3 3 4
```