

# INTRODUÇÃO À PROGRAMAÇÃO UTILIZANDO O R

Aula 1 – Conceitos básicos  
Parte 2

# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ ***Primeiro contato: instalando***
  - *Operações aritméticas*
- ✓ *Usando o Rstudio*
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - *Booleanos*
  - *Vetores e Matrizes*
  - *Condicionais e Loops*
  - *Funções*



Primeiro você vai instalar o R e depois o R Studio.

Para instalar o R pela primeira vez, você pode usar este link aqui:

<https://cran-r.c3sl.ufpr.br/>

Escolha seu sistema operacional e faça a instalação. Em termos de língua (como Inglês/Português), não vai fazer muita diferença pois apenas vai mudar os menus, mas os pacotes e a ajuda estão em inglês.

O R Studio você pode baixar daqui:

<https://www.rstudio.com/products/rstudio/download/#download>

Você precisa primeiro concluir a instalação do R pois o R Studio abre o R e roda “por cima” dele.

Nós primeiro vamos fazer alguns exercícios no R e depois vamos passar o resto do curso no R Studio. Se você está estudando por conta quiser pular a parte do R, fique à vontade. 😊





Primeiro você vai instalar o R e depois o R Studio.

Para instalar o R pela primeira vez, você pode usar este link aqui:

<https://cran-r.c3sl.ufpr.br/>

Escolha seu sistema operacional e faça a instalação. Em termos de língua (como Inglês/Português), não vai fazer muita diferença pois apenas vai mudar os menus, mas os pacotes e a ajuda estão em inglês.

**Atenção:** estes slides foram pensados para serem usados na aula presencial, então algumas vezes pode ser ruim de usar eles caso você não tenha assistido a aula antes. Por exemplo, alguns dos exercícios foram feitos para dar errado para que se discuta a mensagem de erro na própria aula.

O R Studio você pode baixar daqui:

<https://www.rstudio.com/products/rstudio/download/#download>

Você precisa primeiro concluir a instalação do R pois o R Studio abre o R e roda “por cima” dele.

Nós primeiro vamos fazer alguns exercícios no R e depois vamos passar o resto do curso no R Studio. Se você está estudando por conta quiser pular a parte do R, fique à vontade. 😊



# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ ***Primeiro contato***
  - *Operações aritméticas*
- ✓ *Usando o Rstudio*
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - *Booleanos*
  - *Vetores e Matrizes*
  - *Condicionais e Loops*
  - *Funções*





R



Todas

Mapas

Notícias

Imagens

Vídeos

Mais ▼

Ferramentas de pesquisa

Aproximadamente 8.830.000.000 resultados (0,43 segundos)

### R: The R Project for Statistical Computing

<https://www.r-project.org/> ▼ Traduzir esta página

R, also called GNU S, is a strongly functional language and environment to statistically explore data sets, make many graphical displays of data from custom ...

[Mirrors](#) - [FAQs](#) - [Manuals](#) - [Books](#)

Você visitou esta página 3 vezes. Última visita: 04/05/16

RGui (64-bit)

Arquivo Editar Visualizar Misc Pacotes Janelas Ajuda



R Console

```
R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"  
Copyright (C) 2013 The R Foundation for Statistical Computing  
Platform: x86_64-w64-mingw32/x64 (64-bit)
```

```
R é um software livre e vem sem GARANTIA ALGUMA.  
Você pode redistribuí-lo sob certas circunstâncias.  
Digite 'license()' ou 'licence()' para detalhes de distribuição.
```

```
R é um projeto colaborativo com muitos contribuidores.  
Digite 'contributors()' para obter mais informações e  
'citation()' para saber como citar o R ou pacotes do R em publicações.
```

```
Digite 'demo()' para demonstrações, 'help()' para o sistema on-line de ajuda,  
ou 'help.start()' para abrir o sistema de ajuda em HTML no seu navegador.  
Digite 'q()' para sair do R.
```

```
[Área de trabalho anterior carregada]
```

```
> |
```



# SUJANDO AS MÃOS





A computer monitor with a black bezel and a silver stand. The screen is white and displays five lines of R code in green text. The code is as follows:

```
# Usando o R como calculadora  
# Calcule dois mais três  
  
# Calcule sete vezes trinta mais quatro menos vinte  
  
# Calcule sete vezes (trinta mais quatro) menos vinte  
  
# Calcule sete vezes (trinta mais quatro menos vinte)
```

# Usando o R como calculadora

# Calcule dois mais três

# Calcule sete vezes trinta mais quatro menos vinte

# Calcule sete vezes (trinta mais quatro) menos vinte

# Calcule sete vezes (trinta mais quatro menos vinte)

A computer monitor with a black bezel and a silver stand. The screen is white and displays several lines of R code in green and red text. The code demonstrates the use of the R console as a calculator for various arithmetic operations.

```
# Usando o R como calculadora
```

```
# Calcule dois mais três
```

```
> 2+3
```

```
# Calcule sete vezes trinta mais quatro menos vinte
```

```
> 7*30+4-20
```

```
# Calcule sete vezes (trinta mais quatro) menos vinte
```

```
> 7*(30+4)-20
```

```
# Calcule sete vezes (trinta mais quatro menos vinte)
```

```
> 7*(30+4-20)
```

# Usando o R como calculadora

# Calcule dois mais três

> 2+3

# Calcule sete vezes trinta mais quatro menos vinte

> 7\*30+4-20

194

# Calcule sete vezes (trinta mais quatro) menos vinte

> 7\*(30+4)-20

218

# Calcule sete vezes (trinta mais quatro menos vinte)

> 7\*(30+4-20)

98



# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ ***Primeiro contato***
  - *Operações aritméticas*
- ✓ *Usando o Rstudio*
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - *Booleanos*
  - *Vetores e Matrizes*
  - *Condicionais e Loops*
  - *Funções*



## OPERAÇÕES BÁSICAS:

- Adição (+)
- Subtração (-)
- Multiplicação (\*)
- Divisão (/)
- Exponenciação (^)
- Módulo (divisão inteira) (%)
- Raiz quadrada (sqrt())



# SUJANDO AS MÃOS







# Calcule catorze menos três

# Calcule três menos catorze

# Calcule três menos catorze ao quadrado e catorze menos três ao quadrado

# Calcule a raiz quadrada de três menos catorze ao quadrado

# Calcule o módulo de catorze por três



# Calcule catorze menos três

> 14 - 3

# Calcule três menos catorze

> 3 - 14

# Calcule três menos catorze ao quadrado e catorze menos três ao quadrado

> (3-14)^2

> (14-3)^2

# Calcule a raiz quadrada de três menos catorze ao quadrado

> sqrt((14-3)^2)

# Calcule o módulo de catorze por três

> 14 %% 3

# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ *Primeiro contato*
- ✓ ***Usando o Rstudio***
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - *Booleanos*
  - *Vetores e Matrizes*
  - *Condicionais e Loops*
  - *Funções*







rstudio



Todas

Vídeos

Imagens

Notícias

Livros

Mais ▼

Ferramentas de pesquisa

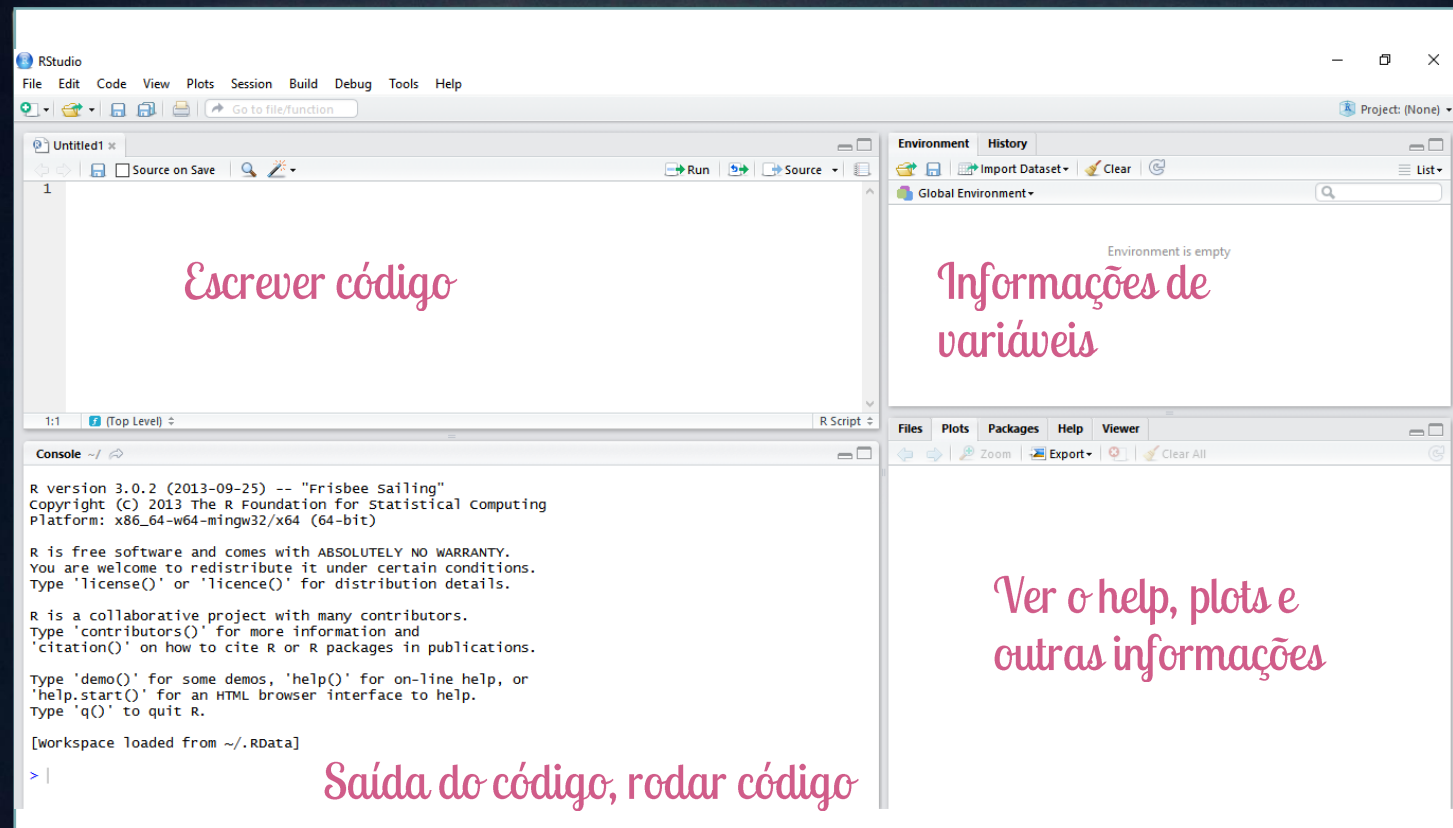
Aproximadamente 472.000 resultados (0,47 segundos)

**RStudio – Open source and enterprise-ready professional software for R**

<https://www.rstudio.com/> ▼ Traduzir esta página

A powerful and productive user interface for R. It's free and open source, and works great on Windows, Mac, and Linux.

Você visitou esta página 4 vezes. Última visita: 30/05/16





# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ *Primeiro contato*
- ✓ *Usando o Rstudio*
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - *Booleanos*
  - *Vetores e Matrizes*
  - *Loops*
  - *Funções*





# SUJANDO AS MÃOS



# Calcule catorze menos três

# Calcule três menos catorze

# Calcule três menos catorze ao quadrado e catorze menos três ao quadrado

# Calcule a raiz quadrada de três menos catorze ao quadrado

# Calcule o módulo de catorze por três

Após fazer no console  
(esquerda abaixo), fazer na  
janela de edição de código  
(esquerda acima)

# Pedindo ajuda

```
> help.start()      # abre a ajuda
> ?hist             # abre a ajuda da função hist()
> help(hist)        # idem anterior
> ??histogram       # procura na ajuda por páginas com histogram
> apropos("hist")   # lista as funções que tenham "hist" no nome
> example(hist)     # mostra um exemplo usando a função hist
```

# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ *Primeiro contato*
- ✓ *Usando o Rstudio*
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - ***Booleanos***
  - *Vetores e Matrizes*
  - *Condicionais e Loops*
  - *Funções*





# BOOLEANOS

- Falso (FALSE ou F)
- Verdadeiro (TRUE ou T)
- Negação (!)
- E (and) (&)
- Ou (or) (|)
- Ou exclusivo (xor)

# BOOLEANOS

- Falso (FALSE ou F ou 0)
- Verdadeiro (TRUE ou T ou 1)
- Negação (!)
- E (and) (&)
- Ou (or) (|)
- Ou exclusivo (xor)

São diferentes:

x & y  
x && y  
x | y  
x || y

O operador simples faz a comparação entre **todos** os elementos de um objeto, enquanto que o segundo compara apenas os **primeiros** elementos.

# REVISÃO DE TABELAS VERDADE

x	y	$\neg x$	$x \& y$	$x   y$	$x \oplus y$
0	0				
0	1				
1	0				
1	1				

# REVISÃO DE TABELAS VERDADE

x	y	$\neg x$	$x \& y$	$x   y$	$x \oplus y$
0	0	1	0	0	0
0	1	1	0	1	1
1	0	0	0	1	1
1	1	0	1	1	0



# SUJANDO AS MÃOS



# Associe o vetor  $(0,0,1,1)$  à variável  $x$  e  $(0,1,0,1)$  à variável  $y$  e calcule a tabela do exemplo anterior

# Associe o vetor (0,0,1,1) à variável x e (0,1,0,1) à variável y e calcule a tabela do exemplo anterior

> x<- c(0,0,1,1)

x	y	!x	x&y	x y	xor(x,y)
0	0	1	0	0	0
0	1	1	0	1	1
1	0	0	0	1	1
1	1	0	1	1	0

# Associe o vetor (0,0,1,1) à variável x e (0,1,0,1) à variável y e calcule a tabela do exemplo anterior

```
> x<- c(0,0,1,1)
```

```
> y<- c(0,1,0,1)
```

```
>!x
```

```
> x&y
```

```
> x&&y
```

```
> x|y
```

```
> x||y
```

```
> xor(x,y)
```



Um parênteses sobre usar uma ou outra forma para criar variáveis no R.

É <- ou  
= ???





**= BOLACHA**



**= BISCOITO**



A computer monitor with a black bezel and a silver stand. The screen is white and displays three lines of green text, which are R code comments. The text is left-aligned and uses a monospaced font. The monitor is centered in the frame.

```
# Verificando a precedência (quem é calculado antes):
```

```
# Armazene em a (usando flecha) o termo b = 1
```

```
# Armazene em a (usando =) o termo b <- 1 e imprima a e b
```



A computer monitor with a black bezel and a silver stand. The screen is white and displays several lines of R code in green and red text. The code is as follows:

```
# Verificando a precedência (quem é calculado antes):  
# Armazene em a (usando flecha) o termo b = 1  
a <- b = 1  
  
# Armazene em a (usando =) o termo b <- 1 e imprima a e b  
  
a = b <- 1  
a  
b
```

# Verificando a precedência (quem é calculado antes):

# Armazene em a (usando flecha) o termo b = 1

a <- b = 1

# Armazene em a (usando =) o termo b <- 1 e imprima a e b

a = b <- 1

a

b

# Verificando a precedência (quem é calculado antes):

# Armazene em a (usando flecha) o termo b = 1

```
a <- b = 1
```

# Armazene em a (usando =) o termo b <- 1 e imprima a e b

```
a = b <- 1
```

```
a
```

```
b
```

Observação:

Nas funções, os argumentos sempre usarão o sinal de igual:

```
rnorm(10, mean = 10, sd = 1)
```

# Verificando a precedência (quem é calculado antes):

# Armazene em a (usando flecha) o termo b = 1

a <- b = 1

# Armazene em a (usando =) o termo b <- 1 e imprima a e b

a = b <- 1

a

b

Mas, como tudo na vida, tem exceção:

median(x = 1:10) versus median(x <- 1:10)

Existe diferença entre atribuir valor usando '<-' ou '=' em R?

<http://pt.stackoverflow.com/questions/160029/existe-diferen%C3%A7a-entre-atribuir-valor-usando-ou-em-r>

What are differences in the assignment operators '=' and '<-' in R?

<http://stackoverflow.com/questions/1741820/assignment-operators-in-r-and>

R Style Guide:

<http://adv-r.had.co.nz/Style.html>



# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ *Primeiro contato*
- ✓ *Usando o Rstudio*
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - *Booleanos*
  - *Vetores e Matrizes* ← “Guardando” valores em variáveis e tipos de variáveis
  - *Condicionais e Loops*
  - *Funções*



# TIPOS DE VARIÁVEIS NO R

- Numérico (valores decimais)

# TIPOS DE VARIÁVEIS NO R

- Numérico (valores decimais)
- Inteiro (valores inteiros)

# TIPOS DE VARIÁVEIS NO R

- Numérico (valores decimais)
- Inteiro (valores inteiros)
- Complexo (números complexos)



## TIPOS DE VARIÁVEIS NO R

- Numérico (valores decimais)
- Inteiro (valores inteiros)
- Complexo (números complexos)
- Lógico (falso/verdadeiro)

## TIPOS DE VARIÁVEIS NO R

- Numérico (valores decimais)
- Inteiro (valores inteiros)
- Complexo (números complexos)
- Lógico (falso/verdadeiro)
- Caracter ('palavras')

# SUJANDO AS MÃOS



A computer monitor with a black bezel and a silver stand. The screen is white and displays several lines of R code in green and red text. The code is as follows:

```
# Associe o valor 10.5 à variável x  
  
# Mostre o valor de x na tela  
> x  
  
# Veja qual a "classe" da variável x  
> class(x)  
  
# Faça a mesma coisa, mas associando o número 4 à variável y  
  
# Utilize o comando is.integer(y)
```

# Associe o valor 10.5 à variável x

# Mostre o valor de x na tela

> x

# Veja qual a "classe" da variável x

> class(x)

# Faça a mesma coisa, mas associando o número 4 à variável y

# Utilize o comando is.integer(y)



A computer monitor with a black bezel and a silver stand. The screen displays R code in green and red text. The code is as follows:

```
# Associe o valor 10.5 à variável x
> x<- 10.5

# Imprima (print) o valor de x na tela
> x
# Veja qual a "classe" da variável x
> class(x)

# Faça a mesma coisa, mas associando o número 4 à variável y
> y<- 4
> Y
> class(y)

# Utilize o comando is.integer(y)
```

# Associe o valor 10.5 à variável x

> x<- 10.5

# Imprima (print) o valor de x na tela

> x

# Veja qual a "classe" da variável x

> class(x)

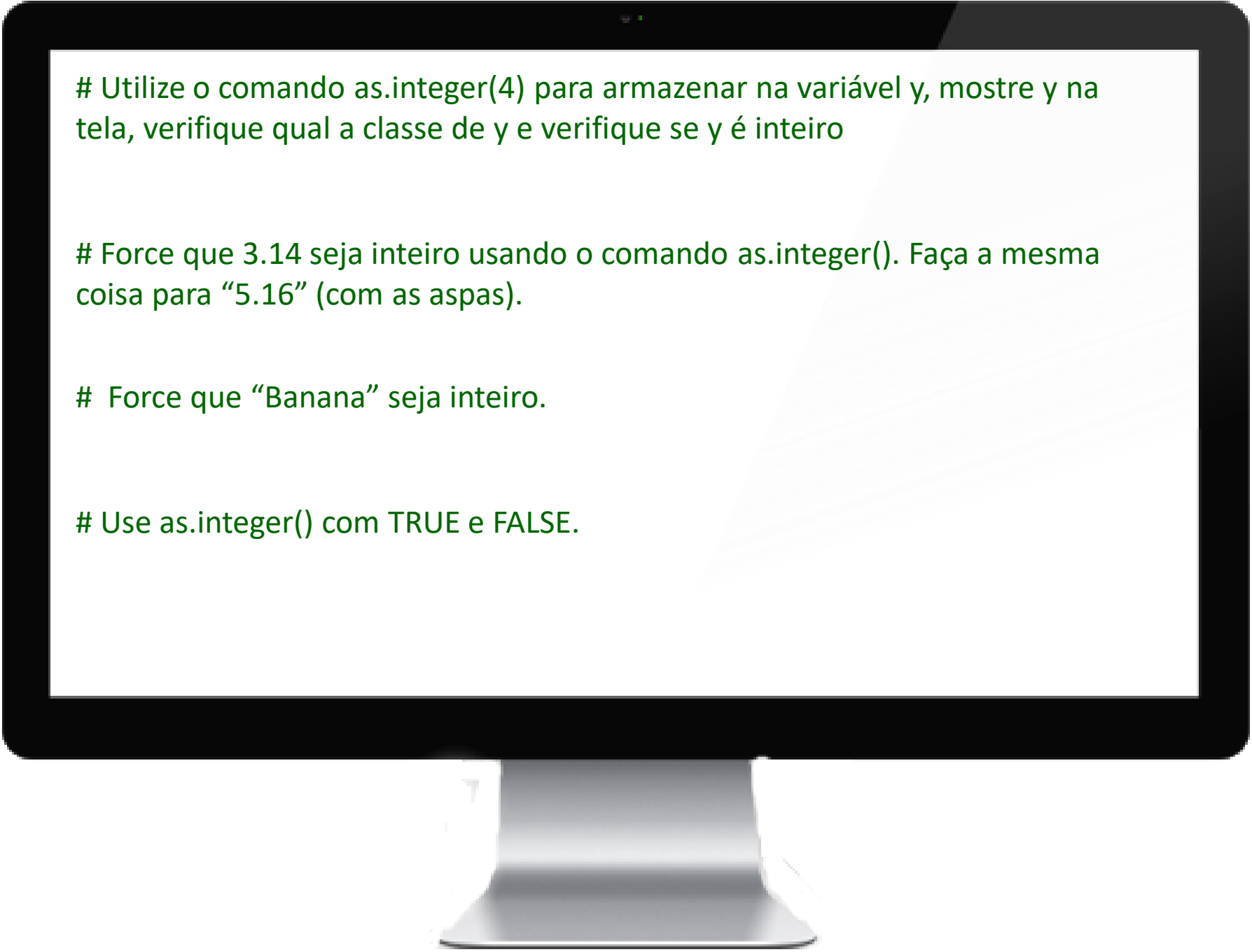
# Faça a mesma coisa, mas associando o número 4 à variável y

> y<- 4

> Y

> class(y)

# Utilize o comando is.integer(y)

A computer monitor with a black bezel and a silver stand. The screen is white and displays four lines of green text. The text is a mix of comments and instructions in Portuguese, related to the Python `as.integer()` function.

# Utilize o comando `as.integer(4)` para armazenar na variável `y`, mostre `y` na tela, verifique qual a classe de `y` e verifique se `y` é inteiro

# Force que `3.14` seja inteiro usando o comando `as.integer()`. Faça a mesma coisa para `"5.16"` (com as aspas).

# Force que `"Banana"` seja inteiro.

# Use `as.integer()` com `TRUE` e `FALSE`.

# Utilize o comando `as.integer(4)` para armazenar na variável `y`, mostre `y` na tela, verifique qual a classe de `y` e verifique se `y` é inteiro

```
> y <- as.integer(4)
> y
> class(y)
> is.integer(y)
```

# Verifique a classe de `3.14`. Force que `3.14` seja inteiro usando o comando `as.integer()`. Faça a mesma coisa para `"5.16"` (com as aspas).

```
> class(3.14)
> as.integer(3.14)
> class("5.16")
> as.integer("5.16")
```

A computer monitor with a black bezel and a silver stand. The screen is white and displays R code. The code is color-coded: comments are green and commands are red. The code demonstrates how to force a string to be an integer and how to use the as.integer() function with TRUE and FALSE.

```
# Force que "Banana" seja inteiro.  
> as.integer("Banana")  
  
# Use as.integer() com TRUE e FALSE.  
> as.integer(TRUE)  
> as.integer(FALSE)
```



A computer monitor with a black bezel and a silver stand. The screen is white and displays four lines of R code. The first line is a comment in green. The second line is an assignment in red. The third and fourth lines are comments in green.

```
# Exemplo com números complexos  
> z <- 1 + 2i  
# Imprima o valor de z  
# Verifique a classe de z
```

A computer monitor with a black bezel and a silver stand. The screen is white and displays R code in green and red text. The code is as follows:

```
# Exemplo com números complexos  
> z <- 1 + 2i  
  
# Imprima o valor de z  
  
> z  
  
# Verifique a classe de z  
  
> class(z)
```

# Exemplo com números complexos

> z <- 1 + 2i

# Imprima o valor de z

> z

# Verifique a classe de z

> class(z)



```
# Armazene 1 e 2 em x e y, respectivamente
```

```
# Verifique se  $x > y$ 
```

```
# Crie uma variável z onde z é dada por  $x > y$ 
```

```
# Imprima z e verifique a classe de z
```

A computer monitor with a black bezel and a silver stand. The screen is white and displays R code in green and red text. The code is as follows:

```
# Armazene 1 e 2 em x e y, respectivamente  
> x <- 1  
> y <- 2  
  
# Verifique se x > y  
> x > y  
  
# Crie uma variável z onde z é dada por x > y  
> z <- x > y  
  
# Imprima z e verifique a classe de z  
> z  
> class(z)
```

# Armazene 1 e 2 em x e y, respectivamente

> x <- 1

> y <- 2

# Verifique se x > y

> x > y

# Crie uma variável z onde z é dada por x > y

> z <- x > y

# Imprima z e verifique a classe de z

> z

> class(z)



A computer monitor with a black bezel and a silver stand. The screen is white and displays four lines of green text. The text is a list of instructions in Portuguese, likely for a programming exercise. The instructions are: 1. Store the value 3.14 in a variable named x, print x, and check its class. 2. Create a variable named primeiro with a first name (in quotes) and a variable named ultimo with a last name. Use the paste() function to print both variables together (if necessary, use the help). 3. Use the sprintf() function to print "Meu nome é %s %s", primeiro, ultimo. 4. At home, read the help of the sub() function.

# Armazene a variável 3.14 como caractere em x, imprima x e verifique a classe de x

# Crie uma variável chamada primeiro e armazene seu primeiro nome (com aspas) e uma variável chamada ultimo com seu sobrenome. Utilize o comando paste() para imprimir as variáveis juntas (se necessário, use a ajuda)

# Utilize o comando sprintf("Meu nome é %s %s", primeiro, ultimo)

# Em casa, leia o help da função sub()

# Armazene a variável 3.14 como caractere em x, imprima x e verifique a classe de x

```
> x<-as.character(3.14)
```

```
> x
```

```
> class(x)
```

# Crie uma variável chamada primeiro e armazene seu primeiro nome (com aspas) e uma variável chamado ultimo com seu sobrenome. Utilize o comando paste() para imprimir as variáveis juntas (se necessário, use a ajuda)

```
> primeiro <- "Aishameriane"
```

```
> ultimo <- "Schmidt"
```

```
> paste(primeiro, ultimo)
```

```
> sprintf("Meu nome é %s %s", primeiro, ultimo)
```

# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ *Primeiro contato*
- ✓ *Usando o Rstudio*
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - *Booleanos*
  - ***Vetores e Matrizes*** ← “Guardando” valores em variáveis e tipos de variáveis
  - *Condicionais e Loops*
  - *Funções*



## VETORES EM R

- Sequencia de elementos de um mesmo tipo
- Podemos usar a função de concatenar `c()` para declarar um vetor
- Podemos fazer operações de soma, multiplicação, etc, elemento a elemento quando dois vetores tem o mesmo tamanho



# SUJANDO AS MÃOS



# Armazene o vetor (2,3,4,5) na variável x , imprima e verifique sua classe.

# Armazene o vetor (T,F,T,F) na variável y, imprima e verifique sua classe.

# Armazene seu nome e sobrenome na variável nome , imprima e verifique sua classe.

# Armazene (1,2,"aa",3) na variável z , imprima e verifique sua classe.

# Utilize o comando length() para verificar o tamanho de y

# Armazene o vetor (2,3,4,5) na variável x , imprima e verifique sua classe.

```
x<- c (2,3,4,5)
```

```
x
```

```
class(x)
```

# Armazene o vetor (T,F,T,F) na variável y, imprima e verifique sua classe.

```
y<- c (T,F,T,F)
```

```
y
```

```
class(y)
```

# Armazene seu nome e sobrenome na variável nome , imprima e verifique sua classe.

```
nome <- c("Aisha","Schmidt")
```

```
nome
```

```
class(nome)
```

A computer monitor with a black bezel and a silver stand. The screen is white and displays R code in green and red text. The code includes a comment, variable assignment, printing, class checking, and length checking.

```
# Armazene (1,2,"aa",3) na variável z , imprima e verifique sua classe.  
z <- c(1,2,"aa",3)  
z  
class(z)  
  
# Utilize o comando length() para verificar o tamanho de y  
length(y)
```



A computer monitor with a black bezel and a silver stand. The screen is white and displays a list of seven vector operations in green text. The operations are: concatenating vectors x and z, creating vectors a and b with specific values, adding a and b, multiplying a by 5, subtracting b from a, multiplying a and b, and dividing a by b.

# Concatene os vetores x e z

# Crie os vetores a e b com os valores 1,3,4,5 e 1,2,3,4

# Faça a soma de a e b

# Multiplique a por 5

# Subtraia b de a

# Multiplique a e b

# Divida a e b

A computer monitor with a black bezel and a silver stand. The screen is white and displays several lines of R code. The code is color-coded: comments are in green and expressions are in red. The code includes concatenation, vector creation, arithmetic operations (addition, multiplication, subtraction, division), and assignment.

```
# Concatene os vetores x e z  
c(x,z)  
# Crie os vetores a e b com os valores 1,3,4,5 e 1,2,3,4  
a<-c(1,3,4,5)  
b<-c(1,2,3,4)  
# Faça a soma de a e b  
a + b  
# Multiplique a por 5  
5*a  
# Subtraia b de a  
a-b  
# Multiplique a e b  
a*b  
# Divida a por b  
a/b
```

# Salve o vetor 1 a 10 em x utilizando <- e o vetor 11 a 20 em y utilizando =  
(dica: utilize alt e o sinal de menos para o atalho da flechinha)  
Imprima os valores de x e y na tela

# Calcule a média de x utilizando mean(x) e depois calcule  
mean( z<- c(1,2,3,4,5,6,7,8,9,10))

# Calcule a média de y utilizando mean(y) e depois calcule  
mean(w = c(11,12,13,14,15,16,17,18,19,20))

# Salve o vetor 1 a 10 em x utilizando <- e o vetor 11 a 20 em y utilizando =  
(dica: utilize alt e o sinal de menos para o atalho da flechinha)

Imprima os valores de x e y na tela

```
x<- c(1,2,3,4,5,6,7,8,9,10)
```

```
y<- c(11,12,13,14,15,16,17,18,19,20)
```

```
x
```

```
y
```

# Calcule a média de x utilizando mean(x) e depois calcule  
mean( z<- c(1,2,3,4,5,6,7,8,9,10))

# Calcule a média de y utilizando mean(y) e depois calcule  
mean(w = c(11,12,13,14,15,16,17,18,19,20))

**O erro é de propósito.** Na linha anterior, ele cria a variável z e calcula a média. Nesta linha de baixo ele não sabe o que fazer com w.





```
# Calcule 25/3
```

```
# Calcule 25 %/% 3
```

```
# Calcule 25 %% 3 e compare com os dois resultados anteriores
```

```
# Monte o vetor da salada de frutas que contenha: laranja, maçã, banana,  
tangerina e limão. Monte a sua sacola de compras com laranja e banana.  
Compare as duas variáveis utilizando o símbolo ==
```

```
# Calcule 25/3
```

```
25/3
```

```
# Calcule 25 %/% 3
```

```
25 % / % 3
```

```
# Calcule 25 %% 3 e compare com os dois resultados anteriores
```

```
25 %% 3
```

```
8.3333
```

(divisão exata)

```
8
```

(parte inteira)

```
1
```

(depois de dividir 25 por 3, sobra 1)

```
# Monte o vetor da salada de frutas que contenha: laranja, maçã, banana,  
tangerina e limão. Monte a sua sacola de compras com laranja e banana.
```

```
Compare as duas variáveis utilizando o símbolo ==
```

```
salada_de_frutas <- c("laranja", "maçã", "banana", "tangerina", "limão")
```

```
minha_sacola <- c("laranja", "banana")
```

```
minha_sacola == saladade_frutas
```

A comparação aqui é posição a posição, ou seja, o R vai verificar se a posição 1 da sacola e da salada de frutas são iguais, depois a posição 2 e depois vai repetir a minha sacola até poder verificar para todos elementos da salada de frutas.

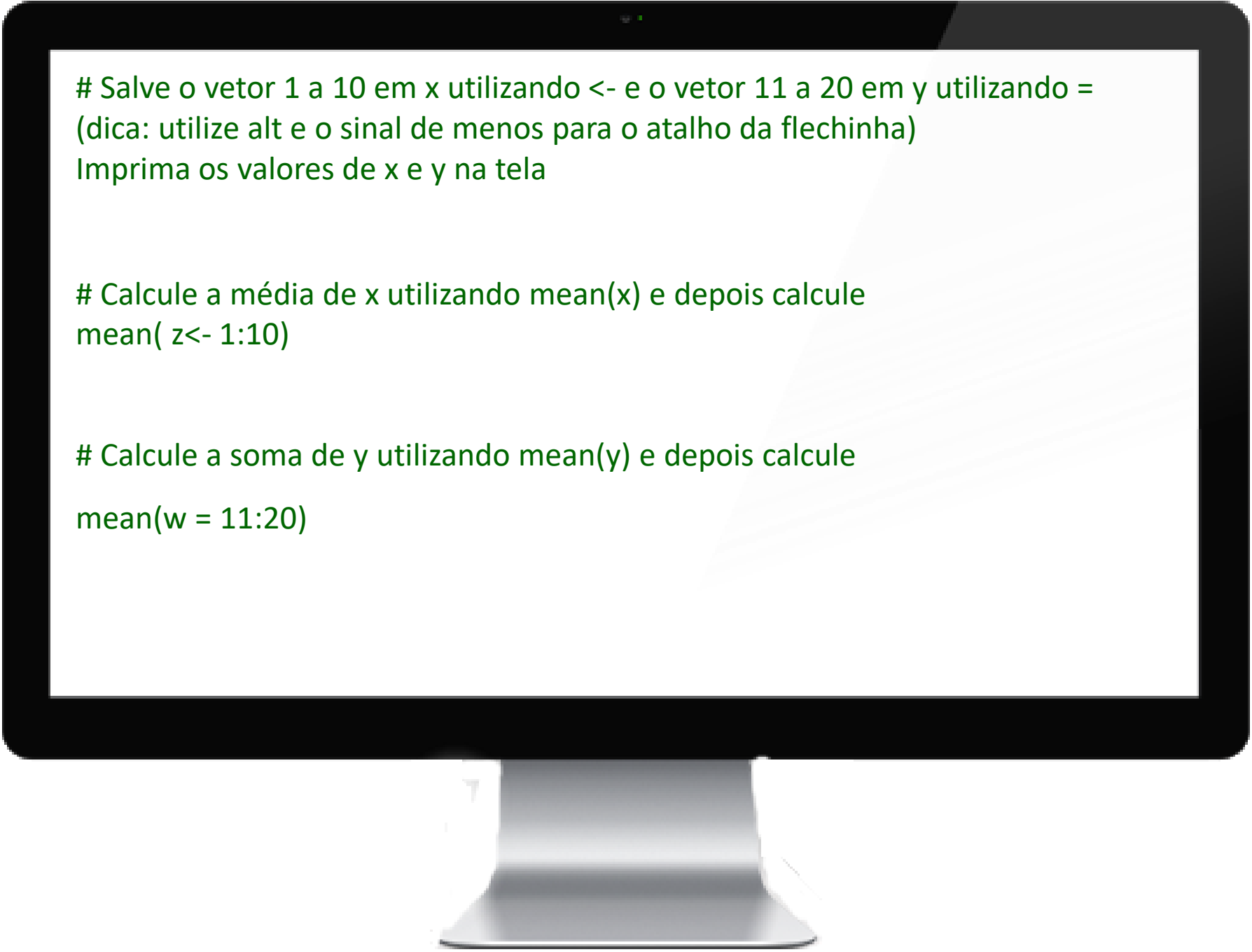
## CRIANDO REPETIÇÕES E SEQUÊNCIAS

- O R tem funções que podem ser utilizadas para automatizar a criação de vetores e sequencias
- `n:m` faz o R exibir um vetor de valores que vão de `n` a `m` com intervalos de 1 unidade

# SUJANDO AS MÃOS







# Salve o vetor 1 a 10 em x utilizando <- e o vetor 11 a 20 em y utilizando =  
(dica: utilize alt e o sinal de menos para o atalho da flechinha)  
Imprima os valores de x e y na tela

# Calcule a média de x utilizando mean(x) e depois calcule  
mean(z<- 1:10)

# Calcule a soma de y utilizando mean(y) e depois calcule  
mean(w = 11:20)

# Salve o vetor 1 a 10 em x utilizando <- e o vetor 11 a 20 em y utilizando =  
(dica: utilize alt e o sinal de menos para o atalho da flechinha)

Imprima os valores de x e y na tela

```
x<- 1:10
```

```
y<- 11:20
```

```
x
```

```
y
```

# Calcule a média de x utilizando mean(x) e depois calcule  
mean( z<- 1:10)

# Calcule a média de y utilizando mean(y) e depois calcule  
mean(w = 11:20)

A computer monitor with a black bezel and a silver stand. The screen is white and displays five lines of text in green, each preceded by a hash symbol (#).

# Exiba os números de 1 a 10 em ordem crescente

# Armazene os números de 1 a 10 em ordem decrescente na variável y e verifique seu tamanho e tipo

# Construa uma variável x que tenha os números de 78.7 a 200, em ordem crescente

# Imprima x na tela

# Verifique o tamanho e a classe de x

A computer monitor with a black bezel and a silver stand. The screen displays R code in green and red text. The code includes comments in green and commands in red. The background of the screen has a faint, light blue geometric pattern.

```
# Exiba os números de 1 a 10 em ordem crescente
```

```
1 : 10
```

```
# Armazene os números de 1 a 10 em ordem decrescente na variável y e  
verifique seu tamanho e tipo
```

```
y <- 10 : 1
```

```
length(y)
```

```
class(y)
```

```
# Construa uma variável x que tenha os números de 78.7 a 200, em ordem  
crescente
```

```
x <- 78.7 : 200
```

```
# Imprima x na tela
```

```
x
```

```
# Verifique o tamanho e a classe de x
```

```
length(x)
```

```
class(x)
```

# FUNÇÕES `rep()` E `seq()`

## `rep {base}`

`rep` replicates the values in `x`. It is a generic function, and the (internal) default method is described here.

Fonte: Texto de ajuda da função `rep()`.

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/rep.html>

## `seq {base}`

Generate regular sequences. `seq` is a standard generic with a default method. `seq.int` is a primitive which can be much faster but has a few restrictions. `seq_along` and `seq_len` are very fast primitives for two common cases.

Fonte: Texto de ajuda da função `rep()`.

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/rep.html>



# FUNÇÕES REP() E SEQ()

```
rep {base}
```

```
rep(x, ...)
```

Fonte: Texto de ajuda da função rep().

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/rep.html>

```
seq {base}
```

```
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)), length.out = NULL,  
along.with = NULL, ...)
```

Fonte: Texto de ajuda da função rep().

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/seq.html>

# FUNÇÕES `rep()` E `seq()`

```
rep {base}
```

```
rep(x, ...)
```

Argumentos:

x - valor que se deseja repetir

each - quantas vezes cada valor vai ser repetido

times - quantas vezes cada repetição vai aparecer

len - tamanho da sequencia gerada

Fonte: Texto de ajuda da função `rep()`.

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/rep.html>

# FUNÇÕES REP() E SEQ()

seq {base}

```
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)), length.out = NULL,  
    along.with = NULL, ...)
```

*Ponto inicial. Se não informado, o R começa em 1*

# FUNÇÕES REP() E SEQ()

seq {base}

```
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)), length.out = NULL,  
    along.with = NULL, ...)
```

*Ponto final. Se não informado, o R termina em 1*

# FUNÇÕES REP() E SEQ()

seq {base}

```
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)), length.out = NULL,  
    along.with = NULL, ...)
```



*Intervalos entre um ponto e outro.  
Se não informado, o R faz a amplitude dividida pelo comprimento.*



# FUNÇÕES REP() E SEQ()

seq {base}

```
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)), length.out = NULL,  
along.with = NULL, ...)
```

*Especifica um tamanho para a sequência  
(não pode ser usado junto com os argumentos by e along.with)*

# FUNÇÕES REP() E SEQ()

`seq {base}`

```
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)), length.out = NULL,  
along.with = NULL, ...)
```



*Utiliza um outro objeto e cria uma sequência de mesmo tamanho*

# SUJANDO AS MÃOS





# Use o comando rep(1:4)

# Use o comando rep(1:4, each=2) e depois rep(1:4, c(2,2,2,2)). Modifique algum dos números 2 para outro inteiro qualquer.

# Armazene no vetor x uma sequencia com vinte números quatro

# Crie um vetor z que tenha uma sequencia com 1 número um, 2 números dois, três números 3 e quatro números 4. (Dica: Concatene variáveis)

A computer monitor with a black bezel and a silver stand. The screen displays R code in green and red text. The code includes comments and function calls for the 'rep' function. The background of the screen is white with a faint, light blue geometric pattern.

```
# Use o comando rep(1:4)
```

```
rep(1:4)
```

```
# Use o comando rep(1:4, each=2) e depois rep(1:4, c(2,2,2,2)). Modifique  
algun dos números 2 para outro inteiro qualquer.
```

```
rep(1:4, each=2)
```

```
rep(1:4, c(2,2,2,2))
```

```
rep(1:4, c(2,4,2,4))
```

```
# Armazene no vetor x uma sequencia com vinte números quatro
```

```
x <- rep(4, 20)
```

```
# Crie um vetor z que tenha uma sequencia com 1 número um, 2 números  
dois, três números 3 e quatro números 4.
```

```
x <- rep(1,1)
```

```
y <- rep(2,2)
```

```
t <- rep(3,3)
```

```
w <- rep(4,4)
```

```
z <- c(x,y,t,w)
```



# Use o comando rep(1:4)

rep(1:4)

# Use o comando rep(1:4, each=2) e depois rep(1:4, c(2,2,2,2)). Modifique algum dos números 2 para outro inteiro qualquer.

rep(1:4, each=2)

rep(1:4, c(2,2,2,2))

rep(1:4, c(2,4,2,4))

# Armazene no vetor x uma sequencia com vinte números quatro

x <- rep(4, 20)

# Crie um vetor z que tenha uma sequencia com 1 número um, 2 números dois, três números 3 e quatro números 4.

x <- rep(1,1)

y <- rep(2,2)

t <- rep(3,3)

w <- rep(4,4)

z <- c(x,y,t,w)

Alternativa:

z <- rep(1:4, 1:4)

A computer monitor with a black bezel and a silver stand. The screen is white and displays green text. The text is organized into two sections: a single line at the top and a list of five lines below it.

# Utilize o comando rep para gerar a sequência de 3 a 6.

# Inclua as seguintes combinações de argumentos:

# each = 2 e len = 4

# each = 2 e len = 8

# each = 2 e len = 10

# each = 3 e times = 3

# each = 2 e times = 4

# Utilize o comando rep para gerar a sequência de 3 a 6.

rep(3:6)

# Inclua as seguintes combinações de argumentos:

# each = 2 e len = 4

rep(3:6, each = 2, len = 4)

# each = 2 e len = 8

rep(3:6, each = 2, len = 8)

# each = 2 e len = 10

rep(3:6, each = 2, len = 10)

# each = 2 e times = 3

rep(3:6, each = 2, times = 3)

# each = 3 e times = 2

rep(3:6, each = 3, times = 2)

# Crie uma sequência de 50 a 100, com “pulos” iguais a 4 usando seq()

# Crie uma sequência de 100 a 50, com “pulos” iguais a 4 (Dica: inverta o sinal do pulo)

# Crie uma variável x que tenha o vetor com as palavras “eu”, “adoro”, “programar”, “em” e “R”.

# Crie uma variável y usando o comando seq(along.with=x)

# Crie uma sequência de números de 100 a 1000 com exatamente 20 valores.

# Crie uma sequência de 50 a 100, com “pulos” iguais a 4  
`seq(50,100,4)`

# Crie uma sequência de 100 a 50, com “pulos” iguais a 4 (Dica: inverta o sinal do pulo)  
`seq(100,50,-4)`

# Crie uma variável x que tenha o vetor com as palavras “eu”, “adoro”, “programar”, “em” e “R”.  
`x<-c(“eu”, “adoro”, “programar”, “em”, “R”)`

# Crie uma variável y usando o comando seq(along.with=x)  
`y<- seq(along.with=x)`

# Crie uma sequência de números de 100 a 1000 com exatamente 20 valores.  
`seq(100, 1000, length.out=20)`



# AULA 1

- ✓ *Avisos gerais*
- ✓ *Programação: motivação e recursos online*
- ✓ *R: histórico, aplicações e recursos*
- ✓ *Primeiro contato*
- ✓ *Usando o Rstudio*
- ✓ *Sujando as mãos:*
  - *Operações aritméticas*
  - *Booleanos*
  - ***Vetores e Matrizes (cont.)*** ← “Guardando” valores em variáveis e tipos de variáveis
  - *Condicionais e Loops*
  - *Funções*



# POSIÇÕES DE OBJETOS EM VETORES

- Cada elemento de um vetor possui uma posição específica que pode ser acessada utilizando

`vetor[número]`

- Por exemplo, se `x <- 5:10` então `x[1]` retorna o número 5

```
# Calculando os gastos na festinha junina  
# Ganhos no jogo da pescaria, por dia da semana (seg a sex)  
pescaria <- c(14, -5, 20, -12, 24)
```

```
# Ganhos no jogo de argola, também por dia  
argola <- c(-2.4, -5, 10, -3.5, 1)
```

```
# Utilize o comando names() em cada vetor e atribua os nomes dos dias da  
semana, conforme o exemplo abaixo:
```

```
x <- c(1,0,2)  
names(x) <- c("Um", "Zero", "Dois")
```

```
# Calculando os gastos na festinha junina
# Ganhos no jogo da pescaria, por dia da semana (seg a sex)
pescaria <- c(14, -5, 20, -12, 24)
# Ganhos no jogo de argola, também por dia
argola <- c(-2.4, -5, 10, -3.5, 1)
# Utilize o comando names() em cada vetor e atribua os nomes dos dias da
semana, conforme o exemplo abaixo:
  x <- c(1,0,2)
  names(x) <- c("Um", "Zero", "Dois")
names(pescaria) <- c("Seg", "Ter", "Qua", "Qui", "Sex")
names(argola) <- c("Seg", "Ter", "Qua", "Qui", "Sex")
# Alternativa
dias_da_semana <- c("Seg", "Ter", "Qua", "Qui", "Sex")
names(pescaria) <- dias_da_semana
names(argola) <- dias_da_semana
```



# Calculando os gastos na festinha junina

# Imprima na tela os ganhos com pescaria da quarta-feira (dia 3)

# Imprima na tela dos ganhos da argola na segunda e quarta feira (dias 1 e 3)

# Imprima na tela os ganhos da pescaria de segunda a quinta (dias 1 a 4)



A computer monitor with a black bezel and a silver stand. The screen displays R code in green and red text. The code includes comments in green and function calls in red. The background of the screen is white with a faint, light blue geometric pattern.

```
# Calculando os gastos na festinha junina
```

```
# Imprima na tela os ganhos com pescaria da quarta-feira (dia 3)
```

```
pescaria[3]
```

```
# Imprima na tela dos ganhos da argola na segunda e quarta feira (dias 1 e 3)
```

```
argola[c(1,3)]
```

```
# Imprima na tela os ganhos da pescaria de segunda a quinta (dias 1 a 4)
```

```
pescaria[1:4]
```

```
OU
```

```
pescaria[-5]
```



# Repita os passos mas ao invés de usar os números, utilize o nome dos dias da semana (entre aspas)

# Imprima na tela os ganhos com pescaria da quarta-feira (dia 3)

# Imprima na tela dos ganhos da argola na segunda e quarta feira (dias 1 e 3)

# Imprima na tela os ganhos da pescaria de segunda a quinta (dias 1 a 4)

# Calculando os gastos na festinha junina

# Imprima na tela os ganhos com pescaria da quarta-feira (dia 3)

pescaria["Qua"]

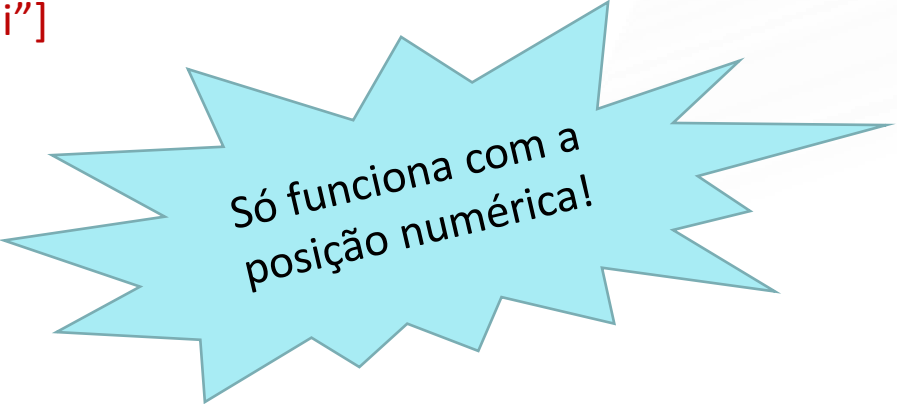
# Imprima na tela dos ganhos da argola na segunda e quarta feira (dias 1 e 3)

argola[c("Seg","Qua")]

# Imprima na tela os ganhos da pescaria de segunda a quinta (dias 1 a 4)

pescaria["Seg":"Qui"]

pescaria[-"Sex"]



Só funciona com a  
posição numérica!

*Dúvidas?*



