

# Manipulação e Apresentação de Dados

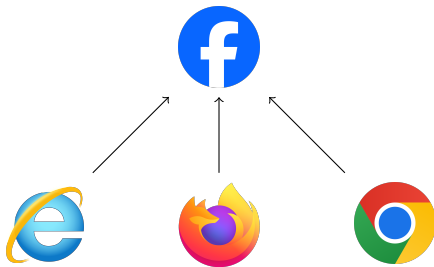
Guilherme Bovi Ambrosano

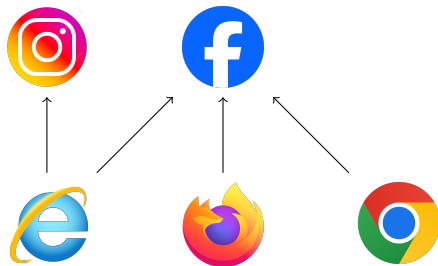


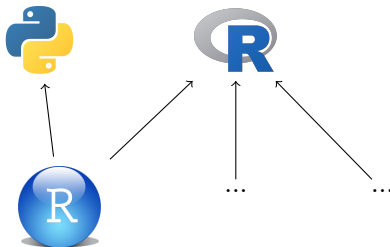
- ▶ Introdução
- ▶ R básico
- ▶ Classes (ou tipos) dos objetos
- ▶ Estruturas de dados
- ▶ Exercícios
- ▶ A família apply
- ▶ Pacotes stringr e forcats
- ▶ Lendo arquivos externos
- ▶ Exercícios

# Introdução











- ▶ R
  - ▶ <https://www.r-project.org/>
- ▶ RStudio
  - ▶ <https://posit.co/download/rstudio-desktop/>
- ▶ Pacotes do R

```
install.packages(c("tidyverse",  
                  "readxl", "lubridate", "broom"))
```





- ▶ forcats: <https://forcats.tidyverse.org/>
- ▶ stringr: <https://stringr.tidyverse.org/>
- ▶ lubridate: <https://lubridate.tidyverse.org/>
- ▶ tibble: <https://tibble.tidyverse.org/>
- ▶ readr: <https://readr.tidyverse.org/>
- ▶ readxl: <https://readxl.tidyverse.org/>
- ▶ dplyr: <https://dplyr.tidyverse.org/>
- ▶ tidyr: <https://tidyr.tidyverse.org/>
- ▶ purrr: <https://purrr.tidyverse.org/>
- ▶ ggplot2: <https://ggplot2.tidyverse.org/>

# R básico

# Interface do RStudio



The screenshot displays the RStudio interface with four main panes:

- Source:** Contains the R script `ggplot2.R` with the following code:

```
1 library(ggplot2)
2 mpg_plot <- ggplot(mpg, aes(x = displ, y = hwy)) +
3   geom_point(aes(colour = class))
4
5 mpg_plot
6
```
- Environment:** Shows the Global Environment with a table of objects:

Name	Type	Len...	Size	Value
mpg_plot	gg	9	29.1	List of 9
- Console:** Shows the execution of the code from the Source pane:

```
> library(ggplot2)
> mpg_plot <- ggplot(mpg, aes(x = displ, y = hwy)) +
+   geom_point(aes(colour = class))
>
> mpg_plot
>
```
- Plots:** Displays a scatter plot of `hwy` (y-axis) versus `displ` (x-axis). The points are colored by the `class` variable. A legend on the right lists the classes: 2seater, compact, midsize, minivan, pickup, subcompact, and suv.

<https://docs.posit.co/ide/user/ide/guide/ui/ui-panes.html>

# Classes (ou tipos) dos objetos

numeric, character e factor



## Tipos de objetos no R

Objetos de classe numeric

```
# Numérico
```

```
nFolhas1 <- 15 # número de folhas no ramo 1
```

```
nFolhas <- c(nFolhas1, 20, 25, 17, 26, 19) # ramos 1 a 6
```

```
length(nFolhas)
```

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

```
nFolhas[3]
```

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



## Tipos de objetos no R

Objetos de classe numeric

```
# Numérico
```

```
nFolhas1 <- 15 # número de folhas no ramo 1
```

```
nFolhas <- c(nFolhas1, 20, 25, 17, 26, 19) # ramos 1 a 6
```

```
length(nFolhas)
```

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

```
nFolhas[3]
```

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



## Tipos de objetos no R

Objetos de classe numeric

```
# Numérico
```

```
nFolhas1 <- 15 # número de folhas no ramo 1
```

```
nFolhas <- c(nFolhas1, 20, 25, 17, 26, 19) # ramos 1 a 6
```

```
length(nFolhas)
```

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

```
nFolhas[3]
```

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



## Tipos de objetos no R

Objetos de classe numeric

```
# Numérico
```

```
nFolhas1 <- 15 # número de folhas no ramo 1
```

```
nFolhas <- c(nFolhas1, 20, 25, 17, 26, 19) # ramos 1 a 6
```

```
length(nFolhas)
```

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

```
nFolhas[3]
```

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





## Tipos de objetos no R

Objetos do classe character

```
# Caractere  
cidade <- "Piracicaba"  
nchar(cidade)
```

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

```
nomes <- c("Alice", "Bob", "Charlie", "David")  
length(nomes)
```

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

```
nomes[3]
```

```
## [1] "Charlie"
```



## Tipos de objetos no R

Objetos do classe character

```
# Caractere  
cidade <- "Piracicaba"  
nchar(cidade)
```

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

```
nomes <- c("Alice", "Bob", "Charlie", "David")  
length(nomes)
```

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

```
nomes[3]
```

```
## [1] "Charlie"
```



```
grep("e", nomes, value=T)
```

```
## [1] "Alice"    "Charlie"
```

```
substr(cidade, 1, 4)
```

```
## [1] "Pira"
```

```
gsub("cicaba", "ssununga", cidade)
```

```
## [1] "Pirassununga"
```

```
paste("Olá, meu nome é", nomes)
```

```
## [1] "Olá, meu nome é Alice"    "Olá, meu nome é Bob"
```

```
## [3] "Olá, meu nome é Charlie" "Olá, meu nome é David"
```



```
grep("e", nomes, value=T)
```

```
## [1] "Alice"    "Charlie"
```

```
substr(cidade, 1, 4)
```

```
## [1] "Pira"
```

```
gsub("cicaba", "ssununga", cidade)
```

```
## [1] "Pirassununga"
```

```
paste("Olá, meu nome é", nomes)
```

```
## [1] "Olá, meu nome é Alice"    "Olá, meu nome é Bob"
```

```
## [3] "Olá, meu nome é Charlie" "Olá, meu nome é David"
```



```
grep("e", nomes, value=T)
```

```
## [1] "Alice"    "Charlie"
```

```
substr(cidade, 1, 4)
```

```
## [1] "Pira"
```

```
gsub("cicaba", "ssununga", cidade)
```

```
## [1] "Pirassununga"
```

```
paste("Olá, meu nome é", nomes)
```

```
## [1] "Olá, meu nome é Alice"    "Olá, meu nome é Bob"
```

```
## [3] "Olá, meu nome é Charlie" "Olá, meu nome é David"
```



```
grep("e", nomes, value=T)
```

```
## [1] "Alice"    "Charlie"
```

```
substr(cidade, 1, 4)
```

```
## [1] "Pira"
```

```
gsub("cicaba", "ssununga", cidade)
```

```
## [1] "Pirassununga"
```

```
paste("Olá, meu nome é", nomes)
```

```
## [1] "Olá, meu nome é Alice"    "Olá, meu nome é Bob"
```

```
## [3] "Olá, meu nome é Charlie" "Olá, meu nome é David"
```



```
textoComprido <- "Um texto muito comprido, que pode ser quebrado em várias  
linhas <- strwrap(textoComprido, width=10)  
linhas
```

```
## [1] "Um texto" "muito" "comprido," "que pode" "ser" "que  
## [7] "em várias" "linhas"
```

```
paste(nomes, collapse=", ")
```

```
## [1] "Alice, Bob, Charlie, David"
```

```
paste(linhas, collapse=" ")
```

```
## [1] "Um texto muito comprido, que pode ser quebrado em várias linhas"
```

```
cat(paste(linhas, collapse="\n"))
```

```
## Um texto  
## muito  
## comprido,  
## que pode  
## ser  
## quebrado  
## em várias
```



```
textoComprido <- "Um texto muito comprido, que pode ser quebrado em várias  
linhas <- strwrap(textoComprido, width=10)  
linhas
```

```
## [1] "Um texto" "muito" "comprido," "que pode" "ser" "que  
## [7] "em várias" "linhas"
```

```
paste(nomes, collapse=", ")
```

```
## [1] "Alice, Bob, Charlie, David"
```

```
paste(linhas, collapse=" ")
```

```
## [1] "Um texto muito comprido, que pode ser quebrado em várias linhas"
```

```
cat(paste(linhas, collapse="\n"))
```

```
## Um texto  
## muito  
## comprido,  
## que pode  
## ser  
## quebrado  
## em várias
```





```
textoComprido <- "Um texto muito comprido, que pode ser quebrado em várias  
linhas <- strwrap(textoComprido, width=10)  
linhas
```

```
## [1] "Um texto" "muito" "comprido," "que pode" "ser" "que  
## [7] "em várias" "linhas"
```

```
paste(nomes, collapse=", ")
```

```
## [1] "Alice, Bob, Charlie, David"
```

```
paste(linhas, collapse=" ")
```

```
## [1] "Um texto muito comprido, que pode ser quebrado em várias linhas"
```

```
cat(paste(linhas, collapse="\n"))
```

```
## Um texto  
## muito  
## comprido,  
## que pode  
## ser  
## quebrado  
## em várias
```



```
textoComprido <- "Um texto muito comprido, que pode ser quebrado em várias  
linhas <- strwrap(textoComprido, width=10)  
linhas
```

```
## [1] "Um texto" "muito" "comprido," "que pode" "ser" "quebrado em várias"  
## [7] "em várias" "linhas"
```

```
paste(nomes, collapse=", ")
```

```
## [1] "Alice, Bob, Charlie, David"
```

```
paste(linhas, collapse=" ")
```

```
## [1] "Um texto muito comprido, que pode ser quebrado em várias linhas"
```

```
cat(paste(linhas, collapse="\n"))
```

```
## Um texto  
## muito  
## comprido,  
## que pode  
## ser  
## quebrado  
## em várias
```



## Tipos de objetos no R

Objetos de classe factor

```
# Fator
tratamentos <- factor(c("Trat 1", "Trat 1", "Trat 2",
                        "Trat 2", "Trat 3", "Trat 3"))
blocos <- factor(c(1, 2, 1, 2, 1, 2))
```

nomes

```
## [1] "Alice" "Bob" "Charlie" "David"
```

tratamentos

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3
## Levels: Trat 1 Trat 2 Trat 3
```



## Tipos de objetos no R

Objetos de classe factor

```
# Fator
tratamentos <- factor(c("Trat 1", "Trat 1", "Trat 2",
                        "Trat 2", "Trat 3", "Trat 3"))
blocos <- factor(c(1, 2, 1, 2, 1, 2))
```

nomes

```
## [1] "Alice" "Bob" "Charlie" "David"
```

tratamentos

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3
## Levels: Trat 1 Trat 2 Trat 3
```



```
tratamentos
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 1 Trat 2 Trat 3
```

```
levels(tratamentos)
```

```
## [1] "Trat 1" "Trat 2" "Trat 3"
```

```
tratamentos <- relevel(tratamentos, 2)  
tratamentos
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 2 Trat 1 Trat 3
```



```
tratamentos
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 1 Trat 2 Trat 3
```

```
levels(tratamentos)
```

```
## [1] "Trat 1" "Trat 2" "Trat 3"
```

```
tratamentos <- relevel(tratamentos, 2)  
tratamentos
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 2 Trat 1 Trat 3
```



```
ola <- function(x) paste("Olá,", x)  
ola("mundo!")
```

```
## [1] "Olá, mundo!"
```

```
ola(nomes)
```

```
## [1] "Olá, Alice" "Olá, Bob" "Olá, Charlie" "Olá, David"
```

```
divisao <- function(a, b) a/b  
divisao(1, 2)
```

```
## [1] 0.5
```



```
ola <- function(x) paste("Olá,", x)  
ola("mundo!")
```

```
## [1] "Olá, mundo!"
```

```
ola(nomes)
```

```
## [1] "Olá, Alice" "Olá, Bob" "Olá, Charlie" "Olá, David"
```

```
divisao <- function(a, b) a/b  
divisao(1, 2)
```

```
## [1] 0.5
```





```
funcao <- function(argumento1, argumento2,  
                    argumento3) {  
  cat(paste0("0 primeiro argumento é ", argumento1, ".\n",  
            "0 segundo argumento é ", argumento2, ".\n",  
            "0 terceiro argumento é ", argumento3, "."))  
}
```



```
funcao(1, 2, 3)
```

```
## 0 primeiro argumento é 1.
```

```
## 0 segundo argumento é 2.
```

```
## 0 terceiro argumento é 3.
```

```
funcao(argumento2=1, argumento3=2, 3)
```

```
## 0 primeiro argumento é 3.
```

```
## 0 segundo argumento é 1.
```

```
## 0 terceiro argumento é 2.
```



```
funcao(argumento3=1, 2, 3)
```

```
## O primeiro argumento é 2.  
## O segundo argumento é 3.  
## O terceiro argumento é 1.
```

```
funcao(1,2,argumento1=3)
```

```
## O primeiro argumento é 3.  
## O segundo argumento é 1.  
## O terceiro argumento é 2.
```

# Estruturas de dados

vetores, matrizes, data-frames e listas



## Criando vetores

```
paste("Tratamento", seq(1,4))
```

```
## [1] "Tratamento 1" "Tratamento 2" "Tratamento 3" "Tratamento 4"  
seq(0, 1, length.out=5)
```

```
## [1] 0.00 0.25 0.50 0.75 1.00  
seq(1, by=2, length.out=10)
```

```
## [1] 1 3 5 7 9 11 13 15 17 19  
rep(seq(1,2), times=5)
```

```
## [1] 1 2 1 2 1 2 1 2 1 2  
rep(seq(4,1), each=5)
```

```
## [1] 4 4 4 4 4 3 3 3 3 3 2 2 2 2 2 1 1 1 1 1  
rep(seq(3,15,3), times=3, each=2)
```

```
## [1] 3 3 6 6 9 9 12 12 15 15 3 3 6 6 9 9 12 12 15 15 3  
## [26] 9 12 12 15 15
```



## Criando vetores

```
paste("Tratamento", seq(1,4))
```

```
## [1] "Tratamento 1" "Tratamento 2" "Tratamento 3" "Tratamento 4"  
seq(0, 1, length.out=5)
```

```
## [1] 0.00 0.25 0.50 0.75 1.00  
seq(1, by=2, length.out=10)
```

```
## [1] 1 3 5 7 9 11 13 15 17 19  
rep(seq(1,2), times=5)
```

```
## [1] 1 2 1 2 1 2 1 2 1 2  
rep(seq(4,1), each=5)
```

```
## [1] 4 4 4 4 4 3 3 3 3 3 2 2 2 2 2 1 1 1 1 1  
rep(seq(3,15,3), times=3, each=2)
```

```
## [1] 3 3 6 6 9 9 12 12 15 15 3 3 6 6 9 9 12 12 15 15 3  
## [26] 9 12 12 15 15
```



## Criando vetores

```
paste("Tratamento", seq(1,4))
```

```
## [1] "Tratamento 1" "Tratamento 2" "Tratamento 3" "Tratamento 4"  
seq(0, 1, length.out=5)
```

```
## [1] 0.00 0.25 0.50 0.75 1.00  
seq(1, by=2, length.out=10)
```

```
## [1] 1 3 5 7 9 11 13 15 17 19  
rep(seq(1,2), times=5)
```

```
## [1] 1 2 1 2 1 2 1 2 1 2  
rep(seq(4,1), each=5)
```

```
## [1] 4 4 4 4 4 3 3 3 3 3 2 2 2 2 2 1 1 1 1 1  
rep(seq(3,15,3), times=3, each=2)
```

```
## [1] 3 3 6 6 9 9 12 12 15 15 3 3 6 6 9 9 12 12 15 15 3  
## [26] 9 12 12 15 15
```



## Criando vetores

```
paste("Tratamento", seq(1,4))
```

```
## [1] "Tratamento 1" "Tratamento 2" "Tratamento 3" "Tratamento 4"  
seq(0, 1, length.out=5)
```

```
## [1] 0.00 0.25 0.50 0.75 1.00  
seq(1, by=2, length.out=10)
```

```
## [1] 1 3 5 7 9 11 13 15 17 19  
rep(seq(1,2), times=5)
```

```
## [1] 1 2 1 2 1 2 1 2 1 2  
rep(seq(4,1), each=5)
```

```
## [1] 4 4 4 4 4 3 3 3 3 3 2 2 2 2 2 1 1 1 1 1  
rep(seq(3,15,3), times=3, each=2)
```

```
## [1] 3 3 6 6 9 9 12 12 15 15 3 3 6 6 9 9 12 12 15 15 3  
## [26] 9 12 12 15 15
```





## Criando vetores

```
paste("Tratamento", seq(1,4))
```

```
## [1] "Tratamento 1" "Tratamento 2" "Tratamento 3" "Tratamento 4"  
seq(0, 1, length.out=5)
```

```
## [1] 0.00 0.25 0.50 0.75 1.00  
seq(1, by=2, length.out=10)
```

```
## [1] 1 3 5 7 9 11 13 15 17 19  
rep(seq(1,2), times=5)
```

```
## [1] 1 2 1 2 1 2 1 2 1 2  
rep(seq(4,1), each=5)
```

```
## [1] 4 4 4 4 4 3 3 3 3 3 2 2 2 2 2 1 1 1 1 1  
rep(seq(3,15,3), times=3, each=2)
```

```
## [1] 3 3 6 6 9 9 12 12 15 15 3 3 6 6 9 9 12 12 15 15 3  
## [26] 9 12 12 15 15
```



## Criando vetores

```
paste("Tratamento", seq(1,4))
```

```
## [1] "Tratamento 1" "Tratamento 2" "Tratamento 3" "Tratamento 4"  
seq(0, 1, length.out=5)
```

```
## [1] 0.00 0.25 0.50 0.75 1.00  
seq(1, by=2, length.out=10)
```

```
## [1] 1 3 5 7 9 11 13 15 17 19  
rep(seq(1,2), times=5)
```

```
## [1] 1 2 1 2 1 2 1 2 1 2  
rep(seq(4,1), each=5)
```

```
## [1] 4 4 4 4 4 3 3 3 3 3 2 2 2 2 2 1 1 1 1 1  
rep(seq(3,15,3), times=3, each=2)
```

```
## [1] 3 3 6 6 9 9 12 12 15 15 3 3 6 6 9 9 12 12 15 15 3  
## [26] 9 12 12 15 15
```



```
altura <- c(1.65, 1.67, 1.70, 1.58, 1.69, 1.73, 1.60, 1.70)
```

```
cut(altura, seq(1.55, by=.05, length.out=5))
```

```
## [1] (1.6,1.65] (1.65,1.7] (1.65,1.7] (1.55,1.6] (1.65,1.7] (1.7,1.75]
```

```
## [8] (1.65,1.7]
```

```
## Levels: (1.55,1.6] (1.6,1.65] (1.65,1.7] (1.7,1.75]
```

```
factor(ifelse(altura<median(altura), "Pequeno", "Alto"),  
       levels=c("Pequeno", "Alto"))
```

```
## [1] Pequeno Pequeno Alto Pequeno Alto Alto Pequeno A
```

```
## Levels: Pequeno Alto
```



```
altura <- c(1.65, 1.67, 1.70, 1.58, 1.69, 1.73, 1.60, 1.70)
```

```
cut(altura, seq(1.55, by=.05, length.out=5))
```

```
## [1] (1.6,1.65] (1.65,1.7] (1.65,1.7] (1.55,1.6] (1.65,1.7] (1.7,1.75]
```

```
## [8] (1.65,1.7]
```

```
## Levels: (1.55,1.6] (1.6,1.65] (1.65,1.7] (1.7,1.75]
```

```
factor(ifelse(altura<median(altura), "Pequeno", "Alto"),  
       levels=c("Pequeno", "Alto"))
```

```
## [1] Pequeno Pequeno Alto Pequeno Alto Alto Pequeno A
```

```
## Levels: Pequeno Alto
```



```
altura <- c(1.65, 1.67, 1.70, 1.58, 1.69, 1.73, 1.60, 1.70)
```

```
cut(altura, seq(1.55, by=.05, length.out=5))
```

```
## [1] (1.6,1.65] (1.65,1.7] (1.65,1.7] (1.55,1.6] (1.65,1.7] (1
```

```
## [8] (1.65,1.7]
```

```
## Levels: (1.55,1.6] (1.6,1.65] (1.65,1.7] (1.7,1.75]
```

```
factor(ifelse(altura<median(altura), "Pequeno", "Alto"),  
       levels=c("Pequeno", "Alto"))
```

```
## [1] Pequeno Pequeno Alto Pequeno Alto Alto Pequeno A
```

```
## Levels: Pequeno Alto
```



```
nFolhas
```

```
## [1] 15 20 25 17 26 19
```

```
# comprimentos dos ramos
```

```
compRamos <- c(6.0, 7.5, 9.2, 7.2, 8.9, 7.8)
```

```
nFolhas[1]/compRamos[1] # folhas por cm no ramo 1
```

```
## [1] 2.5
```

```
nFolhas/compRamos # folhas por cm
```

```
## [1] 2.500000 2.666667 2.717391 2.361111 2.921348 2.435897
```

## Operações com vetores



```
nFolhas
```

```
## [1] 15 20 25 17 26 19
```

```
# comprimentos dos ramos
```

```
compRamos <- c(6.0, 7.5, 9.2, 7.2, 8.9, 7.8)
```

```
nFolhas[1]/compRamos[1] # folhas por cm no ramo 1
```

```
## [1] 2.5
```

```
nFolhas/compRamos # folhas por cm
```

```
## [1] 2.500000 2.666667 2.717391 2.361111 2.921348 2.435897
```

## Operações com vetores



```
nFolhas
```

```
## [1] 15 20 25 17 26 19
```

```
# comprimentos dos ramos
```

```
compRamos <- c(6.0, 7.5, 9.2, 7.2, 8.9, 7.8)
```

```
nFolhas[1]/compRamos[1] # folhas por cm no ramo 1
```

```
## [1] 2.5
```

```
nFolhas/compRamos # folhas por cm
```

```
## [1] 2.500000 2.666667 2.717391 2.361111 2.921348 2.435897
```





## Operações com vetores

```
for(i in seq(1,6)) {  
  print(paste("Iterando na posição", i))  
  print(nFolhas[i]/compRamos[i])  
}
```

```
## [1] "Iterando na posição 1"  
## [1] 2.5  
## [1] "Iterando na posição 2"  
## [1] 2.666667  
## [1] "Iterando na posição 3"  
## [1] 2.717391  
## [1] "Iterando na posição 4"  
## [1] 2.361111  
## [1] "Iterando na posição 5"  
## [1] 2.921348  
## [1] "Iterando na posição 6"  
## [1] 2.435897
```



## Vetores e matrizes

```
# vetor  
c(1,2,3)
```

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

```
seq(1,50)
```

```
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
```

```
## [26] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44
```

```
# matriz  
matrix(seq(1,25), nrow=5)
```

```
##      [,1] [,2] [,3] [,4] [,5]  
## [1,]    1    6   11   16   21  
## [2,]    2    7   12   17   22  
## [3,]    3    8   13   18   23  
## [4,]    4    9   14   19   24  
## [5,]    5   10   15   20   25
```



```
M <- matrix(c(1,4,7, 2,5,8, 3,6,9), nrow=3)
dim(M)
```

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

```
M[1,3]
```

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

```
M[2,]
```

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

```
M[,3]
```

```
## [1] 3 6 9
```



```
M <- matrix(c(1,4,7, 2,5,8, 3,6,9), nrow=3)
dim(M)
```

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

```
M[1,3]
```

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

```
M[2,]
```

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

```
M[,3]
```

```
## [1] 3 6 9
```



```
(dados <- data.frame(blocos, tratamentos, nFolhas, compRamos))
```

##	blocos	tratamentos	nFolhas	compRamos
## 1	1	Trat 1	15	6.0
## 2	2	Trat 1	20	7.5
## 3	1	Trat 2	25	9.2
## 4	2	Trat 2	17	7.2
## 5	1	Trat 3	26	8.9
## 6	2	Trat 3	19	7.8



```
dados$blocos
```

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

```
## Levels: 1 2
```

```
dados$nFolhas
```

```
## [1] 15 20 25 17 26 19
```

```
str(dados)
```

```
## 'data.frame':    6 obs. of  4 variables:
##  $ blocos      : Factor w/ 2 levels "1","2": 1 2 1 2 1 2
##  $ tratamentos: Factor w/ 3 levels "Trat 2","Trat 1",...: 2 2
##  $ nFolhas     : num  15 20 25 17 26 19
##  $ compRamos   : num  6 7.5 9.2 7.2 8.9 7.8
```



```
dados$blocos
```

```
## [1] 1 2 1 2 1 2  
## Levels: 1 2
```

```
dados$nFolhas
```

```
## [1] 15 20 25 17 26 19
```

```
str(dados)
```

```
## 'data.frame':    6 obs. of  4 variables:  
## $ blocos      : Factor w/ 2 levels "1","2": 1 2 1 2 1 2  
## $ tratamentos: Factor w/ 3 levels "Trat 2","Trat 1",...: 2 2  
## $ nFolhas     : num  15 20 25 17 26 19  
## $ compRamos   : num  6 7.5 9.2 7.2 8.9 7.8
```



```
names(dados)
```

```
## [1] "blocos"      "tratamentos" "nFolhas"      "compRamos"
```

```
rownames(dados)
```

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

```
head(dados)
```

```
##   blocos tratamentos nFolhas compRamos
## 1      1      Trat 1      15      6.0
## 2      2      Trat 1      20      7.5
## 3      1      Trat 2      25      9.2
## 4      2      Trat 2      17      7.2
## 5      1      Trat 3      26      8.9
## 6      2      Trat 3      19      7.8
```





```
names(dados)
```

```
## [1] "blocos"      "tratamentos" "nFolhas"      "compRamos"
```

```
rownames(dados)
```

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

```
head(dados)
```

```
##      blocos  tratamentos  nFolhas  compRamos
## 1         1      Trat 1        15         6.0
## 2         2      Trat 1        20         7.5
## 3         1      Trat 2        25         9.2
## 4         2      Trat 2        17         7.2
## 5         1      Trat 3        26         8.9
## 6         2      Trat 3        19         7.8
```



```
names(dados)
```

```
## [1] "blocos"      "tratamentos" "nFolhas"      "compRamos"
```

```
rownames(dados)
```

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

```
head(dados)
```

```
##   blocos tratamentos nFolhas compRamos
## 1      1      Trat 1      15      6.0
## 2      2      Trat 1      20      7.5
## 3      1      Trat 2      25      9.2
## 4      2      Trat 2      17      7.2
## 5      1      Trat 3      26      8.9
## 6      2      Trat 3      19      7.8
```



```
dados[4,3]
```

```
## [1] 17
```

```
dados[dados$tratamentos=="Trat 2",]
```

```
##   blocos tratamentos nFolhas compRamos
## 3      1      Trat 2      25        9.2
## 4      2      Trat 2      17        7.2
```

```
dados$tratamentos=="Trat 2"
```

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



## Data frames

```
dados[4,3]
```

```
## [1] 17
```

```
dados[dados$tratamentos=="Trat 2",]
```

```
##   blocos tratamentos nFolhas compRamos
## 3      1      Trat 2      25         9.2
## 4      2      Trat 2      17         7.2
```

```
dados$tratamentos=="Trat 2"
```

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



```
dados[4,3]
```

```
## [1] 17
```

```
dados[dados$tratamentos=="Trat 2",]
```

```
##   blocos tratamentos nFolhas compRamos
## 3      1      Trat 2      25         9.2
## 4      2      Trat 2      17         7.2
```

```
dados$tratamentos=="Trat 2"
```

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



```
dados$nRamos <- c(1, 1, 2, 1, 2, 2)
```

```
dados
```

##	blocos	tratamentos	nFolhas	compRamos	nRamos
## 1	1	Trat 1	15	6.0	1
## 2	2	Trat 1	20	7.5	1
## 3	1	Trat 2	25	9.2	2
## 4	2	Trat 2	17	7.2	1
## 5	1	Trat 3	26	8.9	2
## 6	2	Trat 3	19	7.8	2



```
levels(dados$tratamentos)
```

```
## [1] "Trat 2" "Trat 1" "Trat 3"
```

```
levels(dados$tratamentos) <- c("Tratamento 2", "Tratamento 1",  
                                "Tratamento 3")
```

```
dados
```

```
##   blocos  tratamentos nFolhas compRamos nRamos  
## 1      1 Tratamento 1      15        6.0      1  
## 2      2 Tratamento 1      20        7.5      1  
## 3      1 Tratamento 2      25        9.2      2  
## 4      2 Tratamento 2      17        7.2      1  
## 5      1 Tratamento 3      26        8.9      2  
## 6      2 Tratamento 3      19        7.8      2
```



```
names(dados)
```

```
## [1] "blocos"      "tratamentos" "nFolhas"      "compRamos"    "
```

```
names(dados) <- c("bloc", "trat", "nFol", "cRam", "nRam")  
dados
```

```
##   bloc      trat nFol cRam nRam  
## 1    1 Tratamento 1   15  6.0    1  
## 2    2 Tratamento 1   20  7.5    1  
## 3    1 Tratamento 2   25  9.2    2  
## 4    2 Tratamento 2   17  7.2    1  
## 5    1 Tratamento 3   26  8.9    2  
## 6    2 Tratamento 3   19  7.8    2
```





```
aggregate(dados$nFol, mean, by=list(Trat=dados$trat))
```

```
##           Trat      x
## 1 Tratamento 2 21.0
## 2 Tratamento 1 17.5
## 3 Tratamento 3 22.5
```



```
expand.grid(c("Trat 1", "Trat 2", "Trat 3"),  
            c("Bloco 1", "Bloco 2"))
```

```
##      Var1    Var2  
## 1 Trat 1 Bloco 1  
## 2 Trat 2 Bloco 1  
## 3 Trat 3 Bloco 1  
## 4 Trat 1 Bloco 2  
## 5 Trat 2 Bloco 2  
## 6 Trat 3 Bloco 2
```



```
(lista <- list(c(1,2,3), M, dados))
```

```
## [[1]]
## [1] 1 2 3
##
## [[2]]
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
##
## [[3]]
##      bloc      trat nFol cRam nRam
## 1      1 Tratamento 1   15  6.0    1
## 2      2 Tratamento 1   20  7.5    1
## 3      1 Tratamento 2   25  9.2    2
## 4      2 Tratamento 2   17  7.2    1
## 5      1 Tratamento 3   26  8.9    2
## 6      2 Tratamento 3   19  7.8    2
```

# Exercícios



A partir do conjunto de dados `datasets::attitude`:

1. Criar um vetor da classe fator que divida a coluna `attitude$ratings` em dois níveis pela mediana.
2. Obter as médias de `learning` para valores abaixo e acima da mediana de `critical`.
3. Criar uma função que dicotomize qualquer vetor pela mediana, como feito no exercício 1, e depois aplicar essa função para todas as colunas do data frame `attitude`.

A família apply

## A família apply



```
(folhasERamos <- cbind(nFolhas, compRamos))
```

```
##      nFolhas compRamos
## [1,]      15      6.0
## [2,]      20      7.5
## [3,]      25      9.2
## [4,]      17      7.2
## [5,]      26      8.9
## [6,]      19      7.8
```

```
apply(folhasERamos, 1, function(x) x[1]/x[2])
```

```
## [1] 2.500000 2.666667 2.717391 2.361111 2.921348 2.435897
```

```
apply(folhasERamos, 2, sum)
```

```
##      nFolhas compRamos
##      122.0      46.6
```

## A família apply



```
(folhasERamos <- cbind(nFolhas, compRamos))
```

```
##      nFolhas compRamos
## [1,]      15      6.0
## [2,]      20      7.5
## [3,]      25      9.2
## [4,]      17      7.2
## [5,]      26      8.9
## [6,]      19      7.8
```

```
apply(folhasERamos, 1, function(x) x[1]/x[2])
```

```
## [1] 2.500000 2.666667 2.717391 2.361111 2.921348 2.435897
```

```
apply(folhasERamos, 2, sum)
```

```
##      nFolhas compRamos
##      122.0      46.6
```



## A família apply



```
(folhasERamos <- cbind(nFolhas, compRamos))
```

```
##      nFolhas compRamos
## [1,]      15      6.0
## [2,]      20      7.5
## [3,]      25      9.2
## [4,]      17      7.2
## [5,]      26      8.9
## [6,]      19      7.8
```

```
apply(folhasERamos, 1, function(x) x[1]/x[2])
```

```
## [1] 2.500000 2.666667 2.717391 2.361111 2.921348 2.435897
```

```
apply(folhasERamos, 2, sum)
```

```
##      nFolhas compRamos
##      122.0      46.6
```



```
for(i in seq(1, length(nFolhas))) {  
  nFolhas[i]/compRamos[i]  
}
```

```
sapply(seq(1, length(nFolhas)),  
       function(i) nFolhas[i]/compRamos[i])
```

```
## [1] 2.500000 2.666667 2.717391 2.361111 2.921348 2.435897
```

## A família apply



```
sapply(attitude, function(x) cut(x, c(-Inf, median(x), Inf)))
```

```
##      rating      complaints      privileges      learning      r
## [1,] "(-Inf,65.5]" "(-Inf,65]" "(-Inf,51.5]" "(-Inf,56.5]" "
## [2,] "(-Inf,65.5]" "(-Inf,65]" "(-Inf,51.5]" "(-Inf,56.5]" "
## [3,] "(65.5, Inf]" "(65, Inf]" "(51.5, Inf]" "(56.5, Inf]" "
## [4,] "(-Inf,65.5]" "(-Inf,65]" "(-Inf,51.5]" "(-Inf,56.5]" "
## [5,] "(65.5, Inf]" "(65, Inf]" "(51.5, Inf]" "(56.5, Inf]" "
## [6,] "(-Inf,65.5]" "(-Inf,65]" "(-Inf,51.5]" "(-Inf,56.5]" "
## [7,] "(-Inf,65.5]" "(65, Inf]" "(-Inf,51.5]" "(-Inf,56.5]" "
## [8,] "(65.5, Inf]" "(65, Inf]" "(-Inf,51.5]" "(-Inf,56.5]" "
## [9,] "(65.5, Inf]" "(65, Inf]" "(51.5, Inf]" "(56.5, Inf]" "
## [10,] "(65.5, Inf]" "(-Inf,65]" "(-Inf,51.5]" "(-Inf,56.5]" "
## [11,] "(-Inf,65.5]" "(-Inf,65]" "(51.5, Inf]" "(56.5, Inf]" "
## [12,] "(65.5, Inf]" "(-Inf,65]" "(-Inf,51.5]" "(-Inf,56.5]" "
## [13,] "(65.5, Inf]" "(-Inf,65]" "(51.5, Inf]" "(-Inf,56.5]" "
## [14,] "(65.5, Inf]" "(65, Inf]" "(51.5, Inf]" "(-Inf,56.5]" "
## [15,] "(65.5, Inf]" "(65, Inf]" "(51.5, Inf]" "(56.5, Inf]" "
## [16,] "(65.5, Inf]" "(65, Inf]" "(-Inf,51.5]" "(56.5, Inf]" "
## [17,] "(65.5, Inf]" "(65, Inf]" "(51.5, Inf]" "(56.5, Inf]" "
```



## A família apply

```
# Criando uma lista
```

```
lapply(c("a", "e", "i", "o", "u"),  
       function(x) {  
         gsub("[aeiou]", x,  
              "o sapo nao lava o pe")  
       })
```

```
## [[1]]  
## [1] "a sapa naa lava a pa"  
##  
## [[2]]  
## [1] "e sepe nee leve e pe"  
##  
## [[3]]  
## [1] "i sipi nii livi i pi"  
##  
## [[4]]  
## [1] "o sopo noo lovo o po"  
##  
## [[5]]
```



## A família apply

*# Criando um vetor*

```
sapply(c(1,2,3), divisao, a=2)
```

```
## [1] 2.0000000 1.0000000 0.6666667
```

```
sapply(c(1,2,3), divisao, b=2)
```

```
## [1] 0.5 1.0 1.5
```

Pacotes stringr e forcats



```
library(stringr)

str_sub(cidade, 1, 4) # substr(cidade, 1, 4)

## [1] "Pira"

str_detect(nomes, "e") # grepl("e", nomes)

## [1] TRUE FALSE TRUE FALSE

# sapply(tratamentos, function(x) {
#   regmatches(x, regexpr("\\d", x))})
str_extract(tratamentos, "\\d")

## [1] "1" "1" "2" "2" "3" "3"

str_extract_all(cidade, "[PrCb][ia]")

## [[1]]
## [1] "Pi" "ra" "ci" "ca" "ba"
```



## Pacote stringr

```
library(stringr)

str_sub(cidade, 1, 4) # substr(cidade, 1, 4)

## [1] "Pira"

str_detect(nomes, "e") # grepl("e", nomes)

## [1] TRUE FALSE TRUE FALSE
# sapply(tratamentos, function(x) {
#   regmatches(x, regexpr("\\d", x))})
str_extract(tratamentos, "\\d")

## [1] "1" "1" "2" "2" "3" "3"

str_extract_all(cidade, "[PrCb][ia]")

## [[1]]
## [1] "Pi" "ra" "ci" "ca" "ba"
```





## Pacote stringr

```
library(stringr)

str_sub(cidade, 1, 4) # substr(cidade, 1, 4)

## [1] "Pira"

str_detect(nomes, "e") # grepl("e", nomes)

## [1] TRUE FALSE TRUE FALSE

# sapply(tratamentos, function(x) {
#   regmatches(x, regexpr("\\d", x))})
str_extract(tratamentos, "\\d")

## [1] "1" "1" "2" "2" "3" "3"

str_extract_all(cidade, "[PrCb][ia]")

## [[1]]
## [1] "Pi" "ra" "ci" "ca" "ba"
```



## Pacote stringr

```
library(stringr)

str_sub(cidade, 1, 4) # substr(cidade, 1, 4)

## [1] "Pira"

str_detect(nomes, "e") # grepl("e", nomes)

## [1] TRUE FALSE TRUE FALSE

# sapply(tratamentos, function(x) {
#   regmatches(x, regexpr("\\d", x))})
str_extract(tratamentos, "\\d")

## [1] "1" "1" "2" "2" "3" "3"

str_extract_all(cidade, "[PrCb][ia]")

## [[1]]
## [1] "Pi" "ra" "ci" "ca" "ba"
```



```
# gsub("cicaba", "ssununga", cidade)
str_replace(cidade, "cicaba", "ssununga")
```

```
## [1] "Pirassununga"
```

```
# strwrap(textoComprido, width=10)
str_wrap(textoComprido, width=10)
```

```
## [1] "Um texto\nmuito\ncomprido,\nque\npode ser\nquebrado\nem"
```



```
# gsub("cicaba", "ssununga", cidade)
str_replace(cidade, "cicaba", "ssununga")
```

```
## [1] "Pirassununga"
```

```
# strwrap(textoComprido, width=10)
str_wrap(textoComprido, width=10)
```

```
## [1] "Um texto\nmuito\ncomprido,\nque\npode ser\nquebrado\nem"
```



```
library(forcats)
```

```
# relevel(tratamentos, "Trat 3")  
fct_relevel(tratamentos, "Trat 3")
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 3 Trat 2 Trat 1
```

```
# factor(tratamentos, levels=levels(tratamentos),  
#        labels=gsub("Trat", "Tratamento",  
#                    levels(tratamentos)))  
fct_relabel(tratamentos, str_replace, "Trat", "Tratamento")
```

```
## [1] Tratamento 1 Tratamento 1 Tratamento 2 Tratamento 2 Trata  
## [6] Tratamento 3  
## Levels: Tratamento 2 Tratamento 1 Tratamento 3  
fct_reorder(tratamentos, nFolhas)
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 1 Trat 2 Trat 3
```



```
library(forcats)
```

```
# relevel(tratamentos, "Trat 3")  
fct_relevel(tratamentos, "Trat 3")
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 3 Trat 2 Trat 1
```

```
# factor(tratamentos, levels=levels(tratamentos),  
#        labels=gsub("Trat", "Tratamento",  
#                     levels(tratamentos)))  
fct_relabel(tratamentos, str_replace, "Trat", "Tratamento")
```

```
## [1] Tratamento 1 Tratamento 1 Tratamento 2 Tratamento 2 Trata  
## [6] Tratamento 3  
## Levels: Tratamento 2 Tratamento 1 Tratamento 3
```

```
fct_reorder(tratamentos, nFolhas)
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 1 Trat 2 Trat 3
```



```
library(forcats)
```

```
# relevel(tratamentos, "Trat 3")  
fct_relevel(tratamentos, "Trat 3")
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 3 Trat 2 Trat 1
```

```
# factor(tratamentos, levels=levels(tratamentos),  
#        labels=gsub("Trat", "Tratamento",  
#                     levels(tratamentos)))  
fct_relabel(tratamentos, str_replace, "Trat", "Tratamento")
```

```
## [1] Tratamento 1 Tratamento 1 Tratamento 2 Tratamento 2 Trata  
## [6] Tratamento 3  
## Levels: Tratamento 2 Tratamento 1 Tratamento 3
```

```
fct_reorder(tratamentos, nFolhas)
```

```
## [1] Trat 1 Trat 1 Trat 2 Trat 2 Trat 3 Trat 3  
## Levels: Trat 1 Trat 2 Trat 3
```

Lendo arquivos externos



## Lendo arquivos externos



```
library(readr)
library(readxl)

read_xls("../dados/diario2023.xls")
```

```
## # A tibble: 36,366 x 24
##   Departamento de Engenh~1 ...2 ...3 ...4 ...5 `Ano 2020
##   <chr>                  <chr> <chr> <chr> <chr> <chr>
## 1 "Escola Superior de Agr~ <NA> <NA> <NA> <NA> <NA>
## 2 "Universidade de São Pa~ <NA> <NA> <NA> <NA> <NA>
## 3 "Estação Meteorológica ~ <NA> <NA> <NA> <NA> <NA>
## 4 "Local: Piracicaba (SP)~ <NA> <NA> <NA> <NA> <NA>
## 5 <NA>                  <NA> <NA> <NA> <NA> <NA>
## 6 "TOA5"                POST~ CR10~ 75696 CR10~ CPU:Estac
## 7 "TIMESTAMP"          RECO~ Batt~ Tar_~ UR_i~ Vvento_ms
## 8 "TS"                 RN    Volts Deg C %    meters/se
## 9 <NA>                 <NA> Avg   Avg   Smp   Avg
## 10 "44926"             8102 12.5~ 20.2~ 98    0.1000000
## # i 36,356 more rows
## # i abbreviated name: 1: `Departamento de Engenharia de Biosa
```

## Lendo arquivos externos



```
read_lines("../dados/DCE2023.TXT")
```

```
## [1] "=====
```

##	[2]	"No	ANO	DIA	MES	R.GLOBA	INSO-PRECIPIUMIDADE		
##	[3]	"				2	LACAO TACAO RELATIV		
##	[4]	"				cal/cm.	h/d	mm	%
##	[5]	"							
##	[6]	"1	2023	1	JAN	492	6,1	10,9	83
##	[7]	"2	2023	2	JAN	514	6,7	4,3	80
##	[8]	"3	2023	3	JAN	434	4,6	1,5	84
##	[9]	"4	2023	4	JAN	331	1,9	16,8	89
##	[10]	"5	2023	5	JAN	167	0,0	9,1	88
##	[11]	"6	2023	6	JAN	308	1,3	0,0	78
##	[12]	"7	2023	7	JAN	339	2,1	0,0	74
##	[13]	"8	2023	8	JAN	435	4,6	0,3	82
##	[14]	"9	2023	9	JAN	260	0,0	4,6	89
##	[15]	"10	2023	10	JAN	329	1,8	48,8	90
##	[16]	"11	2023	11	JAN	360	2,7	8,0	89
##	[17]	"12	2023	12	JAN	408	3,9	3,6	85
##	[18]	"13	2023	13	JAN	458	5,2	82,2	88

## Lendo arquivos externos



```
read_lines("http://www.leb.esalq.usp.br/leb/exceldados/DCE2023.TXT")
```

```
## [1] "=====
```

	No	ANO	DIA	MES	R.GLOBA	INSO-PRECIPIUMIDADE	VENTO		
					2	LACAO TACAO RELATIV	MAXIMO		
					cal/cm.	h/d	mm	%	m/s
## [6]	"1	2023	1	JAN	492	6,1	10,9	83	10,2
## [7]	"2	2023	2	JAN	514	6,7	4,3	80	12,0
## [8]	"3	2023	3	JAN	434	4,6	1,5	84	11,1
## [9]	"4	2023	4	JAN	331	1,9	16,8	89	10,4
## [10]	"5	2023	5	JAN	167	0,0	9,1	88	8,9
## [11]	"6	2023	6	JAN	308	1,3	0,0	78	12,3
## [12]	"7	2023	7	JAN	339	2,1	0,0	74	11,3
## [13]	"8	2023	8	JAN	435	4,6	0,3	82	8,6
## [14]	"9	2023	9	JAN	260	0,0	4,6	89	8,5
## [15]	"10	2023	10	JAN	329	1,8	48,8	90	8,1
## [16]	"11	2023	11	JAN	360	2,7	8,0	89	6,4
## [17]	"12	2023	12	JAN	408	3,9	3,6	85	6,7
## [18]	"13	2023	13	JAN	458	5,2	83,2	88	12,6
## [19]	"14	2023	14	JAN	514	6,7	4,6	84	8,6

# Exercícios



## Exercício 1

Transformar os objetos criados abaixo na estrutura a seguir.

```
tamanho <- c("Pequeno", "Médio", "Grande")  
id <- c(1, 2, 3, 1, 2, 1, 2, 3)
```

`dados.ex1`

```
##   id tamanho  
## 1  1      Peq  
## 2  2      Peq  
## 3  3      Peq  
## 4  1      Méd  
## 5  2      Méd  
## 6  1      Gra  
## 7  2      Gra  
## 8  3      Gra
```

`str(dados.ex1)`

```
## 'data.frame':   8 obs. of  2 variables:  
##  $ id      : num  1 2 3 1 2 1 2 3  
##  $ tamanho: chr  "Peq" "Peq" "Peq" "Méd" ...
```

## Exercício 2



Transformar os objetos criados abaixo na estrutura a seguir.

```
frase <- "eu vou comer laranjas e bananas"
```

```
vetor.ex2
```

```
##                                a
## "aa vaa camar laranjas a bananas" "ee vee cemer lerenjes e be
##                                i
## "ii vii cimir lirinjis i bininis" "oo voo comor loronjos o bo
##                                u
## "uu vuu cumur lurunjus u bununus"
```

```
str(vetor.ex2)
```

```
##  Named chr [1:5] "aa vaa camar laranjas a bananas" ...
## - attr(*, "names")= chr [1:5] "a" "e" "i" "o" ...
```

## Exercício 3



Transformar os objetos criados abaixo na estrutura a seguir.

```
nome <- c("Cecília", "Pedro", "Helena", "Júlia",  
          "Lorenzo", "Benício", "Lívia", "Theo")
```

```
vetor.ex3
```

```
##  Cecília    Pedro    Helena    Júlia    Lorenzo    Benício    Lívia  
##  "Menina" "Menino" "Menina" "Menina" "Menino" "Menino" "Menina"
```

```
str(vetor.ex3)
```

```
##  Named chr [1:8] "Menina" "Menino" "Menina" "Menina" "Menino"  
##  - attr(*, "names")= chr [1:8] "Cecília" "Pedro" "Helena" "Jú"
```

## Exercício 4



Transformar os objetos criados abaixo na estrutura a seguir.

```
estadoCivil <- c("Casado", "Solteiro", "Viúvo",  
                 "Solteiro", "Casado")
```

```
vetor.ex4
```

```
## [1] Casado   Solteiro Viúvo    Solteiro Casado  
## Levels: Solteiro Casado Viúvo
```

```
str(vetor.ex4)
```

```
## Factor w/ 3 levels "Solteiro","Casado",...: 2 1 3 1 2
```



## Exercício 5



Transformar os objetos criados abaixo na estrutura a seguir.

```
Cor <- c("Claro", "Escuro")
Tamanho <- c("Pequeno", "Médio", "Grande")
Rep <- seq(1,5)
Medida <- c(1.8, 15.4, 5.2, 2.2, 12.4, 20.0, 5.4, NA, 5.8, 5.4,
            NA, 7.0, 16.33, 27.33, 26.67, 12, 2, NA, 16.33, 11.17,
            12.5, 16, 17.75, 14.25, NA, 37, 49.5, NA, 47.5, 26.5)
```

```
head(dados.ex5)
```

```
##      Rep      Cor Tamanho Medida
## 1      1  Claro  Grande    1.8
## 2      2  Claro  Grande   15.4
## 3      3  Claro  Grande    5.2
## 4      4  Claro  Grande    2.2
## 5      5  Claro  Grande   12.4
## 6      1  Escuro  Grande   20.0
```

```
str(dados.ex5)
```

```
## 'data.frame':    30 obs. of  4 variables:
## $ Rep      : int  1 2 3 4 5 1 2 3 4 5 ...
## $ Cor       : Factor w/ 2 levels "Claro","Escuro": 1 1 1 1 1 2 2 2 2 2 ...
## $ Tamanho  : Factor w/ 3 levels "Grande","Médio",...: 1 1 1 1 1 1 1 1 1 1 ...
## $ Medida   : num  1.8 15.4 5.2 2.2 12.4 20 5.4 NA 5.8 5.4 ...
```

## Exercício 6



Transformar os objetos criados abaixo na estrutura a seguir.

```
col1 <- c(12,13,11,10,12,13,11)
col2 <- c(10,11,12,13,11,13,11)
vetor.num <- seq(12,15)
vetor.chr <- c("g.102", "g.104", "d.202", "d.104")
```

lista.ex6

```
## $Matriz
##      [,1] [,2]
## [1,]  12  10
## [2,]  13  11
## [3,]  11  12
## [4,]  10  13
## [5,]  12  11
## [6,]  13  13
## [7,]  11  11
##
## $Vetor
## [1] 12 13 14 15
##
## $DataFrame
##   vetor.num vetor.chr
## 1       12     g.102
## 2       13     g.104
## 3       14     d.202
## 4       15     d.104
```

## Exercício 7



Transformar os objetos criados abaixo na estrutura a seguir.

```
posicoes <- c(1,5,9,16,21)
consoantes <- letters[-posicoes]
vogais <- letters[posicoes]
```

lista.ex7

```
## $vogais
##           a           e           i           p           u
## "a é uma vogal" "e é uma vogal" "i é uma vogal" "p é uma vogal" "u é uma vogal"
##
## $consoantes
##           b           c           d           f
## "b é uma consoante" "c é uma consoante" "d é uma consoante" "f é uma consoante"
##           g           h           j           k
## "g é uma consoante" "h é uma consoante" "j é uma consoante" "k é uma consoante"
##           l           m           n           o
## "l é uma consoante" "m é uma consoante" "n é uma consoante" "o é uma consoante"
##           q           r           s           t
## "q é uma consoante" "r é uma consoante" "s é uma consoante" "t é uma consoante"
##           v           w           x           y
## "v é uma consoante" "w é uma consoante" "x é uma consoante" "y é uma consoante"
##           z
## "z é uma consoante"
```

## Exercício 8



Transformar os objetos criados abaixo na estrutura a seguir.

```
posicao <- c(seq(1,5),seq(1,4),seq(1,5),seq(1,3))
```

```
grupo <- rep(seq(1,4),c(5,4,5,3))
```

```
dados.ex8
```

```
##      grupo posicao primeiros últimos posicao.fct
## 1      1      1      TRUE  FALSE  Primeiro
## 2      1      2     FALSE  FALSE    Meio
## 3      1      3     FALSE  FALSE    Meio
## 4      1      4     FALSE  FALSE    Meio
## 5      1      5     FALSE  TRUE   Último
## 6      2      1      TRUE  FALSE  Primeiro
## 7      2      2     FALSE  FALSE    Meio
## 8      2      3     FALSE  FALSE    Meio
## 9      2      4     FALSE  TRUE   Último
## 10     3      1      TRUE  FALSE  Primeiro
## 11     3      2     FALSE  FALSE    Meio
## 12     3      3     FALSE  FALSE    Meio
## 13     3      4     FALSE  FALSE    Meio
## 14     3      5     FALSE  TRUE   Último
## 15     4      1      TRUE  FALSE  Primeiro
## 16     4      2     FALSE  FALSE    Meio
## 17     4      3     FALSE  TRUE   Último
```

```
str(dados.ex8)
```

```
## 'data.frame': 17 obs. of 5 variables:
## $ grupo : int 1 1 1 1 1 2 2 2 2 3 ...
## $ posicao : int 1 2 3 4 5 1 2 3 4 1 ...
## $ primeiros : logi TRUE FALSE FALSE FALSE TRUE ...
## $ últimos : logi FALSE FALSE FALSE FALSE TRUE FALSE ...
## $ posicao.fct: Factor w/ 3 levels "Primeiro","Meio",...: 1 2 2 2 3 1 2 2 3 1 ...
```

## Exercício 9



Transformar os objetos criados abaixo na estrutura a seguir.

```
fator <- factor(c(rep(seq(1,3),each=3)), levels=c(2, 1, 3))
```

```
fator
```

```
## [1] Nível 1 Nível 1 Nível 1 Nível 2 Nível 2 Nível 2 Nível 3 N  
## Levels: Nível 1 Nível 2 Nível 3
```

## Exercício 10



Transformar os objetos criados abaixo usando a função `str_replace` ou a função `gsub`.

```
nomes <- c("Larissa Cunha Azevedo", "Marisa Alves Cardoso",  
           "Enzo Martins Costa", "Luis Cunha Rodrigues",  
           "Antônio Castro Carvalho")
```

```
nomes
```

```
## [1] "Azevedo, Larissa C." "Cardoso, Marisa A." "Costa, Enzo  
## [4] "Rodrigues, Luis C." "Carvalho, Antônio C."
```