

Programação no R - Aula 6

Disciplina: Lógica da programação (de computadores) e
análise de dados no R

Prof. Maurício Garcia de Camargo. IO-FURG.

2025-10-10



Exercício 8 (DESAFIO da aula passada).

Crie uma função que receberá dois números (a e b) e imprima a tabuada de a até b.

Por exemplo: imprimir a tabuada de 1 até 10.



Estruturas de repetição WHILE no R

Estruturas de repetição (loops ou laços) circulam (iteram) pelos itens até que parem.

Diferenças entre FOR e WHILE

- **FOR:** executa um bloco de código um número fixo de vezes, iterando sobre um vetor.

```
1 for (i in 1:5) {  
2   print(i)  
3 }
```



- **WHILE:** executa um bloco de código enquanto uma condição for verdadeira.

```
while (condição_lógica) {  
# bloco de código  
}
```

Exemplo 1. Contar até 5 com while:

```
1  i = 1                # Variável de inicialização  
2  while (i <= 5) {  
3      print(i)  
4      i = i + 1  
5  }
```

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



Exemplo 2. Realizar uma contagem regressiva

```
1 i = 10
2 while (i >= 1) {
3     print(i)
4     i = i - 1
5 }
```

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



Exemplo 3. Realizar a soma dos 100 primeiros números

```
1 soma = 0
2 i = 1
3 while (i <= 100) {
4     soma = soma + i
5     i = i + 1
6 }
7
8 print(soma)
```

```
[1] 5050
```



Encerramos os 5 fundamentos da lógica da programação!!!

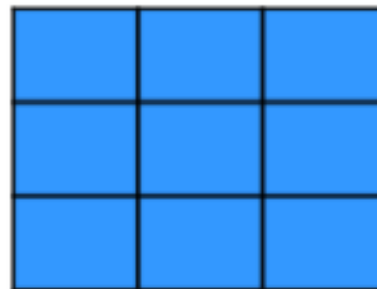
- Function
- IF
- ELSE
- FOR
- WHILE



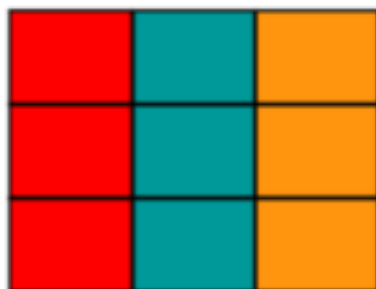
Retornando às estruturas de dados do R



Vector

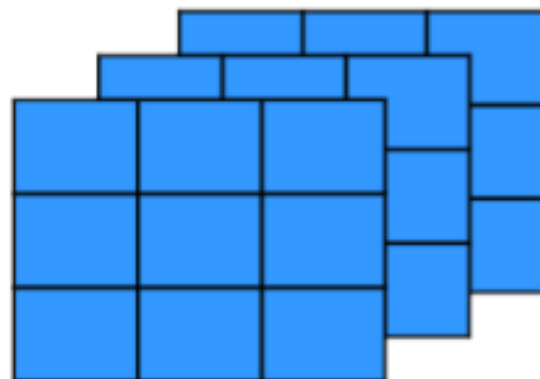


Matrix



Data frame

Colunas podem ser de
tipos diferentes de dados



Array



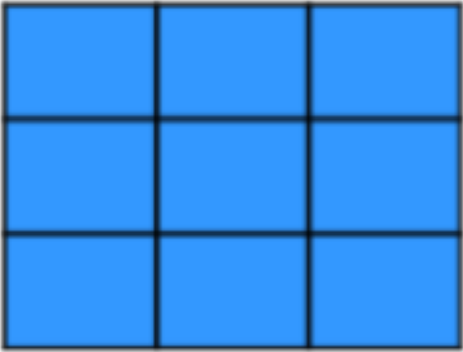
VETOR: série de números (ou caracteres). Vetor tem uma dimensão.



```
1  ##### Exemplos:
2  v = c(1,2,3,4,5)
3  vs = c('A','B','C','D')
4
5  ##### Funções vetoriais
6  length(v)
7  sum(v)
8
9  ##### Índices de vetores
10 v[1]           # Primeiro elemento
11 v[length(v)]  # Último elemento
12 v[3] = 99      #Atribuir um elemento individual
13
14 ##### Filtragem lógica com operadores booleanos
15 v[(v < 3) & (v != 0)]
```



MATRIZ: série de vetores do mesmo tipo (numérico ou de caracteres). Matriz tem 2 dimensões (linhas e colunas).



```
1  ### Exemplos:  
2  v1 = c(1,2,3,4)  
3  v2 = c(6,7,8,9)  
4  v3 = c(10,11,12,13)  
5  m = cbind(v1,v2,v3)  
6  m
```

	v1	v2	v3
[1,]	1	6	10
[2,]	2	7	11
[3,]	3	8	12
[4,]	4	9	13



MATRIZES: Índices de matrizes

```
1 m = matrix(1:12, nrow=4)
2 m
```

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

```
1 ##### Usando índices de matrizes
2 ##### m[linha,coluna]
3 m[2,3]      # Selecionando o elemento da linha 2 e coluna 3
```

```
[1] 10
```

```
1 m[3,]      # Para selecionar a terceira linha inteira (vetor)
```

```
[1]  3  7 11
```

```
1 m[,2]      # Para selecionar a segunda coluna inteira (vetor)
```

```
[1] 5 6 7 8
```



MATRIZES: Índices de matrizes

```
1 m[c(1,3),] # Selecionando o primeira e terceira linhas (submatriz)
```

	[,1]	[,2]	[,3]
[1,]	1	5	9
[2,]	3	7	11

```
1 m[2:4,c(1,3)] # Selecionando as linhas de 2 a 4 e as colunas 1 e 3 (subma
```

	[,1]	[,2]
[1,]	2	10
[2,]	3	11
[3,]	4	12



Funções específicas para matrizes

```
1 ncol(m)      # Número de colunas da matriz m
```

```
[1] 3
```

```
1 nrow(m)      # Número de linhas da matriz m
```

```
[1] 4
```

```
1 dim(m)       # Dimensões da matriz (vetor do número de linhas e colunas)
```

```
[1] 4 3
```

```
1 t(m)         # Transposição da matriz (linhas viram colunas)
```

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



ARRAY: muitas dimensões.

Array generaliza a matriz para n-dimensões.

Em array, todos os dados são do mesmo tipo, assim como as matrizes.

Array é pouco usada em problemas aplicados.

Array é usada em conceitos matemáticos multi-dimensionais.

Não se preocupe muito com elas, apenas entenda o conceito de multi-dimensional, baseado em Linhas X Colunas X Camadas, e que isso pode mais mais dimensões ainda.



Criando arrays

```
1 x = 1:24
2 A = array(x, dim = c(3,4,2)) # Array 3D: (linhas, colunas, "camadas")
3 A
```

```
, , 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

```
1 dim(A)
```

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



DATAFRAME = planilha (colunas de diferentes tipos de dados)



- Dataframe sempre é bidimensional (Linhas X Colunas).
- Cada coluna é um vetor do mesmo comprimento.
- Cada coluna pode ser de um tipo diferente de dado (numérico ou de caracteres).
- R pode substituir o Excel em planilhagem.



Criando um dataframe

Como criar um Dataframe usando a função `data.frame()`

```
1 nome = c('Fulano', 'Beltrano', 'Ciclano') # Criando o vetor de nomes
2 idade = c(12, 18, 20)
3 id = 101:103
4 df = data.frame(nome, idade, id)
5 df
```

	nome	idade	id
1	Fulano	12	101
2	Beltrano	18	102
3	Ciclano	20	103

Seleção de dados feita como matrizes, usando os índices bidimensionais (Linhas X Colunas).

```
1 df[1:2, 2:3]
```

	idade	id
1	12	101
2	18	102



Extraindo os vetores das colunas usando \$ e o nome da coluna

```
1 df$idade
```

```
[1] 12 18 20
```

```
1 df$nome
```

```
[1] "Fulano" "Beltrano" "Ciclano"
```

Para criar uma nova coluna usando \$, basta atribuir o valor a um novo nome de coluna (apelido).

```
1 df$apelido = c('A', 'B', 'C')  
2 df
```

	nome	idade	id	apelido
1	Fulano	12	101	A
2	Beltrano	18	102	B
3	Ciclano	20	103	C



Exemplo 1:

Crie um dataframe com 4 colunas e 4 linhas, assim:

id (1..4), sexo (M/F), altura (m), peso (kg).

Depois, calcule o IMC ($= \text{peso} / \text{altura}^2$) como nova coluna.

```
1 id = 1:4
2 sexo = c('F', 'M', 'M', 'F')
3 altura = c(1.58, 1.87, 1.75, 1.72)
4 peso = c(62, 91, 78, 72)
5 df1 = data.frame(id, sexo, altura, peso)
6 df1$imc = trunc(df1$peso / df1$altura^2)
7 df1
```

	id	sexo	altura	peso	imc
1	1	F	1.58	62	24
2	2	M	1.87	91	26
3	3	M	1.75	78	25
4	4	F	1.72	72	24



Índices em dataframes: Assim como matrizes, dataframes são bidimensionais (Linhas X Colunas).

```
1 df1
```

```
  id sexo altura peso imc
1   1    F   1.58   62  24
2   2    M   1.87   91  26
3   3    M   1.75   78  25
4   4    F   1.72   72  24
```

```
1 df1[1,2] # Extrai o elemento da linha 1 e coluna 2.
```

```
[1] "F"
```

```
1 df1[3,] # Extrai apenas a terceira linha.
```

```
  id sexo altura peso imc
3   3    M   1.75   78  25
```

```
1 df1[,2] # Extrai apenas a segunda coluna.
```

```
[1] "F" "M" "M" "F"
```

```
1 df1$sexo # Também extrai a segunda coluna
```

```
[1] "F" "M" "M" "F"
```



```
1 df1
```

	id	sexo	altura	peso	imc
1	1	F	1.58	62	24
2	2	M	1.87	91	26
3	3	M	1.75	78	25
4	4	F	1.72	72	24

```
1 df1[c(1,3),] # Extrai a linha 1 e a linha 3 inteiras.
```

	id	sexo	altura	peso	imc
1	1	F	1.58	62	24
3	3	M	1.75	78	25

```
1 df1[c(1,3),2] # Extrai a linha 1 e a linha 3 da segunda coluna.
```

```
[1] "F" "M"
```

```
1 df1[1:3,1:3] # Extrai da linha 1 até a linha 3 das colunas de 1 a 3.
```

	id	sexo	altura
1	1	F	1.58
2	2	M	1.87
3	3	M	1.75



Como o vetor **df1** foi criado com os nomes das colunas, podemos selecioná-las pelo nome.

```
1 df1
```

	id	sexo	altura	peso	imc
1	1	F	1.58	62	24
2	2	M	1.87	91	26
3	3	M	1.75	78	25
4	4	F	1.72	72	24

```
1 # Selecionando todas as linhas e 2 colunas.
```

```
2
```

```
3 df1[,c('id','sexo')]
```

	id	sexo
1	1	F
2	2	M
3	3	M
4	4	F



Os poderosos filtros em dataframes no R

Os filtros funcionam tal como em vetores e matrizes, aplicados separadamente às linhas e colunas do dataframe.

```
1 df1
```

	id	sexo	altura	peso	imc
1	1	F	1.58	62	24
2	2	M	1.87	91	26
3	3	M	1.75	78	25
4	4	F	1.72	72	24

```
1 # Selecionando as linhas com imc > 24
```

```
2
```

```
3 df1[df1$imc > 24, ]
```

	id	sexo	altura	peso	imc
2	2	M	1.87	91	26
3	3	M	1.75	78	25



```
1 df1
```

	id	sexo	altura	peso	imc
1	1	F	1.58	62	24
2	2	M	1.87	91	26
3	3	M	1.75	78	25
4	4	F	1.72	72	24

```
1 # Selecionando as linhas das mulheres com peso < 77
```

```
2
```

```
3 df1[(df1$sexo=='F') & (df1$peso < 77) , ]
```

	id	sexo	altura	peso	imc
1	1	F	1.58	62	24
4	4	F	1.72	72	24

```
1 # Selecionando as linhas dos homens com peso > 80 e mostrando apenas o id e
```

```
2
```

```
3 df1[(df1$sexo=='M') & (df1$peso > 80) , c('id','imc')]
```

	id	imc
2	2	26



Exercícios de dataframes

Considere o seguinte dataframe proveniente de uma campanha de amostragem de bentos:

```
1 local = c('L1', 'L1', 'L1', 'L2', 'L2', 'L2')
2 amostra = c('a1', 'a2', 'a3', 'a1', 'a2', 'a3')
3 abund = c(23, 45, 0, 56, 0, 25)
```

Com esses dados, crie um dataframe chamado “dados” e responda:

- 1 - Extraia o vetor correspondente à primeira coluna.
- 2 - Extraia o elemento da segunda linha e terceira coluna.
- 3 - Extraia apenas as linhas correspondentes aos locais L1.



4. Extraia apenas as linhas dos locais L2 em que a abundância seja maior que 30.
5. Extraia apenas as colunas **amostra** e **abund** dos locais L1 em que a abundância seja diferente de zero.
6. DESAFIO: Crie uma nova coluna no dataframe chamada **pres_aus** com valores de 1 e 0 para presença e ausência de abundância.



Manipulação de arquivos no R para abrir planilhas do Excel como dataframe no R

- **Working dir** no RStudio: como manipular.
- Criar uma planilha no Excel, como nome de coluna e salvar como formato csv.
- No R, abrir o arquivo usando a função `read.csv` ou `read.csv2`.
- Alternativamente, é possível abrir um arquivo Excel (xls ou xlsx) diretamente no R, usando o pacote `readxl`.



