

Estatística Computacional com o R

Prof^a. Angélica Maria T. Ribeiro

Contents

	5
1 Conceitos Básicos	7
1.1 Sobre o R e RStudio	7
1.2 Instalação do R e do RStudio	8
1.3 Estrutura do RStudio	9
1.4 Operações Básicas em R	13

Chapter 1

Conceitos Básicos

1.1 Sobre o R e RStudio

- O R é uma linguagem de programação para análises estatísticas de dados
- É de código aberto, sendo gratuito e de livre distribuição;
- Amplamente utilizado por pesquisadores, professores e estudantes
- Disponível para diferentes sistemas operacionais;
- Conta com inumeros pacotes que disponibilizam funções e dados estatísticos
- Junto ao R, o RStudio é utilizado como ambiente de desenvolvimento integrado

Algumas aplicações de R:

- Construção de Livros:
 - Livro: R Cookbook
 - Livro: Mastering Shiny
 - Galeria do Bookdown
- Sites/páginas pessoais:
 - Minha Página Pessoal
 - Galeria do Blogdown
- Aplicativos dinâmicos:
 - Aplicativos LEG (UFPR)
 - Projeto de Extensão: EducaShiny
 - Galeria do Shiny

Referências/Fontes de ajuda

Livros

- R Cookbook
- R for Data Science

- The Book of R: A First Course in Programming and Statistics
- Hands-On Programming with R: Write Your Own Functions and Simulations
- Learning R: A Step-by-Step Function Guide to Data Analysis

Materiais em português

- Estatística Computacional com R (LEG/UFPR)
- Ciência de Dados em R
- Introdução ao Software R
- Introdução ao R
- Introdução ao R: Curso Básico de Linguagem R
- Introdução ao R (Vinícius A., Tania M. e Davi W.)

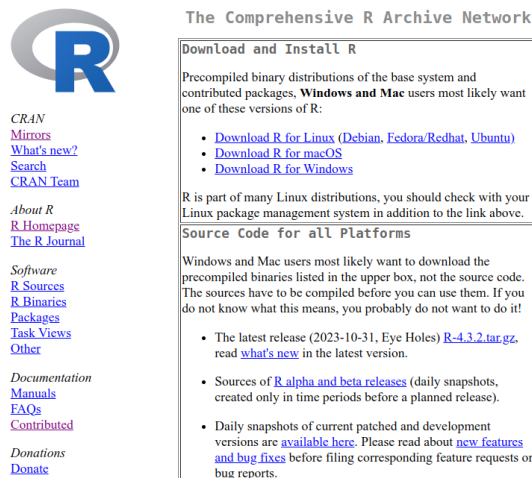
Páginas de ajuda

- StackOverflow
- R-bloggers

1.2 Instalação do R e do RStudio

Disponível para Windows, macOS e Linux a instalação do **R** é simples basta seguir o passo-a-passo:

- Visitar o site do **R**: <https://cran.r-project.org/>
- Clicar no link referente ao sistema operacional correspondente
- Fazer a instalação de acordo com as instruções



The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux \(Debian, Fedora/Redhat, Ubuntu\)](#)
- [Download R for macOS](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2023-10-31, Eye Holes) [R-4.3.2.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.

CRAN
[Mirrors](#)
[What's new?](#)
[Search](#)
[CRAN Team](#)

About R
[R Homepage](#)
[The R Journal](#)

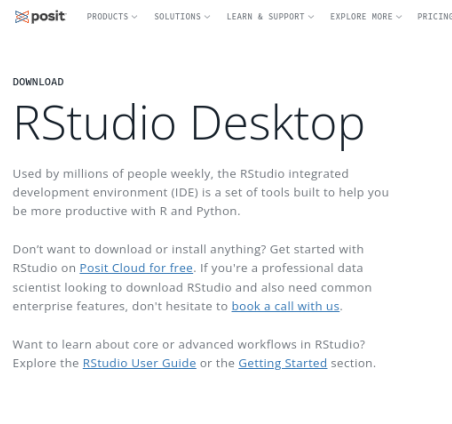
Software
[R Sources](#)
[R Binaries](#)
[Packages](#)
[Task Views](#)
[Other](#)

Documentation
[Manuals](#)
[FAQs](#)
[Contributed](#)

Donations
[Donate](#)

Após a instalação do **R** seguir o passo-a-passo para a instalação do **Rstudio**:

- Visitar a página do RStudio
- Fazer o download do arquivo referente ao sistema operacional correspondente
- Fazer a instalação de acordo com as instruções

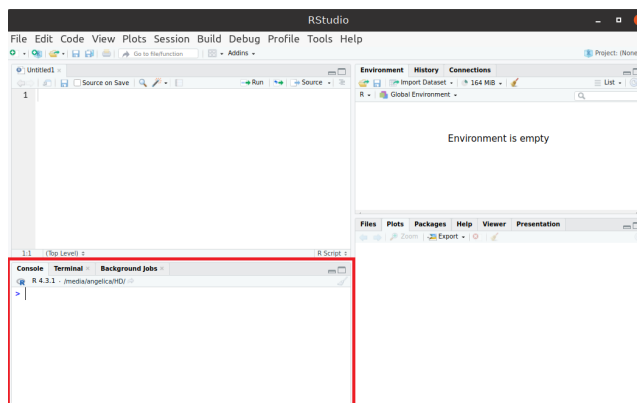


1.3 Estrutura do RStudio

O Rstudio é dividido em quatro janelas principais:

1. Console (Janela Inferior Esquerda):

- Permite executar comandos R diretamente.
- Exibe resultados e mensagens (saídas de funções, erros, avisos).



Dicas:

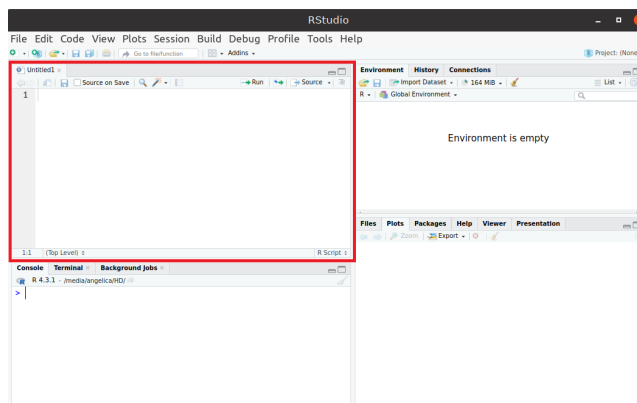
- Para rodar um código, digite e pressione **Enter**.
- Use as setas ↑ e ↓ para acessar comandos anteriores.
- O atalho **Ctrl + L** (Windows/Linux) ou **Cmd + L** (Mac) limpa o console.

Exemplo Digite os códigos abaixo no *Console*. Use as setas ↑ e ↓ para acessar comandos anteriores. Limpe o console.

```
x <- 3 # pressione Enter
y <- 2 # pressione Enter
print(x) # pressione Enter
print(y) # pressione Enter
```

2. Editor de Scripts (Janela Superior Esquerda):

- Editor de arquivos R (.R), RMarkdown (.Rmd), Shiny Apps, entre outros.
- Permite escrever e salvar códigos para execução posterior.



Dicas:

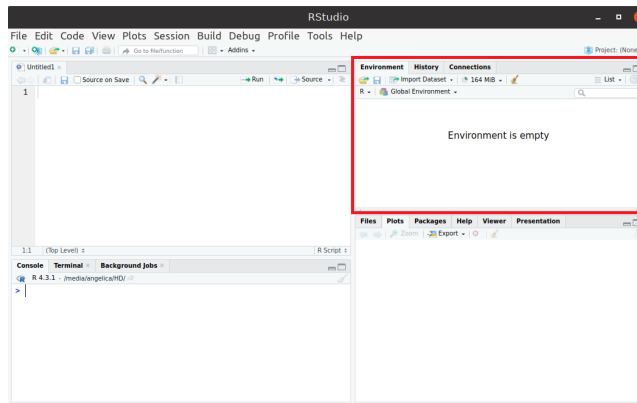
- **Ctrl + Enter** → Executa a linha atual.
- **Ctrl + Shift + Enter** ou **Ctrl + A** e **Ctrl + Enter** → Executa o script inteiro.
- **Ctrl + S** → Salva o arquivo.
- Para criar um novo script: **File** → **New File** → **R Script**.

Exemplo Digite os códigos abaixo no *Editor de Scripts*. Utilize **Ctrl + Enter** para compilar cada linha separadamente. Utilize **Ctrl + Shift + Enter** (ou **Ctrl + A** e **Ctrl + Enter**) para compilar todo o script. Salve o arquivo em algum repositório.

```
a <- -1
b <- 5
a
b
```

3. Ambiente e Histórico (Janela Superior Direita):

- **Environment:** Mostra objetos carregados na sessão (data frames, variáveis, funções).
- **History:** Lista de comandos executados no console.
- **Connections:** Permite conectar-se a bancos de dados.



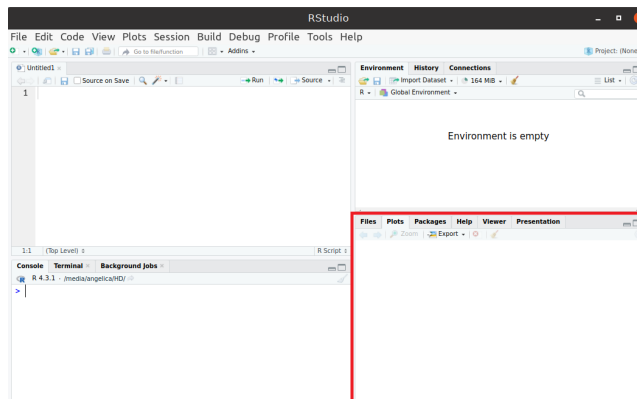
Dicas:

- Use `ls()` no console para listar os objetos carregados.
- Para limpar o ambiente: `rm(list = ls())`

Exemplo Consulte as abas *Ambiente* e *Histórico*. Visualize as variáveis criadas e funções utilizadas. Utilize `rm(list = ls())` para limpar o ambiente.

4. Janela de Arquivos, Plots, Pacotes e Ajuda (Janela Inferior Direita):

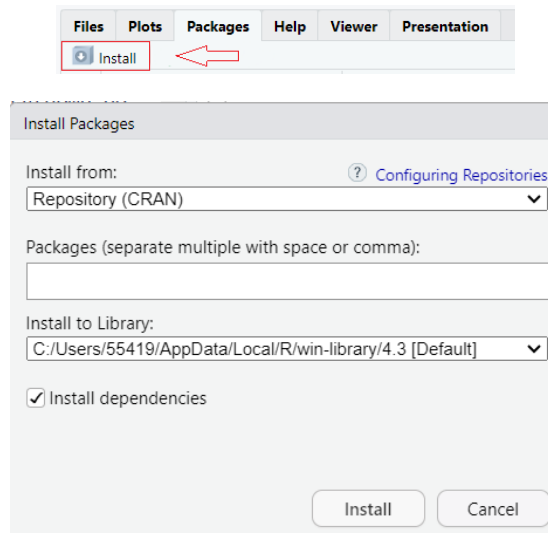
- **Files:** Gerenciador de arquivos do diretório de trabalho.
- **Plots:** Exibe gráficos gerados pelo R.
- **Packages:** Lista pacotes instalados e permite carregar ou instalar novos.
- **Help:** Ajuda e documentação de funções do R.
- **Viewer:** Exibe documentos HTML, visualizações interativas (como Shiny).
- **Presentation:** Permite criar apresentações de slides (RMarkdown).



Dicas:

- Instalar um pacote: `install.packages("nome_do_pacote")` ou clicar no botão **Install** na aba *Packages*.

- Carregar um pacote: `library("nome_do_pacote")` ou “marcar” o pacote na aba *Packages*.
- Visualizar ajuda de uma função (ou pacote): `?nome_da_função` ou `help(nome_da_função)`.
- Listar pacotes carregados: `search()`



Exemplo

- Copie e cole o código `plot(1:10)`. Visualize o resultado na aba *Plots*. Exporte o gráfico.
- Instale e carregue o pacote `ggplot2`. Encontre a documentação do pacote no cran do R.
- Consulte na aba *help* a documentação da função `mean()`. Qual argumento permite ignorar valores NA nos cálculos de média?

1.3 Exercícios da Sessão

1. No **Console**, execute os seguintes comandos e observe os resultados:

```
x <- 10
y <- 5
soma <- x + y
soma
```

Qual foi o valor impresso no console? Use o comando `ls()` no console, o que ele faz?

2. No **Editor de Scripts**, crie um novo arquivo R: **File** → **New File** → **R Script**. Copie e cole o seguinte código no script:

```
# Criando um vetor
numeros <- c(1, 2, 3, 4, 5)
# Calculando a média
media <- mean(numeros)
# Mostrando o resultado
print(media)
```

- Salve o script como meuscript.R
- Execute o script inteiro usando **Ctrl + A** e **Ctrl + Enter**.
- O que foi impresso no console?

3. No **Editor de Scripts** cole este código e execute:

```
x <- seq(-10, 10, 0.1)
y <- sin(x)
plot(x, y, type = "l", col = "blue", main = "Gráfico de Seno")
```

- Vá até a aba **Plots** e veja o gráfico gerado.
- Clique em **Export** para salvar o gráfico.

4. Na aba *Packages*, instale e carregue o pacote de manipulação de dados **dplyr**. Encontre a documentação do pacote no *Cran* do R.

5. No *Console* do RStudio, consulte a documentação das seguintes funções:

```
?mean
?sd
?sum
?seq
?sample
```

- Descreva resumidamente o que faz cada função.
- Anote os principais **argumentos** de cada função.

1.4 Operações Básicas em R

1.4.1 Operações Aritméticas

R funciona como uma calculadora, permitindo operações básicas com números e variáveis.

Exemplo

```
# Soma
2 + 3 # Resultado: 5
# Subtração
10 - 4 # Resultado: 6
# Multiplicação
```

```

5 * 2 # Resultado: 10
# Divisão
9 / 3 # Resultado: 3
# Exponenciação
2^3 # Resultado: 8
# Raiz quadrada
sqrt(16) # Resultado: 4
# Resto da divisão
10 %% 3 # Resultado: 1
# Parte inteira da divisão
10 %/% 3 # Resultado: 3

```

Ordem de Prioridade dos Operadores Aritméticos

- Parênteses () – maior prioridade
- Exponenciação ^
- Multiplicação e Divisão *, /
- Soma e Subtração +, -

Exemplo

```

2 + 3 * 4 # Multiplicação ocorre primeiro -> 14
(2 + 3) * 4 # Parênteses ocorre primeiro -> 20
2 * 5 ^ 2 # Exponenciação ocorre primeiro -> 25
(2 * 5) ^ 2 # Parênteses ocorre primeiro -> 100
5 * 3 / 2 # Ordem indiferente
5 + 3 - 2 # Ordem indiferente

```

1.4.2 Criando Variáveis (objetos)

Para criar uma variável ou objeto, usamos o símbolo de atribuição <-

```

## Atribuindo um número a um objeto:
x <- 5; y <- 3

```

Operações entre objetos numéricos:

```

x+y # soma
x-y # subtração
x*y # produto
x/y # quociente

```

Restrições nos nomes das variáveis (objetos)

1. Começo do Nome: Não pode começar com número ou caracteres especiais. Deve começar com uma letra (A-Z ou a-z).

```
nome_valido <- 5 # Correto
1nome_invalido <- 15 # Erro: nome não pode começar com número
@nome_invalido <- 15 # Erro: nome não pode começar com caracteres especiais
```

2. Espaços e Caracteres: O nome não deve conter espaços nem caracteres especiais como @, #, -, \$, etc. O underline (__) é um caractere permitido no meio do nome.

```
variavel_valida_1 <- 20 # Correto
variavel#invalida <- 40 # Erro: caractere inválido (#)
variavel@invalida <- 40 # Erro: caractere inválido (@)
```

3. Diferenciação de Maiúsculas e Minúsculas: O R diferencia maiúsculas de minúsculas. Assim, var, Var e VAR são considerados nomes diferentes.

```
var <- 100
Var <- 200
print(var) # 100
print(Var) # 200
```

4. Palavras Reservadas: Não é permitido usar palavras reservadas do R (como if, else, for, TRUE, FALSE, etc.). Essas palavras têm significados específicos e não podem ser sobrescritas.

```
for <- 10 # Erro: 'for' é uma palavra reservada
if <- 20 # Erro: 'if' é uma palavra reservada
```

5. Pontos e Nomes de funções preexistentes: Não é recomendado utilizar pontos. Também evitar usar nomes que conflitem com funções **preexistentes** do R.

```
## Evitar
nome.com.ponto <- 100 # Correto, mas pode ser confuso
mean <- 2
mean(mean)
```

Dicas para os nomes das variáveis (objetos)

- A convenção em R é usar underline (__) para nomes de objetos (exceto no início).
- Evitar nomes ambíguos e usar nomes descritivos que deixem claro o que o objeto representa.

```
## Exemplos de Nomes Válidos
x_1 <- 5
minha_variavel <- 10
temperatura_curitiba <- 20
altura_estudantes <- 160
```

Exercícios

1. Realize as seguintes operações no R:

- Soma de 45 e 23.
- Subtração de 78 por 35.
- Multiplicação de 12 e 9.
- Divisão de 100 por 4.
- Calcule 5 elevado à potência 3.
- Calcule a raiz quadrada de 64.

2. Atribua os valores 10 e 5 a duas variáveis chamadas **a** e **b**, respectivamente. Em seguida, calcule:

- A soma de **a** e **b**.
- O produto de **a** e **b**.
- A diferença entre **a** e **b**.
- O quociente de **a** dividido por **b**.

3. Algumas regras devem ser seguidas ao nomear variáveis em R. Analise os seguintes nomes e indique quais são válidos e quais causarão erro. Justifique a sua resposta.

```
1numero <- 100
meu_numero <- 25
data <- "2025-03-15"
for <- "teste"
x@y <- 50
x_y <- 5
```

Respostas

3.

```
1numero <- 100 # Inválido: não pode começar com número
meu_numero <- 25 # Válido
data <- "2025-03-15" # Válido, mas desaconselhado (conflito com função data())
for <- "teste" # Inválido: "for" é palavra reservada
x@y <- 50 # Inválido: caractere inválido (@)
x_y <- 5 # Válido
```

Acesso Professor

1.4.3 Operações Lógicas e Comparações

Os operadores lógicos em R são usados para avaliar expressões booleanas (VERDADEIRO ou FALSO) e criar condições. Eles operam em vetores, valores lógicos (TRUE, FALSE) e numéricos e são muito usados em estruturas condicionais e filtros de dados.

1. Operadores de Comparação

Estes operadores retornam `TRUE` ou `FALSE`, dependendo do resultado da comparação.

Operador	Descrição	Exemplo	Resultado
<code>==</code>	Igualdade	<code>10 == 10</code>	<code>TRUE</code>
<code>!=</code>	Diferente	<code>10 != 5</code>	<code>TRUE</code>
<code>></code>	Maior que	<code>10 > 5</code>	<code>TRUE</code>
<code><</code>	Menor que	<code>10 < 5</code>	<code>FALSE</code>
<code>>=</code>	Maior ou igual	<code>10 >= 10</code>	<code>TRUE</code>
<code><=</code>	Menor ou igual	<code>10 <= 5</code>	<code>FALSE</code>

Obs.: Para consultar o *help* destas funções usar `?Comparison` ou `help("operador")`, para algum operador específico.

2. Operadores Lógicos

Operador	Descrição	Exemplo	Resultado
<code>!</code>	Negação lógica	<code>!TRUE</code>	<code>FALSE</code>
<code>&</code>	E lógico	<code>(TRUE & FALSE)</code>	<code>FALSE</code>
<code> </code>	OU lógico	<code>(TRUE FALSE)</code>	<code>TRUE</code>
<code>&&</code>	E lógico	<code>(TRUE && FALSE)</code>	<code>FALSE</code>
<code> </code>	OU lógico	<code>(TRUE FALSE)</code>	<code>TRUE</code>

Obs.: Para consultar o *help* destas funções usar `?Logic` ou `help("operador")`, para algum operador específico.

Diferença entre `&` e `&&`, `|` e `||`

- As formas mais curtas (`&` e `|`) realizam comparações elemento a elemento.
- As formas mais longas (`&&` e `||`) avaliam da esquerda para a direita, prosseguindo apenas até que o resultado seja determinado (em geral, mais eficiente computacionalmente).

Exemplos

```
## Operadores de Comparação
5 > 3
5 != 3

## Operadores Lógicos
TRUE & TRUE   # ambos são verdadeiros? Sim -> TRUE, Não -> FALSE
TRUE | FALSE  # pelo menos um deles é verdadeiro? Sim -> TRUE, Não -> FALSE
## Operadores de Comparação e Lógicos
```

```
(5>3) & (4>6) # FALSE  
(2<3) | (4<6) # TRUE
```

Exercícios

1. Considere os seguintes números: `a <- 7`, `b <- 3`. No R, verifique as condições:

- Se `a` é maior que `b`
- Se `a` é divisível por 2
- Se `a` é maior que `b` E (lógico) se `a` é divisível por 2, usando o operador lógico `&&`

2. Identifique, primeiramente **sem o uso do R**, o que retornará as relações a seguir (TRUE ou FALSE). Na sequência, confira os resultados usando o R.

- `10 > 5`
- `5 == 5`
- `5 != 3`
- `(3 + 2) >= 5`
- `10 / 2 < 6`
- `(4 > 2) & (10 <= 20)`
- `(7 < 3) | (8 == 8)`
- `!TRUE`
- `!(5 > 3)`

Respostas

1.

```
a <- 7  
b <- 3  
- a>b  
- (a%%2==0)  
- a>b && (a%%2==0)
```

2.

```
10 > 5
```

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

```
5 == 5
```

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

```
5 != 3
```

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

```
(3 + 2) >= 5
```

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

```
10 / 2 < 6
```

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

```
(4 > 2) & (10 <= 20)
```

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

```
(7 < 3) | (8 == 8)
```

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

```
!TRUE
```

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

```
!(5 > 3)
```

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

Acesso Professor

1.4 Exercícios da Sessão

1. Resolva as expressões abaixo (primeiramente sem o R) e explique o resultado com base na ordem de prioridade dos operadores. Na sequência, confira os resultados usando o R.

```
# a)
```

```
resultado1 <- 5 + 3 * 2
```

```
# b)
```

```
resultado2 <- (5 + 3) * 2
```

```
# c)
```

```
resultado3 <- 10 / 2 + 3
```

```
# d)
```

```
resultado4 <- 10 / (2 + 3)
```

```
# e)
```

```
resultado5 <- 4^2 / 2
```

2. Dados os valores:

```
x <- 15; y <- 10; z <- 20
```

Quais das relações a seguir serão TRUE e quais serão FALSE:

```
# a)
x > y & y < z
# b)
x == 15 | y > z
# c)
!(x < z)
# d)
(x >= y) & (y != 10)
# e)
(x < y) | (z > x)
```

Respostas

1.

```
# a)
(resultado1 <- 5 + 3 * 2)
```

```
## [1] 11
```

```
# b)
(resultado2 <- (5 + 3) * 2)
```

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

```
# c)
(resultado3 <- 10 / 2 + 3)
```

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

```
# d)
(resultado4 <- 10 / (2 + 3))
```

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

```
# e)
(resultado5 <- 4^2 / 2)
```

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

2.

```
x <- 15
y <- 10
z <- 20
# a)
x > y & y < z
```

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

```
# b)
x == 15 | y > z
```

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

```
# c)
!(x < z)
```

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

```
# d)
(x >= y) & (y != 10)
```

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

```
# e)
(x < y) | (z > x)
```

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

Acesso Professor