Aula 1

Eduardo

September 26, 2016

Em nossa primeira aula discutimos a instalação do R e Rstudio, instalação de pacotes e uma idéia inicial do uso do Rmarkdown para elaboração de relatórios integrados com R junto ao texto.

Esse documento é um exemplo da utilização de Rmarkdown para revisarmos os conceitos da linguagem R vistas em aula.

Calculadora simples

Vimos que podemos utilizar o R como uma simples calculadora:

```
2+2
```

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

Podendo realizar operações mais elaboradas:

```
30*(573-273)/22 + 14^2
```

```
## [1] 605.0909
```

Tendo a possibilidade de atribuir o resultado de determinado cálculo a um objeto com um nome escolhido:

```
(resultado <- 122/5 + 91)
```

```
## [1] 115.4
```

E sempre que fizermos referência ao objeto resultado seu valor será recuperado:

```
resultado
```

```
## [1] 115.4
```

```
resultado + 1000
```

```
## [1] 1115.4
```

E assim podemos utilizar esses objetos para facilitar a realização de cálculos. Uma extensão natural do uso desses objetos que representam um número é a utilização de vetores.

Vetores

Para criar um vetor vamos utilizar o comando *c()* e incluimos os elementos do vetor separados por vírgulas:

```
primeiro_vetor <- c(1,30,500)
primeiro_vetor
```

```
## [1]   1  30 500
```

```
segundo_vetor <- c(100,3,50)
segundo_vetor
```

```
## [1] 100   3  50
```

E sempre que utilizarmos o comando *c()* os elementos serão concatenados em um vetor. No exemplo seguinte vamos criar um vetor concatenando os dois vetore criados anteriormente.

```
vetores_reunidos <- c(primeiro_vetor,segundo_vetor)
vetores_reunidos
```

```
## [1]   1  30 500 100   3  50
```

Assim como vimos a utilização de um objeto que armazenava um único número para realização de cálculos podemos utilizar também os objetos que armazenam vetores para realização de cálculos. Como primeiro exemplo vamos somar 1000 ao primeiro_vetor:

```
primeiro_vetor
```

```
## [1]   1  30 500
```

```
primeiro_vetor + 1000
```

```
## [1] 1001 1030 1500
```

Ou alternativamente podemos calcular 25% de cada um dos valores do vetor

```
primeiro_vetor
```

```
## [1]   1  30 500
```

```
primeiro_vetor * 0.25
```

```
## [1]   0.25   7.50 125.00
```

Podemos também realizar uma soma entre os elementos dos dois vetores:

```
primeiro_vetor
```

```
## [1]   1  30 500
```

```
segundo_vetor
```

```
## [1] 100   3  50
```

```
primeiro_vetor +  segundo_vetor
```

```
## [1] 101  33 550
```

Note que a soma foi realizada elemento a elemento de forma que o primeiro elemento do primeiro vetor foi somado ao primeiro elemento do segundo vetor e o mesmo comportamento foi repetido para cada elemento dos vetores.

Vimos em aula que o R adota um comportamento chamado de reciclagem quando vai realizar operações com vetores de tamanhos diferentes. No exemplo a seguir vamos realizar a soma entre um vetor com 4 elementos e um vetor com 2 elementos.

```
v1 <- c(1,2,3,4)
v2 <- c(10,20)
v1
```

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

```
v2
```

```
## [1] 10 20
```

```
v1 + v2
```

```
## [1] 11 22 13 24
```

Note que os elementos do segundo vetor são reciclados e reutilizados na soma com os elementos 3 e 4 do primeiro vetor. Esse é o comportamento padrão no R e é de grande importância saber como tirar proveito desse comportamento.

Vimos que em caso das dimensões dos vetores não serem conformes o comportamento de reciclagem é utilizado mas um aviso é passado para o usuário, como pode ser visto no exemplo a seguir:

```
v1 <- c(1,2,3,4)
v2 <- c(10,20,30)
v1
```

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

```
v2
```

```
## [1] 10 20 30
```

```
v1 + v2
```

```
## Warning in v1 + v2: longer object length is not a multiple of shorter
## object length
```

```
## [1] 11 22 33 14
```

Outros tipos de dados

Vimos a manipulação e utilização de elementos e vetores de elementos de tipo numérico, mas sabemos que em aplicações reais precisaremos trabalhar com outros tipos de variávies. Vimos exemplos de variávies de tipo *string* e *lógico*

```
carros <- c("gol","corsa","uno")
carros
```

```
## [1] "gol"   "corsa" "uno"
```

```
logico <- c(T,TRUE,F,FALSE)
logico
```

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

```
valores <- c(1,5,15,20)
valores
```

```
## [1]  1  5 15 20
```

```
valores >= 10
```

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

Para criar elementos do tipo *string* precisamos utilizar as aspas para indicar que aquele elemento é do tipo *string* e não um determinado nome de objeto definido para o R.

Para criar elementos do tipo *lógico* podemos utilizar as palavras completas *TRUE* e *FALSE* ou suas abreviações *T* e *F*, lembrando que no R existe distinção entre letras maiúsculas e minúsculas e as variáveis do tipo lógico sempre serão definidas com letras maiúsculas.

No exemplo vimos que podemos criar um vetor com valores lógicos ou podemos ter esses valores lógicos como resultados de avaliações de comparações em que verificamos quais dos elementos do vetor eram maiores ou iguais ao número 10. Dentre as operações de comparação podemos utilizar:

- $>$ maior
- $<$ menor
- $==$ igualdade
- $>=$ maior ou igual
- $<=$ menor ou igaul
- $!=$ desigual

Coerção

Vimos que os vetores são objetos do tipo homogêneo, o que indica que todos os elementos de um vetor devem ser do mesmo tipo (*numérico*,*string*,*lógico*). Se colocarmos elementos de tipos diferentes dentro de um mesmo vetor o R vai aplicar uma coerção em alguns elementos para fazer com que todos tenham o mesmo tipo.

Vimos que se juntarmos elementos *numéricos* com elementos do tipo *string*, os elementos *numéricos* serão convertidos para o tipo *string*

```
mistura <- c("palavra","texto",100,200)
mistura
```

```
## [1] "palavra" "texto"   "100"     "200"
```

note que aos números foram acrescidas aspas, indicando que esses valores são do tipo *string*

Vimos que se juntarmos elementos *lógicos* com elementos do tipo *string*, os elementos *lógicos* serão convertidos para o tipo *string*

```
mistura2 <- c(TRUE, "milho", F, "soja")
mistura2
```

```
## [1] "TRUE"  "milho" "FALSE" "soja"
```

note que aos elementos *lógicos* foram acrescidas aspas, indicando que esses valores são do tipo *string*

Por fim vimos que se juntarmos elementos *numéricos* com elementos do tipo *lógico*, os elementos *lógicos* serão convertidos para o tipo *numérico*, de forma que os valores do tipo *TRUE* serão convertidos para o numeral 1 e os valores do tipo *FALSE* serão convertidos para o numeral 0.

```
mistura3 <- c(100,200,TRUE,FALSE,500,T,F)
mistura3
```

```
## [1] 100 200   1   0 500   1   0
```