Treinamento em Programação no Ambiente R

GENt

May 16, 2019

Dia 1

Este relatório foi feito utilizando R Markdown. Ele pode ser exportado em `.html` ou `.pdf`, basta alterar o `output:` no cabeçalho entre `html_document` e `pdf_document`.

Primeiro, um "oi" para o mundo.

```
cat("Hello world!")
```

```
## Hello world!
```

A função `cat` significa "concatenar" e ela vai imprimir o que eu escrevi no console.

Estabelecendo diretório de trabalho

É a pasta no meu computador que o R "conversa", ou seja, que vai buscar os arquivos de entrada e solta os arquivos de saída. É uma boa prática salvar os scripts, os dados, os gráficos (tudo referente à análise) num mesmo diretório.

```
# Depende do seu computador
# setwd("~/Documents/CursoR")

getwd() # Se eu não souber onde estou
```

```
## [1] "/home/cris/github/GENt-esalq.github.io/cursoR"
```

Operações básicas

O R é uma grande calculadora.

```
1+1.3                  #Decimal definido com "."
```

```
## [1] 2.3
```

```
2*3
```

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

```
2^3
```

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

```
4/2
```

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

```
sqrt(4)               #raíz quadrada
```

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

```r
log(100, base = 10)  #logarítmo na base 10
```

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

```r
log(100)             #logarítmo com base neperiana
```

```
## [1] 4.60517
```

```r
# Resolvendo problema
((13+2+1.5)/3) + log(96, base = 4)
```

```
## [1] 8.792481
```

Lembrando que o que vem antes do parênteses é uma função, e, sendo uma função, existe um manual para ela dentro do R, acesse com:

```r
# Pedindo ajuda sobre função do R
?log
```

Operação com vetores

```r
# Diferentes formas de criar um vetor
c(1,3,2,5,2)
```

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

```r
1:10
```

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

```r
seq(from=0, to=100, by=5)
```

```
##  [1]   0   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80
## [18]  85  90  95 100
```

```r
# ou
seq(0,100,5) # Se você já souber a ordem dos argumentos da função
```

```
##  [1]   0   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80
## [18]  85  90  95 100
```

```r
seq(from=4, to=30, by=3)
```

```
## [1]  4  7 10 13 16 19 22 25 28
```

```r
rep(3:5, 2)
```

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

```r
# Operações
c(1,4,3,2)*2  # Multiplica todos os elementos por 2
```

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

```r
c(4,2,1,5)+c(5,2,6,1) # Soma 4+5, 2+2, 1+6 e assim por diante
```

```
## [1] 9 4 7 6
```

```r
c(4,2,1,5)*c(5,2,6,1) # Multiplica 4*5, 2*2, 1*6 e assim por diante
```

```
## [1] 20  4  6  5
```

Criando objetos

```
x = c(30.1,30.4,40,30.2,30.6,40.1)
# ou
x <- c(30.1,30.4,40,30.2,30.6,40.1)

y = c(0.26,0.3,0.36,0.24,0.27,0.35)
```

Operações com os objetos

```
x*2
```

```
## [1] 60.2 60.8 80.0 60.4 61.2 80.2
```

```
x + y
```

```
## [1] 30.36 30.70 40.36 30.44 30.87 40.45
```

```
x*y
```

```
## [1]  7.826  9.120 14.400  7.248  8.262 14.035
```

```
z <- (x+y)/2
z
```

```
## [1] 15.180 15.350 20.180 15.220 15.435 20.225
```

```
# Aplicando algumas funções
sum(z)  # soma dos valores de z
```

```
## [1] 101.59
```

```
mean(z) # média
```

```
## [1] 16.93167
```

```
var(z)  # variância
```

```
## [1] 6.427507
```

Obtendo valores internos dos objetos por indexação

```
z[3] # elemento na terceira posição do vetor
```

```
## [1] 20.18
```

```
z[2:4]
```

```
## [1] 15.35 20.18 15.22
```

Para saber algumas características do objeto

```
str(z)
```

```
##  num [1:6] 15.2 15.3 20.2 15.2 15.4 ...
```

Vetor de caracteres

```
clone <- c("GRA02", "URO01", "URO03", "GRA02", "GRA01", "URO01")
```

Vetor de fatores (ou variáveis categóricas)

```
clone_fator <- as.factor(clone)
str(clone_fator)
```

```
##  Factor w/ 4 levels "GRA01","GRA02",..: 2 3 4 2 1 3
```

```
levels(clone_fator)
```

```
## [1] "GRA01" "GRA02" "URO01" "URO03"
```

```
length(clone_fator)
```

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

Vetor lógico

```
logico <- x > 40
logico   # Os elementos são maiores que 40?
```

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

```
# Indica a posição dos TRUE
which(logico)  # Obtendo as posiçoes dos elementos TRUE
```

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

```
x[which(logico)] # Obtendo os números maiores que 40 do vetor x pela posição
```

```
## [1] 40.1
```

Para ficar esperto/a

```
(a <- 1:10)
```

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

```
b <- seq(from = 0.1, to = 1, 0.1)
(b <- b*10)
```

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

```
a==b        # Existe um problema computacional de armazenamento
```

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

```
a==round(b) # Evitar que isso aconteceça arredondando o resultado
```

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

```r
?round      # Fiquei com dúvida nessa função

errado <- c(TRUE, "vish", 1) # Não podemos misturar classes num mesmo vetor
errado
```

```
## [1] "TRUE" "vish" "1"
```

Dia 2 (manhã)

Matrizes

```r
X <- matrix(1:12, nrow = 6, ncol = 2)
X
```

```
##      [,1] [,2]
## [1,]    1    7
## [2,]    2    8
## [3,]    3    9
## [4,]    4   10
## [5,]    5   11
## [6,]    6   12
```

```r
W <- matrix(c(x,y), nrow = 6, ncol =2)
W
```

```
##      [,1] [,2]
## [1,] 30.1 0.26
## [2,] 30.4 0.30
## [3,] 40.0 0.36
## [4,] 30.2 0.24
## [5,] 30.6 0.27
## [6,] 40.1 0.35
```

```r
X*2
```

```
##      [,1] [,2]
## [1,]    2   14
## [2,]    4   16
## [3,]    6   18
## [4,]    8   20
## [5,]   10   22
## [6,]   12   24
```

```r
X*X
```

```
##      [,1] [,2]
## [1,]    1   49
## [2,]    4   64
## [3,]    9   81
## [4,]   16  100
## [5,]   25  121
## [6,]   36  144
```

```r
X%*%t(X)          # Multiplicação matricial
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
```

```
## [1,]   50   58   66   74   82   90
## [2,]   58   68   78   88   98  108
## [3,]   66   78   90  102  114  126
## [4,]   74   88  102  116  130  144
## [5,]   82   98  114  130  146  162
## [6,]   90  108  126  144  162  180
```

```
W[4,2] # Número posicionado na linha 4 e coluna 2
```

```
## [1] 0.24
```

```
colnames(W) <- c("altura", "diametro")
rownames(W) <- clone
W
```

```
##       altura diametro
## GRA02   30.1     0.26
## URO01   30.4     0.30
## URO03   40.0     0.36
## GRA02   30.2     0.24
## GRA01   30.6     0.27
## URO01   40.1     0.35
```

Data.frames

Vou escrever isso aqui só para mostrar que podemos criar itens no markdown

- Olha esse
- E esse
- Mais um
- Só para ter certeza

```
campo1 <- data.frame("clone" = clone,      # Antes do sinal de "="
                     "altura" = x,         # estabelecemos os nomes
                     "diametro" = y,       # das colunas
                     "idade" = rep(3:5, 2),
                     "corte"= logico)
campo1
```

```
##   clone altura diametro idade corte
## 1 GRA02   30.1     0.26     3 FALSE
## 2 URO01   30.4     0.30     4 FALSE
## 3 URO03   40.0     0.36     5 FALSE
## 4 GRA02   30.2     0.24     3 FALSE
## 5 GRA01   30.6     0.27     4 FALSE
## 6 URO01   40.1     0.35     5  TRUE
```

```
# Acessando a coluna de idades
campo1$idade
```

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

```
# ou
campo1[,4]
```

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

```r
# Especificando linha e coluna
campo1[1,2]
```

```
## [1] 30.1
```

```r
# Diâmetro do URO03
campo1[3,3]
```

```
## [1] 0.36
```

```r
# Volume
volume <- 3.14*((campo1$diametro/2)^2)*campo1$altura
volume
```

```
## [1] 1.597287 2.147760 4.069440 1.365523 1.751131 3.856116
```

```r
# Adicionando volume ao data.frame campo1
campo1 <- cbind(campo1, volume)
str(campo1)
```

```
## 'data.frame':    6 obs. of  6 variables:
##  $ clone   : Factor w/ 4 levels "GRA01","GRA02",..: 2 3 4 2 1 3
##  $ altura  : num  30.1 30.4 40 30.2 30.6 40.1
##  $ diametro: num  0.26 0.3 0.36 0.24 0.27 0.35
##  $ idade   : int  3 4 5 3 4 5
##  $ corte   : logi  FALSE FALSE FALSE FALSE FALSE TRUE
##  $ volume  : num  1.6 2.15 4.07 1.37 1.75 ...
```

Listas

```r
minha_lista <- list(campo1 = campo1, media_alt = tapply(campo1$altura, campo1$idade, mean), matrix_ex =
str(minha_lista)
```

```
## List of 3
##  $ campo1   :'data.frame':   6 obs. of  6 variables:
##   ..$ clone   : Factor w/ 4 levels "GRA01","GRA02",..: 2 3 4 2 1 3
##   ..$ altura  : num [1:6] 30.1 30.4 40 30.2 30.6 40.1
##   ..$ diametro: num [1:6] 0.26 0.3 0.36 0.24 0.27 0.35
##   ..$ idade   : int [1:6] 3 4 5 3 4 5
##   ..$ corte   : logi [1:6] FALSE FALSE FALSE FALSE FALSE TRUE
##   ..$ volume  : num [1:6] 1.6 2.15 4.07 1.37 1.75 ...
##  $ media_alt: num [1:3(1d)] 30.1 30.5 40
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ : chr [1:3] "3" "4" "5"
##  $ matrix_ex: num [1:6, 1:2] 30.1 30.4 40 30.2 30.6 40.1 0.26 0.3 0.36 0.24 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:6] "GRA02" "URO01" "URO03" "GRA02" ...
##   .. ..$ : chr [1:2] "altura" "diametro"
```

```r
# Acessando conteúdo das listas
minha_lista[[1]]
```

```
##   clone altura diametro idade corte   volume
## 1 GRA02   30.1     0.26     3 FALSE 1.597287
## 2 URO01   30.4     0.30     4 FALSE 2.147760
## 3 URO03   40.0     0.36     5 FALSE 4.069440
```

```
## 4 GRA02   30.2     0.24     3 FALSE 1.365523
## 5 GRA01   30.6     0.27     4 FALSE 1.751131
## 6 URO01   40.1     0.35     5  TRUE 3.856116
```

```
# ou
minha_lista$campo1
```

```
##   clone altura diametro idade corte   volume
## 1 GRA02   30.1     0.26     3 FALSE 1.597287
## 2 URO01   30.4     0.30     4 FALSE 2.147760
## 3 URO03   40.0     0.36     5 FALSE 4.069440
## 4 GRA02   30.2     0.24     3 FALSE 1.365523
## 5 GRA01   30.6     0.27     4 FALSE 1.751131
## 6 URO01   40.1     0.35     5  TRUE 3.856116
```

```
# Arrays
(meu_array <- array(1:24, dim = c(2,3,4)))
```

```
## , , 1
##
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
##
## , , 2
##
##      [,1] [,2] [,3]
## [1,]    7    9   11
## [2,]    8   10   12
##
## , , 3
##
##      [,1] [,2] [,3]
## [1,]   13   15   17
## [2,]   14   16   18
##
## , , 4
##
##      [,1] [,2] [,3]
## [1,]   19   21   23
## [2,]   20   22   24
```

Exportando e importando dados

```
# Salvo meu objeto campo1
save(campo1, file = "campo1.RData")

# Removo o objeto
rm(campo1)  # Certifique-se que salvou o objeto antes de removê-lo

# Chamo ele de novo
load("campo1.RData")

#save.image() #salva um .RData no meu diretório de trabalho
```

```r
load("campo1.RData")

# Exportando em outros formatos
write.table(campo1, file = "campo1.txt", sep = ";", dec = ".", row.names = FALSE)
write.csv(campo1, file = "campo1.csv", row.names = TRUE)
```

Importando tabelas

```r
campo1_txt <- read.table(file = "campo1.txt", sep=";", dec=".", header = TRUE)
campo1_csv <- read.csv(file = "campo1.csv")
head(campo1_txt)
```

```
##   clone altura diametro idade corte   volume
## 1 GRA02   30.1     0.26     3 FALSE 1.597287
## 2 URO01   30.4     0.30     4 FALSE 2.147760
## 3 URO03   40.0     0.36     5 FALSE 4.069440
## 4 GRA02   30.2     0.24     3 FALSE 1.365523
## 5 GRA01   30.6     0.27     4 FALSE 1.751131
## 6 URO01   40.1     0.35     5  TRUE 3.856116
```

```r
head(campo1_csv)
```

```
##   X clone altura diametro idade corte   volume
## 1 1 GRA02   30.1     0.26     3 FALSE 1.597287
## 2 2 URO01   30.4     0.30     4 FALSE 2.147760
## 3 3 URO03   40.0     0.36     5 FALSE 4.069440
## 4 4 GRA02   30.2     0.24     3 FALSE 1.365523
## 5 5 GRA01   30.6     0.27     4 FALSE 1.751131
## 6 6 URO01   40.1     0.35     5  TRUE 3.856116
```

```r
# Importando a tabela com os dados do formulário
## Se for sistema linux/mac
dados <- read.csv(file = "dados_alunos.csv", stringsAsFactors = FALSE, na.strings="-")
# ou
load("dados_alunos.RData")

## Se for sistema windows
#dados <- read.csv(file = "dados_alunos.csv", stringsAsFactors = FALSE, na.strings="-", fileEncoding =

# Verificando que esta tudo certo
str(dados)
```

```
## 'data.frame':    35 obs. of  12 variables:
##  $ Timestamp                                                          : chr  "5/16/2019
##  $ Idade..Ex..26.                                                     : int  38 27 26 1
##  $ Dia.e.mês.do.aniversário..Ex..05.10.                               : chr  "02/01" "1
##  $ Gênero                                                             : chr  "Masculino
##  $ Cidade.de.Origem..Ex..Piracicaba.SP.                               : chr  "Piracicab
##  $ Altura.em.metros..Ex..1.60.                                        : num  1.82 1.5 1
##  $ Peso.em.Kg..Ex..56.                                                : int  91 58 56 5
##  $ Área.com.a.qual.mais.se.identifica                                 : chr  "Biológica
##  $ Dê.uma.nota.de.0.a.10.para.seu.nível.de.conhecimento.em.R           : int  3 2 1 2 4
##  $ Você.utiliza.alguma.outra.linguagem.de.programação..Qual.is...Ex..C..C....python.: chr  "Não" "Não
```

```
##  $ Para.que.você.utiliza.ou.utilizará.o.R.                                    : chr  "programa
##  $ Qual.a.sua.motivação.para.fazer.este.curso...texto.de.30.a.100.palavras.   : chr  "material
```

```
# também
dim(dados)
```

```
## [1] 35 12
```

Alterando nome das colunas

```
colnames(dados) <- c("Data_pesq", "Idade", "Niver", "Genero", "Cidade",
                     "Altura","Peso", "Area", "ConhecimentoR", "Outras_linguagens",
                     "Utilizacao", "Motivacao")
str(dados)
```

```
## 'data.frame':    35 obs. of  12 variables:
##  $ Data_pesq        : chr  "5/16/2019 11:19:37" "5/16/2019 11:30:40" "5/16/2019 14:21:20" "5/17/2019
##  $ Idade            : int  38 27 26 24 25 34 45 22 31 21 ...
##  $ Niver            : chr  "02/01" "28/05" "21/04" "07/02" ...
##  $ Genero           : chr  "Masculino" "Feminino" "Feminino" "Feminino" ...
##  $ Cidade           : chr  "Piracicaba-SP" "Guaxupé - MG" "São José dos Campos" "Alta Floresta - MT"
##  $ Altura           : num  1.82 1.5 1.56 1.64 1.7 1.64 1.88 1.81 1.73 1.63 ...
##  $ Peso             : int  91 58 56 58 54 56 93 85 75 58 ...
##  $ Area             : chr  "Biológicas" "Biológicas" "Interdiciplinar" "Interdiciplinar" ...
##  $ ConhecimentoR    : int  3 2 1 2 4 2 0 2 3 1 ...
##  $ Outras_linguagens: chr  "Não" "Não" "Não" "Não" ...
##  $ Utilizacao       : chr  "programa livre" "Tese de Doutorado" "Análise de dados de pesquisa" "Anál
##  $ Motivacao        : chr  "material mais robusto para análise de dados" "Importância de aprendizado
```

Paradoxo do aniversário

```
table(dados$Niver)
```

```
##
##      02/01      02/09      06/04      06/12      07/02      08/04
##          1          1          1          1          1          1
##      08/06      08/10      09/04      09/05      10/10      10/11
##          1          1          1          1          1          1
##      11/04      12/73 16/10/1966      17/01      17/06      17/08
##          1          1          1          1          1          1
##      18/12      19/06      20/05      21/04      22/08      23/05
##          1          1          1          1          1          1
##     23/11/      25/08      25/09 25/09/1993 26/05/1989      27/03
##          1          1          1          1          1          1
##      28/05      30/05      30/11      31/08      31/12
##          1          1          1          1          1
```

Estruturas condicionais

```
## If e else
```

```r
if(2 >3){
  print("dois é maior que três")
} else {
  print("dois não é maior que três")
}
```

```
## [1] "dois não é maior que três"
```

```r
if(dados[3,9] == 0){
  print("Nunca é tarde para começar!")
} else {
  print("Já pegou o embalo, agora é só continuar!")
}
```

```
## [1] "Já pegou o embalo, agora é só continuar!"
```

```r
if(dados[7,9] == 0){
  print("Nunca é tarde para começar!")
} else if (dados[3,9] > 0 && dados[3,9] < 5){
  print("Já pegou o embalo, agora é só continuar!")
} else {
  print("Nos avise se estivermos falando algo errado...hehe")
}
```

```
## [1] "Nunca é tarde para começar!"
```

```r
## Switch

switch(dados[5,8],
       Exatas = print("Será que aprendeu alguma linhagem de programação na graduação?"),
       Interdiciplinar = print("Em que foi a gradução?"),
       print("Ta aqui colocando o pezinho na exatas")
)
```

```
## [1] "Ta aqui colocando o pezinho na exatas"
```

Estruturas de repetição

```r
## For

for(i in 1:10){
  print(i)
}
```

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

```r
test <- vector()
for(i in 1:10){
  test[i] <- i+4
}
test
```

```
##  [1]  5  6  7  8  9 10 11 12 13 14
```

```r
for(i in 1:nrow(dados)){
  if(dados[i,9] == 0){
    print("Nunca é tarde para começar!")
  } else if (dados[i,9] > 0 && dados[i,9] < 5){
    print("Já pegou o embalo, agora é só continuar!")
  } else {
    print("Nos avise se estivermos falando algo errado...hehe")
  }
}
```

```
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Nunca é tarde para começar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Nos avise se estivermos falando algo errado...hehe"
## [1] "Nunca é tarde para começar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Nos avise se estivermos falando algo errado...hehe"
## [1] "Nunca é tarde para começar!"
## [1] "Nos avise se estivermos falando algo errado...hehe"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Nunca é tarde para começar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Nunca é tarde para começar!"
## [1] "Nunca é tarde para começar!"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Nos avise se estivermos falando algo errado...hehe"
## [1] "Já pegou o embalo, agora é só continuar!"
## [1] "Já pegou o embalo, agora é só continuar!"
```

```r
# Exemplo do uso da função grepl
grepl("-", dados[1,5]) # A primeira linha contem o caracter "-"
```

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

```r
for(i in 1:nrow(dados)){
  if(grepl("-", dados[i,5])){
    cat("Esse/a seguiu o exemplo direitinho. Parabéns!\n")
  } else {
    cat("Precisamos adicionar mais informações na linha", i, "\n")
  }
}
```

```
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 3
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 7
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 10
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 13
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 15
## Precisamos adicionar mais informações na linha 16
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 20
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 24
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 27
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 30
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
```

```r
corrigir <- vector()
for(i in 1:nrow(dados)){
  if(grepl("-", dados[i,5])){
    cat("Esse/a seguiu o exemplo direitinho. Parabéns!\n")
  } else {
    cat("Precisamos adicionar mais informações na linha", i, "\n")
```

```
    corrigir <- c(corrigir, i)
  }
}
```

```
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 3
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 7
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 10
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 13
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 15
## Precisamos adicionar mais informações na linha 16
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 20
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 24
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 27
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Precisamos adicionar mais informações na linha 30
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
## Esse/a seguiu o exemplo direitinho. Parabéns!
```

Dia 2 (tarde)

Possibilidades de resposta para os exercícios

```
dados[corrigir,5]
```

```
##  [1] "São José dos Campos" "Piracicaba"          "São Paulo"
##  [4] "Piracicaba"          "Uberlandia"          "Piracicaba"
##  [7] "Uberaba"             "São Luis"            "Piracicaba"
## [10] "Piracicaba"
```

```
novo <- c("São José dos Campos - SP", "Piracicaba - SP", "São Paulo - SP", "Piracicaba-SP",
          "Uberlância-MG", "Piracicaba-SP", "Uberaba-MG", "São Luis-MA", "Piracicaba-SP",
          "Piracicaba-SP")
dados[corrigir,5] <- novo

# Verificando se corrigiu
dados[,5]
```

```
##  [1] "Piracicaba-SP"                  "Guaxupé - MG"
##  [3] "São José dos Campos - SP"       "Alta Floresta - MT"
##  [5] "Goiania-Go"                     "Guaíra-SP"
##  [7] "Piracicaba - SP"                "Santos-SP"
##  [9] "Cuiabá-MT"                      "São Paulo - SP"
## [11] "Toledo-PR"                      "Sao Joao Del Rei- Minas Gerais"
## [13] "Piracicaba-SP"                  "Brasília - DF"
## [15] "Uberlância-MG"                  "Piracicaba-SP"
## [17] "Brasília - DF"                  "Jatai-GO"
## [19] "Mogi das Cruzes-SP"             "Uberaba-MG"
## [21] "Belo Jardin-PE"                 "Ji-Paraná/RO"
## [23] "Pará de Minas - MG"             "São Luis-MA"
## [25] "Afogados da Ingazeira-PE"       "Campinas-SP"
## [27] "Piracicaba-SP"                  "Goiatuba-GO"
## [29] "Americana-SP"                   "Piracicaba-SP"
## [31] "Piracicaba-SP"                  "Itajubá-MG"
## [33] "Nova Monte Verde - MT"          "São Paulo - SP"
## [35] "São Paulo - SP"
```

```
decada <- 2019 - dados$Idade

for(i in 1:length(decada)){
  if(decada[i] > 1960 && decada[i] < 1970){
    print("Nasceu na década de 60")
  } else if(decada[i] >= 1970 && decada[i] < 1980){
    print("Nasceu na década de 70")
  } else if(decada[i] >= 1980 && decada[i] < 1990){
    print("Nasceu na década de 80")
  } else if(decada[i] >= 1990 && decada[i] < 2000){
    print("Nasceu na década de 90")
  } else {
    print("Xóvem")
  }
}
```

```
## [1] "Nasceu na década de 80"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 80"
## [1] "Nasceu na década de 70"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 80"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 80"
```

```
## [1] "Nasceu na década de 80"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 80"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 80"
## [1] "Nasceu na década de 60"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 80"
## [1] "Nasceu na década de 80"
## [1] "Nasceu na década de 60"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
## [1] "Nasceu na década de 90"
```

Outra estrutura de repetição: while

```
x <- 1

while(x < 5){
  x <- x + 1
  print(x)
}
```

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

O markdown entende linguagem html também. Por isso eu posso colorir assim:

Cuidado!!! Não rode o código abaixo!

```
## loop infinito (não rodar!)
x <- 1

while(x < 5){
   x + 1
   print(x)
}
```

Repare que se eu usar o eval=FALSE, significa que o código desse chunck não irá ser avaliado quando eu gerar o relatório.

Uso do break e do next

```
x <- 1

while(x < 5){
  x <- x + 1
  if(x==4) break
  print(x)
}
```

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

```
x <- 1

while(x < 5){
  x <- x + 1
  if(x==4) next
  print(x)
}
```

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

Outra estrutura de repetição: repeat

```
x <- 1
repeat{
  x <- x+1
  print(x)
  if(x==4) break
}
```

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

Loops dentro de loops

```
# Criando uma matrix vazia
ex_mat <- matrix(nrow=10, ncol=10)

# cada número dentro da matrix será o produto no índice da coluna pelo índice da linha
for(i in 1:dim(ex_mat)[1]) {
  for(j in 1:dim(ex_mat)[2]) {
    ex_mat[i,j] = i*j
  }
}
```

Elaboração de funções

```
## Gerando função quadra (para elevar ao quadrado)

quadra <- function(x){
  z <- x*x
  return(z)
}

quadra(3)
```

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

```
quadra(4)
```

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

```
qualquer_nome <- 4
quadra(qualquer_nome)
```

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

```
## Agora, uma função com mais sentido
```

Primeiro, como seria se nao fosse uma função

```
## Calcula o índice de massa corporal (IMC) dos participantes
IMC <- dados$Peso/quadra(dados$Altura)

## Calcula a média das idade dos participantes
id_med <- mean(dados$Idade)

## Calcula a mediana das idades dos participantes
id_median <- median(dados$Idade)

## Calcula a porgentagem de mulheres entre os participantes
mul <- 100*(length(which(dados$Genero == "Feminino"))/length(dados$Genero))

## Faz uma lista com todos os resultados
final_list <- list(IMC=IMC, idade_media = id_med,
                   idade_mediana = id_median, porcentagem_mulheres = mul)
```

Agora na versão função

```
minha_funcao <- function(df.entrada){
  ## Calcula o índice de massa corporal (IMC) dos participantes
  IMC <- df.entrada$Peso/quadra(df.entrada$Altura)

  ## Calcula a média das idade dos participantes
  id_med <- mean(df.entrada$Idade)

  ## Calcula a mediana das idades dos participantes
  id_median <- median(df.entrada$Idade)

  ## Calcula a porgentagem de mulheres entre os participantes
  mul <- 100*(length(which(df.entrada$Genero == "Feminino"))/length(df.entrada$Genero))
```

```
  ## Faz uma lista com todos os resultados
  final_list <- list(IMC=IMC, idade_media = id_med,
                     idade_mediana = id_median, porcentagem_mulheres = mul)
  return(final_list)

}

# Rodando
test_list <- minha_funcao(df.entrada = dados)
test_list
```

```
## $IMC
##  [1] 27.47253 25.77778 23.01118 21.56454 18.68512 20.82094 26.31281
##  [8] 25.94548 25.05931 21.82995 25.95156 26.60971 22.34352 23.19109
## [15] 21.51386 20.61313 22.20408 25.71166 28.69265 21.04805 24.38237
## [22] 25.55885 24.52435 24.21229 24.80159 21.35780 22.58955 17.54187
## [29] 21.48437 20.93664 19.78997 20.44444 21.71807 29.98359 22.03857
##
## $idade_media
## [1] 29.85714
##
## $idade_mediana
## [1] 27
##
## $porcentagem_mulheres
## [1] 60
```

Posso colocar uns avisos:

```
minha_funcao <- function(df.entrada){

  if (length(grep("Altura", colnames(df.entrada))) == 0 ||
      length(grep("Peso", colnames(df.entrada))) == 0 ||
      length(grep("Idade", colnames(df.entrada))) == 0 ||
      length(grep("Genero", colnames(df.entrada))) == 0)
    stop("Esta faltando alguma das informações.")


  ## Calcula o índice de massa corporal (IMC) dos participantes
  IMC <- df.entrada$Peso/quadra(df.entrada$Altura)

  ## Calcula a média das idade dos participantes
  id_med <- mean(df.entrada$Idade)

  ## Calcula a mediana das idades dos participantes
  id_median <- median(df.entrada$Idade)

  ## Calcula a porgentagem de mulheres entre os participantes
  mul <- 100*(length(which(df.entrada$Genero == "Feminino"))/length(df.entrada$Genero))

  ## Faz uma lista com todos os resultados
  final_list <- list(IMC=IMC, idade_media = id_med,
                     idade_mediana = id_median, porcentagem_mulheres = mul)
  return(final_list)
```

```r
}

test_list <- minha_funcao(df.entrada = dados)

dados1 <- dados[,-2] # Removendo coluna de idade

# Rodando isso vai dar o erro que eu criei
# test_list <- minha_funcao(df.entrada = dados1)
```