Lorrainy de Sousa Santos

Lista de exercícios para prática de R

# Introdução à Análise de Dados - FACE/UFMG

Robert Iquiapaza 2024-03-26

Explorando o Ambiente R:

1. Utilize o comando help(nome\_da\_funcao) para obter ajuda sobre uma função específica.

Utilizei o comando help(“sin’’), resultado:

- Na aba **console** > help("sin")

- Na aba **help**:

“

Trigonometric Functions

Description

These functions give the obvious trigonometric functions. They respectively compute the cosine, sine, tangent, arc-cosine, arc-sine, arc-tangent, and the two-argument arc-tangent.

cospi(x), sinpi(x), and tanpi(x), compute cos(pi\*x), sin(pi\*x), and tan(pi\*x).

Usage

cos(x)

sin(x)

tan(x)

acos(x)

asin(x)

atan(x)

atan2(y, x)

cospi(x)

sinpi(x)

tanpi(x)

Arguments

x, y numeric or complex vectors.

[...]”

1. Utilize o comando ?nome\_da\_funcao para obter uma visão rápida da função.

- Na aba **console**: > ?sqrt

- Na aba help:

“

Miscellaneous Mathematical Functions

Description

abs(x) computes the absolute value of x, sqrt(x) computes the (principal) square root of x, √x​

The naming follows the standard for computer languages such as C or Fortran.

Usage

abs(x)

sqrt(x)

Arguments

X a numeric or complex vector or array.

[...] “

1. Crie um objeto do tipo numeric com o comando c(1, 2, 3, 4, 5). Nome Teste1

> nome.teste1 <- c(1, 2, 3, 4, 5)

1. Crie um objeto do tipo character com o comando c("a", "b", "c", "d", "e"). Nome Teste2

> nome.teste2 <- c("a", "b", "c", "d", "e")

1. Qual a diferença entre os códigos abaixo?

# Código 1

30 / 9

O código 1 representa apenas uma divisão simples que o R irá fazer e apresentar o resultado.

# Código 2

divi <- 30 / 9

O código 2 é diferente pois está sendo armazenado e dado o nome de “divi”, portanto quando for querer utilizado de novo, em alguma conta ou função será apenas necessário apresentar o nome “divi”.

1. Por que o nome minha-var não pode ser utilizado para criar um objeto? O que significa a mensagem de erro resultante?

minha-var <- 54/5

(minha-var) não pode ser usado para criar um objeto, porque se encontra com o erro em sua sintaxe, não pode utilizar caracteres especiais como “-” no meio das palavras. A mensagem erro: “Erro: objeto 'minha' não encontrado”, significa que o R não encontrou o objeto minha, isso porquê ele não foi definido anteriormente, visto que há um erro na digitação, o R está lendo como se ele tivesse sido definido anteriormente.

1. Se definir minha e var, o que significa a mensagem de erro resultante?‘

minha = 45

var= 20

minha-var <- 54/5

O erro: Error in minha - var <- 54/5 : não foi possível encontrar a função "-<-" Acontece porque já foi definido um valor para (minha) e (var) querer utilizar as duas palavras em conjunto, ainda com o erro de digitação, não irá funcionar, já que (minha) e (var) serão objetos independentes e agora só podem ser usados em funções e outros cálculos

1. Explique o que acontece agora

minha = 45

var= 20

minha-var -> minha\_var

Com a modificação na ultima linha foi feito uma subtração dos dados definidos anteriormente e em seguida atribuidos a minha\_var, portanto, a lingugem R entendeu que era a subtração de VAR de MINHA, resultando em 25 no valor final.

1. Utilize o comando ls() para verificar os objetos disponíveis no ambiente. Use rm() para remover um objeto da memória.

**Utilizando is():**

-base:

nome.teste1 <- c(1, 2, 3, 4, 5);

nome.teste2 <- c("a", "b", "c", "d", "e"); e

minha = 45

var=20

minha-var->minha\_var

Resultado:

> is.numeric(nome.teste1)

1. TRUE

> is.numeric(nome.teste2)

1. FALSE

> is.numeric(minha\_var)

[1] TRUE

Verificando characters:

Resultado:

|  |
| --- |
| > is.character(nome.teste1)   1. FALSE   > is.character(nome.teste2)   1. TRUE   > is.character(minha\_var)  [1] FALSE |

**Utilizando rm():**

Resultado:

> rm(nome.teste1)

> rm(nome.teste2)

> rm(minha\_var)

Confirmando:

is(nome.teste1)

Resultado:

Erro: objeto 'nome.teste1' não encontrado

**Operações Básicas:**

1. Utilize a função paste para concatenar textos. Procure utilizar diferentes valores do parâmetro sep de acordo com a ajuda da função.

lugar = "stackoverflow"

paste("Pedir","ajuda", "no", lugar , sep="\_")

Resultados:

> lugar= "UFMG!"

> paste("Que", "disciplina", "díficil", lugar , sep = " " )

[1] "Que disciplina difícil UFMG!"

1. Defina dois objetos de texto (Cidade e Estado) com Belo Horizonte e MG, respectivamente. Qual código permite combinar os mesmos objetos para produzir Belo Horizonte - MG sem utilizar o argumento sep?

Resultado:

> cidade<- "Belo Horizonte"

> estado<- "MG">

> paste(cidade, "-", estado)

[1] "Belo Horizonte - MG"

Portanto, a função que pode ser usado é paste(), que precisa especificar o sep.

1. Crie dois vetores numéricos x e y, com 5 elementos cada. Utilize os operadores aritméticos (+, -, \*, /) para realizar cálculos.

Resultado:

> x<- c(1:5)

> y<- c(20:25)

> x+y

[1] 21 23 25 27 29 26

> y-x

[1] 19 19 19 19 19 24

> y\*x

1. 20 42 66 92 120 25

> x/y

[1] 0.0500000 0.0952381 0.1363636 0.1739130 0.2083333

[6] 0.0400000

1. Crie dois vetores lógicos a e b, com 3 elementos cada. Utilize os operadores lógicos (&,|, !) para realizar comparações lógicas.

**Resultado:**

> a<- c(7:9)

> b<- c(0:2)

> a&b

[1] FALSE TRUE TRUE

> a|b

[1] TRUE TRUE TRUE

> a!=b

[1] TRUE TRUE TRUE

1. Utilize o comando ifelse() para realizar decisões condicionais, use os vetores dos dois exercícios anteriores.

# Exemplo

n=c(59,60); m= c(70,45)

ifelse(n>=60,"Aprova", "Reprova"); ifelse(m>=60,"Aprova", "Reprova")

Resultado:

> ifelse(a<=8, "Aprova", "Reprova");ifelse(b>=1, "Aprova", "reprova")

[1] "Aprova" "Aprova" "Reprova"

[1] "reprova" "Aprova" "Aprova"

**Vetores e Matrizes:**

1. Crie vetores com diferentes tipos de dados (números, texto, lógicos).

> num<- c(15, 01, 20, 05)

> tex<- c("a", "n" ,"i", "v", "e", "r", "s", "a", "r", "i", "o")

> log<- c("FALSE", "FALSE", "TRUE", "TRUE")

1. Combine vetores com o comando c(). O que acontece com os tipos de dados nas diferentes combinações? Pode usar a função class().

> comb<- c(15, "n", "TRUE", 05)

> comb2<- c(tex, log)

> comb3<- c(num,log)

> class(comb)

1. "character"

> class(num)

1. "numeric"

> class(tex)

1. "character"

> class(comb2)

1. "character"

> class(comb3)

1. "character"

Muitos deles que antes seriam considerados numericos, se transformam em character.

1. Acesse elementos de um vetor com o operador []. Exemplos, uma posição específica, várias posições específicas continuas, mais de uma posição em diferentes partes do vetor.

> ac<- num[2]

> ac2<- tex[3:6]

> ac3<- log[c(1, 4)]

> print(c(ac, ac2, ac3))

[1] "1" "i" "v" "e" "r" "FALSE"

[7] "TRUE"

1. Dado o vetor booleano a seguir de resultados diários da B3. Quantos dias a bolsa subiu? Qual a proporção de dias em se produziu uma subida na bolsa? Use sum() e mean().

bolsa\_subiu <- c(TRUE, TRUE, FALSE, FALSE, TRUE, FALSE, TRUE, FALSE)

bolsa\_subiu<- c(TRUE,TRUE,FALSE,FALSE,TRUE,FALSE,TRUE,FALSE)

> dia\_bolsa\_subiu<- sum(bolsa\_subiu)

> prop\_bolsa\_subiu<- mean(bolsa\_subiu)

> print(dia\_bolsa\_subiu)

1. 4

> print(prop\_bolsa\_subiu)

[1] 0.5

1. O código abaixo vai guardar três números inteiros entre 0 e 10. Determine se: são números maiores do que 5? são menores do que 4? São números pares?

segredo <- round(runif(3, min = 0, max = 10))

> ifelse(segredo<4, TRUE, FALSE)

1. FALSE FALSE FALSE

> ifelse(segredo>5, TRUE, FALSE)

1. TRUE TRUE TRUE

> any(segredo%%2==0)

[1] TRUE

Todos são maiores de 5, nenhum é menor que 4 e todos são números pares.

1. Crie o código para descobrir os números guardados no vetor segredo.]

> print(segredo)[1] 6 8 10

1. Crie matrizes com o comando matrix().

|  |
| --- |
| > matrix(data = 1:25, nrow = 5, ncol = 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 |

1. Acesse elementos de uma matriz com os operadores [ e ,. Exemplos, linha 2 e coluna 2, a coluna 1 completa, a linha 2 completa, duas colunas, duas ou mais linhas.

> print(matrix1[2, 5])

1. 22

> coluna<- matrix1[, 4]

> print(coluna)

1. 16 17 18 19 20

> coluna2<- matrix1[, c(5, 3)]

> print(coluna2)

[,1] [,2]

[1,] 21 11

[2,] 22 12

[3,] 23 13

[4,] 24 14

[5,] 25 15

> linha1<- matrix1[2, ]

> print(linha1)

[1] 2 7 12 17 22

**Vetores e Operações Básicas:**

1. O seguinte código cria dois vetores de 55 e 40 elementos

vetor1 <- runif(55, 12, 40)

vetor2 <- rnorm(40, 25, 8)

1. Utilize a função mean() para calcular a média dos vetores.

> mean(vetor1)

1. 25.61579

> mean(vetor2)

[1] 27.17167

1. Utilize a função median() para calcular a mediana dos vetores.

> median(vetor1)

1. 26.63047

> median(vetor2)

[1] 27.10561

1. Utilize a função sd() para calcular o desvio padrão dos vetores.

> sd(vetor1)

1. 7.907702

> sd(vetor2)

[1] 6.960462

1. Crie um vetor nomeado (“Media”,“Mediana”,“Desvio.padrão”) com as 3 estatísticas an- teriores

> media\_1<- mean(vetor1)

> media\_2<- mean(vetor2)

> média<- c(media\_1, media\_2)

> print(média)

1. 25.61579 27.17167

> mediana\_1<- median(vetor1)

> mediana\_2<- median(vetor2)

> mediana<- c(mediana\_1, mediana\_2)

> print(mediana)

1. 26.63047 27.10561

> dp1<- sd(vetor1)

> dp2<- sd(vetor2)

> desvio.padrão<- c(dp1, dp2)

> print(desvio.padrão)

[1] 7.907702 6.960462

1. Utilize a função summary() para obter um resumo estatístico dos vetores. Compare com os resultados do exercício anterior.

> summary(média, mediana, desvio.padrão)

Min. 1st Qu. Median Mean 3rd Qu. Max.

25.62 26.00 26.39 26.39 26.78 27.17

1. Altere a posição 2 do primeiro vetor para NA (dado ausente), e aplique as funções estatísticas anteriores. Consulte a ajuda para descobrir como ignorar os dados ausentes, e realizar os cálculos corretamente.

vetor1[2] <- NA

vetor1[2]<- NA

> print(média)

NA 23.56523

> print(mediana)

1. NA 24.06875

> print(desvio.padrão)

1. NA 7.971398

> summary(média, mediana, desvio.padrão)

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

23.57 23.57 23.57 23.57 23.57 23.57 1

Procurando ajuda encontrei que o parâmetro “na.rm=TRUE” remove valores ausentes o que faz os cálculos serem feitos corretamente.

Portanto, utilizando esse parameto é percebido que:

> #média

> media\_1<- mean(vetor1, na.rm = TRUE)

> media\_2<- mean(vetor2)

> média<- c(media\_1, media\_2)

> print(média)

1. 25.45510 23.56523

> #mediana

> mediana\_1<- median(vetor1, na.rm = TRUE)

> mediana\_2<- median(vetor2)

> mediana<- c(mediana\_1, mediana\_2)

> print(mediana)

1. 24.54264 24.06875

> #desvio padrão

> dp1<- sd(vetor1, na.rm = TRUE)

> dp2<- sd(vetor2)

> desvio.padrão<- c(dp1, dp2)

> print(desvio.padrão)

1. 8.507208 7.971398

> #resumo estático dos vetores

> summary(média, mediana, desvio.padrão)

Min. 1st Qu. Median Mean 3rd Qu. Max.

23.57 24.04 24.51 24.51 24.98 25.46

1. Crie um vetor vazio e adicione 7 valores numéricos entre 10 e 50 a ele.

> vetor\_teste<- numeric(0)

> for (i in 1:7) { + valor<- sample(10:50, 7)+ + }

> print(valor)

[1] 12 41 16 15 49 21 34

1. Encontre a soma, a média e o produto do vetor anterior.

> sum(valor)

1. 188

> mean(valor)

[1] 26.85714

> prod(valor)

[1] 4.131.146,880

1. Conte o número de valores dentro de um intervalo específico em um vetor, por exemplo entre 15 e 40.

> valor1<- sample(10:50)

> inicio <- 15

> fim<- 40

> num\_valores\_intervalo<- sum(valor1 >= inicio & valor1 <= fim)

> print(num\_valores\_intervalo)

[1] 25

1. Explique os resultados dos seguintes códigos:

# caso 1

function <- 10

Não tem como utilizar a palavra function em casos de atribuir valor, é considerado um erro de sintaxe ou estrutura

# caso 2

mean(5, 10)

Ele devolve o número 5, por entender que essa seria a médica, neste caso deveria ser usado o paramentro c() para que se fosse uma combinação, a resposta viesse de forma correta.

# caso 3

sum(2, 7)

Ele devolve a soma desses dois números.

**Manipulação de Listas:**

1. Crie uma lista com cinco posições onde em cada uma contém valores aleatórios com diferente número de elementos.

> (lista\_familia<- list(nome= "Deise", idade= 14, cidade= "Rio de janeiro", simbolo= "@", quer= "médicina" ))

$nome

[1] "Deise"

$idade

[1] 14

$cidade

[1] "Rio de janeiro"

$simbolo

[1] "@"

$quer

[1] "médicina"

1. Conte o número de elementos da lista anterior. Quantos elementos tem a posição (chave) 2?

> length(lista\_familia[3])

[1] 1

1. Adicione um par de elementos com nomes ou chaves a uma lista existente.

> (lista\_familia<- c(lista\_familia, cabelo= "castanho"))

$nome

[1] "Deise"

$idade

[1] 14

$cidade

[1] "Rio de janeiro"

$simbolo

[1] "@"

$quer

[1] "médicina"

$cabelo

[1] "castanho"

**Trabalhando com Strings:**

1. Converter uma string de caracteres em um nome de variável, use a função assign(). Atribua o vetor c(24,35,65) e essa variável e mostre a média da mesma.

> assign("Kaua", c(24, 35, 65))

> print(Kaua)

1. 24 35 65

> median(Kaua)

[1] 35

1. Conte o número de caracteres em uma string (nchar()).

> (nchar(Kaua))

[1] 2 2 2

> (nchar("Kaua"))

[1] 4

1. Crie um vetor de números e textos e caracteres. Qual a classe do objeto?

> kaua <- c(21, "carioca")

> class(kaua)

[1] "character"

1. No vetor do exercício anterior, converta o vetor de strings de caracteres para numérico e identifique as posições onde existem números. Filtre os números do vetor de strings de caracteres para um novo vetor que contem somente números. Dica: consulte a ajuda de as.numeric() e is.na().

> kaua <- c(21, "carioca", 35, "rio", 42)

> kaua\_numeric <- as.numeric(kaua)

> posicoes\_numeros <- !is.na(kaua\_numeric)

> numeros\_filtrados <- kaua\_numeric[posicoes\_numeros]

> print(kaua)

1. "21" "carioca" "35" "rio" "42"

> print(kaua\_numeric)

1. 21 NA 35 NA 42

> print(posicoes\_numeros)

1. TRUE FALSE TRUE FALSE TRUE

> print(numeros\_filtrados)

[1] 21 35 42

1. Analise o código seguinte e faça as alterações prévias necessárias para que não produza um erro. Qual o tamanho do vetor resultante?.

# definição de ...

paste("eu gosto de", frutas)

A função paste concatena strings ou ou outro elemento, portanto, para dar certo precisaria ser:

> paste("eu", "gosto", "de", "frutas")

[1] "eu gosto de frutas"

Sendo o seu tamanho igual a 1.

**DataFrames:**

1. Crie um DataFrame de 10 linhas a partir de vetores específicos de diferentes tipos de dados (textos, números, valores lógicos, fatores).

textos <- c("texto1", "texto2", "texto3", "texto4", "texto5", "texto6", "texto7", "texto8", "texto9", "texto10")

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

logicos <- c(TRUE, FALSE, TRUE, FALSE, TRUE, FALSE, TRUE, FALSE, TRUE, FALSE)

fatores <- factor(c("A", "B", "C", "A", "B", "C", "A", "B", "C", "A"))

data\_frame <- data.frame(Texto = textos, Numero = numeros, Logico = logicos, Fator = fatores)

print(data\_frame)

Texto Numero Logico Fator

1 texto1 1 TRUE A

2 texto2 2 FALSE B

3 texto3 3 TRUE C

4 texto4 4 FALSE A

5 texto5 5 TRUE B

6 texto6 6 FALSE C

7 texto7 7 TRUE A

8 texto8 8 FALSE B

9 texto9 9 TRUE C

10 texto10 10 FALSE A

1. Acesse e modifique os elementos do DataFrame: troque um valor numérico, altere uma condição de TRUE (Verdadeiro) para FALSE (Falso).

> data\_frame[1, 2] <- 20

> data\_frame[2, 3] <- FALSE

> print(data\_frame)

Texto Numero Logico Fator

1 texto1 20 TRUE A

2 texto2 2 FALSE B

3 texto3 3 TRUE C

4 texto4 4 FALSE A

5 texto5 5 TRUE B

6 texto6 6 FALSE C

7 texto7 7 TRUE A

8 texto8 8 FALSE B

9 texto9 9 TRUE C

10 texto10 10 FALSE A

1. Filtre o DataFrame com base em condições específicas com as colunas de números, a coluna de valores lógicos, e a coluna de fatores.

> filtro\_numeros <- subset(data\_frame, Numero > 5)

> filtro\_logicos <- subset(data\_frame, Logico == TRUE)

> filtro\_fatores <- subset(data\_frame, Fator == "B")

> print(filtro\_numeros)

Texto Numero Logico Fator

1 texto1 20 TRUE A

6 texto6 6 FALSE C

7 texto7 7 TRUE A

8 texto8 8 FALSE B

9 texto9 9 TRUE C

10 texto10 10 FALSE A

> print(filtro\_logicos)

Texto Numero Logico Fator

1 texto1 20 TRUE A

3 texto3 3 TRUE C

5 texto5 5 TRUE B

7 texto7 7 TRUE A

9 texto9 9 TRUE C

> print(filtro\_fatores)

Texto Numero Logico Fator

2 texto2 2 FALSE B

5 texto5 5 TRUE B

8 texto8 8 FALSE B

1. Utilize a função summary() para obter um resumo estatístico do DataFrame. Analise as informações apresentadas.

> summary(data\_frame)

Texto Numero Logico

Length:10 Min. : 2.00 Mode :logical

Class :character 1st Qu.: 4.25 FALSE:5

Mode :character Median : 6.50 TRUE :5

Mean : 7.40

3rd Qu.: 8.75

Max. :20.00

Fator

A:4

B:3

C:3

Análise dessas informações pode fornecer uma visão geral dos dados em cada variável, incluindo tendências centrais, dispersão e possíveis valores extremos. Por exemplo, para variáveis numéricas, você pode observar a média, mediana e desvio padrão, enquanto para variáveis categóricas (como fatores), você pode observar a frequência de cada nível.

1. Calcule a média da coluna numérica para cada nível da coluna fator.

> medias\_por\_nivel <- aggregate(data\_frame$Numero, by = list(data\_frame$Fator), FUN = mean)

> colnames(medias\_por\_nivel) <- c("Fator", "Media\_Numero")

> print(medias\_por\_nivel)

Fator Media\_Numero

1 A 10.25

2 B 5.00

3 C 6.00

1. Mostre a soma da coluna numérica quando a coluna de valores lógicos é Verdadeira. Faça o mesmo para quando é Falsa.

> soma\_verdadeiros <- sum(data\_frame$Numero[data\_frame$Logico == TRUE])

> soma\_falsos <- sum(data\_frame$Numero[data\_frame$Logico == FALSE])

> print(soma\_verdadeiros)

1. 44

> print(soma\_falsos)

1. 30

> soma\_verdadeiros <- sum(data\_frame$Numero[data\_frame$Logico == TRUE])

> soma\_falsos <- sum(data\_frame$Numero[data\_frame$Logico == FALSE])

> print(soma\_verdadeiros)

1. 44

> print(soma\_falsos)

[1] 30

1. Mostre como extrair o vetor mpg do banco de dados mtcars (lembre que o mesmo já existe no R)

> mpg\_vector <- mtcars$mpg

> print(mpg\_vector)

[1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2

[11] 17.8 16.4 17.3 15.2 10.4 10.4 14.7 32.4 30.4 33.9

[21] 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4 15.8 19.7

[31] 15.0 21.4

1. Analise o resultado de aplicar a função str() ao banco mtcars.

> str(mtcars)

'data.frame': 32 obs. of 11 variables:

$ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...

$ cyl : num 6 6 4 6 8 6 8 4 4 6 ...

$ disp: num 160 160 108 258 360 ...

$ hp : num 110 110 93 110 175 105 245 62 95 123 ...

$ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...

$ wt : num 2.62 2.88 2.32 3.21 3.44 ...

$ qsec: num 16.5 17 18.6 19.4 17 ...

$ vs : num 0 0 1 1 0 1 0 1 1 1 ...

$ am : num 1 1 1 0 0 0 0 0 0 0 ...

$ gear: num 4 4 4 3 3 3 3 4 4 4 ...

$ carb: num 4 4 1 1 2 1 4 2 2 4 ...

Essas informações são úteis para entender a estrutura do conjunto de dados e podem ajudar na manipulação e análise dos dados.

1. Filtre a banco airquality (também já existe no R) com apenas linhas em que Ozone

não é NA e Month é igual a 5. Qual o número de linhas e colunas do banco filtrado?

> airquality\_filtrado <- subset(airquality, !is.na(Ozone) & Month == 5)

> linhas <- nrow(airquality\_filtrado)> colunas <- ncol(airquality\_filtrado)

> print(linhas)

1. 26

> print(colunas)

[1] 6

**Operações com Arrays:**

1. Crie um array vazio de dimensão 3x3 e preencha-o com valores de 1 a 9.

> array\_vazio <- array(NA, dim = c(3, 3))

> array\_vazio <- 1:9

> dim(array\_vazio) <- c(3, 3)

> print(array\_vazio)

[,1] [,2] [,3]

[1,] 1 4 7

[2,] 2 5 8

[3,] 3 6 9

1. Realize a multiplicação elementar (por elemento) de 2 arrays.

> array1 <- array(1:9, dim = c(3, 3))

> array2 <- array(9:1, dim = c(3, 3))

> resultado <- array1 \* array2

> print(resultado)

[,1] [,2] [,3]

[1,] 9 24 21

[2,] 16 25 16

[3,] 21 24 9

1. Encontre o índice de linha e coluna do valor máximo em um array de duas dimensões (consulte a ajuda de which()).

> array\_exemplo <- array(1:9, dim = c(3, 3))

> indice\_max <- which.max(array\_exemplo)

> linha\_max <- (indice\_max - 1) %/% nrow(array\_exemplo) + 1

> coluna\_max <- (indice\_max - 1) %% nrow(array\_exemplo) + 1

> print(linha\_max)

1. 3

> print(coluna\_max)

[1] 3

1. Que tipo de objeto é criado com o código seguinte:

meu\_array <- array(1:24, dim = c(3, 4, 2))

> meu\_array <- array(1:24, dim = c(3, 4, 2))> print(meu\_array)

1.

[,1] [,2] [,3] [,4]

[1,] 1 4 7 10

[2,] 2 5 8 11

[3,] 3 6 9 12

2.

[,1] [,2] [,3] [,4]

[1,] 13 16 19 22

[2,] 14 17 20 23

[3,] 15 18 21 24

Ele será um Array tridimensional

1. Qual o resultado se aplicar a função sum() usando apply() no array anterior, definindo o parâmetro MARGIN = c(1,2)? O que esse resultado representa?

> meu\_array <- array(1:24, dim = c(3, 4, 2))

> # Aplicar sum() usando apply() com MARGIN = c(1, 2)> resultado\_sum <- apply(meu\_array, MARGIN = c(1, 2), sum)

> # Exibir o resultado> print(resultado\_sum)

[,1] [,2] [,3] [,4]

[1,] 14 20 26 32

[2,] 16 22 28 34

[3,] 18 24 30 36

1. Separe o conteúdo do array anterior em duas matrizes e calcule e determinante de cada uma.

> meu\_array <- array(1:24, dim = c(3, 4, 2))

> det\_cov1 <- det(cov(meu\_array[, , 1]))

> det\_cov2 <- det(cov(meu\_array[, , 2]))

> print(det\_cov1)

[1] 0

> print(det\_cov2)

[1] 0