Vetores e listas em R

Em programação, **estrutura de dados** refere-se ao formato de organização e armazenamento de dados. É importante entendê-las, pois você irá trabalhar com essas estruturas com frequência ao usar R na análise de dados. As estruturas de dados mais comuns na linguagem de programação R incluem:

- **Vetores**
- **Data frames**
- **Matrizes**
- **Matrizes unidimensionais**

A estrutura de dados é como uma casa que abriga seus dados.

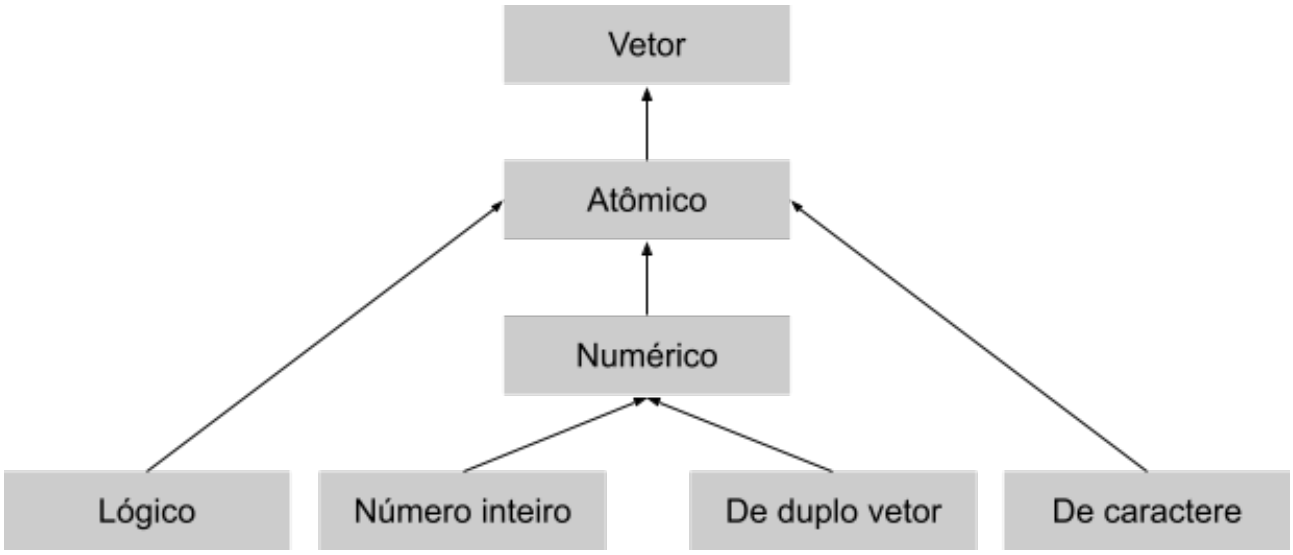
Essa leitura se concentrará nos vetores. Posteriormente, você aprenderá mais sobre os **data frames, matrizes** e **arrays**.
Os vetores dividem-se em dois tipos: **vetores atômicos** e **listas**. A seguir, você aprenderá sobre as propriedades básicas desses dois tipos e como usar o código em R para criá-los.

VETORES ATÔMICOS
Antes de mais nada, vamos explorar os diferentes tipos de vetores atômicos e, em seguida, veremos como usar o código em R para criar, identificar e nomear os vetores. Anteriormente, você aprendeu que um **vetor** é um grupo de elementos de dados do *mesmo* tipo, armazenados em uma sequência em R. É impossível haver um vetor que inclua elementos lógicos e numéricos.

Há seis tipos principais de vetores atômicos: logical (lógico), integer (números inteiros), double (de duplo vetor), character (que inclui strings), complex (números complexos) e raw (números não processados). Os dois últimos tipos (complex e raw) não são tão comuns na análise de dados, portanto, vamos nos concentrar nos quatro primeiros. Juntos, os vetores integer e double são conhecidos como vetores numéricos, pois ambos contêm números. Na tabela abaixo você encontra um resumo dos quatro principais tipos:

Digite	Descrição	Exemplo
Lógico	Verdadeiro/Falso	TRUE
Número inteiro	Valores inteiros positivos e negativos	3
De duplo vetor	Valores decimais	101,175
De caractere	Valores de string/caractere	“Coding”

O diagrama ilustra a hierarquia de relações entre esses quatro principais tipos de vetores:



Como criar vetores

Uma forma de criar um vetor é usar a função **c()** (conhecida como função **"combine"**). A função c() em R combina vários valores em um vetor. Em R, esse função é apenas a letra **"c"** seguida pelos valores que você quer no vetor dentro dos parênteses, separados por uma vírgula: **c(x, y, z, …).**
Você pode usar, por exemplo, a função c() para armazenar dados numéricos em um vetor.
c(2.5, 48.5, 101.5)

Para criar um vetor de números inteiros com a função c(), é necessário inserir a letra "L" logo após cada número.
c(1L, 5L, 15L)

Você também pode criar um vetor com caracteres ou lógicos.
c("Sara" , "Lisa" , "Anna")
c(TRUE, FALSE, TRUE)

Como definir as propriedades dos vetores
Cada vetor criado terá duas principais propriedades: **tipo e comprimento.**

Defina com qual tipo de vetor você trabalha com a função **typeof()**. Insira o código do vetor dentro dos parênteses da função. Ao executá-la, R informará o tipo. Por exemplo:
typeof(c("a" , "b"))
#> [1] "character"

Observe que o resultado da função typeof do exemplo é a palavra "character" (caractere). Da mesma forma, se usar a função typeof em um vetor com valores inteiros, o resultado incluirá, por sua vez, "integer":
typeof(c(1L , 3L))
#> [1] "integer"

Use a função **length()** para determinar o comprimento de um vetor que já existe (ou seja, o número de elementos contidos nele). No exemplo, usamos um operador de atribuição para atribuir o vetor à variável *x* e, então, aplicamos a função length() à variável. Ao executarmos a função, R informa que o comprimento é de 3.

```
x <- c(33.5, 57.75, 120.05)
length(x)
#> [1] 3
```

Você também pode verificar se o vetor é de determinado tipo com a função **is**: **is.logical(), is.double(), is.integer(), is.character()**. No exemplo, R retorna um valor de TRUE pois o vetor contém números inteiros.

```
x <- c(2L, 5L, 11L)
is.integer(x)
#> [1] TRUE
```

No exemplo, R retorna um valor de FALSE pois o vetor *não* inclui caracteres, mas sim elementos lógicos.

```
y <- c(TRUE, TRUE, FALSE)
is.character(y)
#> [1] FALSE
```

COMO NOMEAR VETORES

Todos os tipos de vetores podem ser nomeados. Os nomes são úteis para se escrever um código legível e ao descrever objetos em R. Para nomear os elementos de um vetor, use a função **names()**. Vamos atribuir, por exemplo, a variável x a um novo vetor com três elementos.

```
x <- c(1, 3, 5)
```

Use a função names() para atribuir um nome diferente para cada elemento do vetor.

```
names(x) <- c("a", "b", "c")
```

Ao executar o código, R mostra que o primeiro elemento do vetor tem o nome de a, o segundo b e o terceiro c.

```
x
#> a b c
#> 1 3 5
```

Lembre-se de que um vetor atômico só pode conter elementos do mesmo tipo. Para armazenar elementos de tipos diferentes na mesma estrutura de dados, opte pela lista.

COMO CRIAR LISTAS

As **listas** diferem-se dos vetores atômicos pois seus elementos podem ser de qualquer tipo, como datas, data frames, vetores, matrizes, etc. Elas podem até mesmo conter outras listas.

Use a função **list()** para criar uma lista. Assim como na função c(), a função list() é apenas list seguida dos valores que quer na lista dentro dos parênteses: **list(x, y, z, …)**. No exemplo, criamos uma lista com quatro diferentes tipos de elementos: character ("a"), **integer (1L), double (1.5) e logical (TRUE).**

```
list("a", 1L, 1.5, TRUE)
```

Como já dito antes, as listas podem conter outras listas. Você pode até mesmo armazenar uma lista dentro de uma lista dentro de uma lista (e por aí vai), se quiser.
list(list(list(1 , 3, 5)))

COMO DEFINIR A ESTRUTURA DAS LISTAS
Use a função **str()** para saber os tipos de elementos inclusos em uma lista. Para isso, insira o código da lista dentro dos parênteses da função. Ao executar a função, R mostrará a estrutura de dados da lista com a descrição dos elementos e dos tipos destes. Vamos aplicar a função str() em nosso primeiro exemplo de lista.
str(list("a", 1L, 1.5, TRUE))

Nós executamos a função e, então, R nos informa que a lista contém quatro elementos, os quais consistem em quatro tipos diferentes: **character (chr), integer (int), number (num) e logical (logi)**.
#> List of 4
#> $: chr "a"
#> $: int 1
#> $: num 1.5
#> $: logi TRUE

Vamos usar a função **str()** para descobrir a estrutura do segundo exemplo. Primeiramente, vamos atribuir a lista à variável ***z***. Assim, fica mais fácil inserir na função **str()**.
z <- list(list(list(1 , 3, 5)))

Vamos executar a função.
str(z)
#> List of 1
#> $:List of 1
#> ..$:List of 3
#> .. ..$: num 1
#> .. ..$: num 3
#> .. ..$: num 5

O recuo dos símbolos **$** refletem a estrutura aninhada da lista. Veja que há três níveis (ou seja, há uma lista dentro de uma lista dentro de uma lista).

COMO NOMEAR LISTAS
Assim como os vetores, as listas podem ser nomeadas. Você pode nomear os elementos de uma lista, com a função list(), ao criá-la pela primeira vez:
list('Chicago' = 1, 'New York' = 2, 'Los Angeles' = 3)
$Chicago
[1] 1
$`New York`
[1] 2
$`Los Angeles`
[1] 3

Recurso adicional

Para saber mais sobre vetores e listas, consulte
R for Data Science, capítulo 20: Vetores

. R for Data Science é um recurso clássico que ensina como usar R em ciência e análise de dados. Nele você encontra de tudo, desde como limpar e visualizar a como comunicar seus dados. Para mais detalhes sobre o assunto de vetores e listas, esse capítulo é um ótimo ponto de partida.