Módulo Básico

IME - USP

29 de setembro de 2014

Contents

1 A FAZER******** 1**

2 Introdução à linguagem *script* 1

3 Sintaxe 1

3.1 Operadores Aritméticos 1

3.2 Operadores Lógicos 2

3.3 Operadores diversos 4

3.4 Precedência de operadores 5

3.5 Controle de fluxo 5

1 A FAZER**********

- Tudo.

2 Introdução à linguagem *script*

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

3 Sintaxe

3.1 Operadores Aritméticos

Operadores são funções representadas por símbolos reservados que geralmente são construídos com o intuito de fazer a linguagem mais semelhante às notações matemáticas convencionais. Exemplos de operadores são

os aritméticos, como `-`, `+`, `/` e `*`, (que falaremos a seguir). Estes operadore se dividem em duas classes: os unários, que operam em apenas um objeto; e os binários, que relacionam dois objetos.

Operador	Descrição
x + y	Adição de x com y
x - y	Subtração de y em x
x * y	Multiplicação de x e y
x / y	Divisão de x por y
xˆy ou x ** y	x elevado a y-ésima potência
x %% y	Resto da divisão de x por y (módulo)
x %/% y	Parte inteira da divisão de x por y

Tabela: Lista de operadores aritméticos

Exemplos:

```
1 + 1
```

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

```
10 - 8
```

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

```
2*10
```

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

```
18/3
```

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

```
2^4
```

```
## [1] 16
```

```
9%%2
```

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

```
9%/%2
```

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

3.2 Operadores Lógicos

Operadores lógicos retornarão sempre ou `TRUE` ou `FALSE`, conhecidos também como valores **booleanos**. Eles definem perguntas que aceitam apenas verdadeiro e falso como resposta, como sugere o quadro abaixo.

Operador	Descrição
x < y	x menor que y?
x <= y	x menor ou igual a y?
x > y	x maior que y?
x >= y	x maior ou igual a y?
x == y	x igual a y?
x != y	x diferente de y?
!x	Negativa de x
x | y	x ou y são verdadeiros?
x & y	x e y são verdadeiros?

Tabela: Lista de operadores lógicos

Os operadores `<`, `>`, `<=`, `>=`, `!=`, `==` são usados tanto na matemática quanto em programação e bastante intuitivos. Eles retornarão `TRUE` ou `FALSE` (como bons operadores lógicos) dependendo da veracidade da sentença. Ou seja, se dissermos ao R que `1 != 1` ele prontamente negará tal absurdo retornado `FALSE` como resposta. Abaixo seguem alguns exemplos de testes lógicos entre dois objetos:

```
1 < 1
```

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

```
1 <= 1
```

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

```
c(9, 10) >= 9
```

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

```
1 == 0.999
```

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

```
1 != 0.99999999999999999
```

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

```
"tudo minúsculo" == "TUDO MAIÚSCULO"
```

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

```
"tudo minúsculo" == "tudo minúsculo"
```

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

Muitas vezes precisamos combinarmos diversos testes para tomar uma única decisão. Por exemplo, se quisermos certificar que um valor `x` é um número no intervalo unitário, precisamos saber se `x > 0` e se `x < 1` simultaneamente. Para esse tipo de situação, os operadores `|` (OU) e `&` (E) vêm a calhar. Diferentemente dos operadores visto acima, os valores que eles recebem também devem ser booleanos, ou seja, deve ser passado `TRUE` ou `FALSE` em ambos os lados do operador. Assim, o R entenderia o que gostaríamos de testar para o x com a sentença `x > 0 & x < 1` (em português, lê-se: x é maior que zero E x é menor que 1). Neste caso, se as duas condições forem satisfeitas, então o R retornará `TRUE`.

Para entender melhor

Combinando vários testes lógicos:

```
!TRUE
```

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

```
TRUE | FALSE
```

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

```
FALSE | FALSE
```

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

```
TRUE & FALSE
```

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

```
TRUE & TRUE
```

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

```
(1 == 1 | 2 > 3 | 100 == 0 | "a" != "b") & (4 < 5 | 3 > 0)
```

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

3.3 Operadores diversos

Operador	Descrição
x : y	Sequência de x até y
x = y	x recebe y (atribuição)
?x	Documentação de x
x$y	Extração de y do objeto x
x %*% y	Multiplicação matricial das matrizes x e y

Tabela: Lista de operadores diversos

3.4 Precedência de operadores

Assim como na matemática, operadores têm ordem para serem efetuados, tal como `*` antes de `+`.

3.5 Controle de fluxo

Como em toda boa linguagem de programação, o R possui seus dispositivos para controle de fluxo. Entende-se por controle de fluxo como o conjunto de funções e operadores responsáveis por introduzir condições lógicas com a finalidade de decidir como proceder durante uma execução.

3.5.1 `if`, `else` e `else if`

3.5.2 switch

3.5.3 Laços

3.5.3.1 `while`

3.5.3.2 `for`

3.5.3.3 `repeat`

3.5.3.4 `break` e `next`

```
cat("<table class='container'><tr>")
```

```
cat("<td>")
```

```
plot( rnorm(10) )
```

```
cat("</td>")
```

```
cat("<td>")
```

```
rnorm(10) %>% pander
```

-0.8471, *0.9754*, *-1.1996*, *0.2364*, *-0.8105*, *0.3808*, *-0.5809*, *-0.9274*, *-0.8661* and *-1.5609*

```
cat("</td>")
```

```
cat("</tr></table>")
```