Introdução ao R e ao RStudio

Igo da Costa Andrade

2023-10-26

2.11 Exercícios

1. Qual é a soma dos primeiros 100 números inteiros positivos? A fórmula para a soma dos inteiros de 1 até n é $n\,(n+1)\,/2$. Defina $n = 100$ e então use `R` para calcular a soma de 1 até 100 usando a fórmula. Qual é a soma?

```
n <- 100

soma = n * (n+1) / 2
```

   Resposta: A soma dos primeiros 100 inteiros positivos é 5050.

2. Agora use a mesma fórmula para calcular a soma dos inteiros de 1 a 1000.

```
n <- 1000
soma = n * (n+1) / 2
```

   Resposta: A soma dos primeiros 1000 inteiros positivos é 500500.

3. Observe o resultado da digitação do seguinte códico em `R`:

```
n <- 1000
x <- seq(1, n)
sum(x)
```

```
## [1] 500500
```

   Com base no resultado, o que você acha que as funções `seq` e `sum` fazem?

   a. `sum` cria uma lista de números e `seq` os soma.

   b. `seq` cria uma lista de números e `sum` os soma.

   c. `seq` cria uma lista aleatória e `sum` calcula a soma de 1 a 1.000.

   d. `sum` sempre retorna o mesmo número.

4. Em matemática e programação, dizemos que avaliamos uma função quando substituimos o argumento por um determinado número. Então, se digitarmos `sqrt(4)`, avaliaremos a função `sqrt`. Em `R`, você pode avaliar uma função dentro de outra função. As avaliações acontecem de dendro para fora. Use uma linha de código para calcular o logarítmo, na base 10, da raiz quadrada de 100.

```
log(sqrt(100), base=10)
```

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

5. Qual das opções a seguir sempre retornará o valor numérico armazenado em `x`?

   a. `log(10^x)`

   b. `log10(x^10)`

c. `log(exp(x))`

d. `log(x, base=2)`

6. Certifique-se de que o conjunto de dados de assassinatos nos EUA esteja carregado. Use a função `str` para examinar a estrutura do ibjeto `murder`. Qual das alternativas a seguir descreve melhor as variáveis representadas neste *data frame*.

   a. Os 51 estados.

   b. As taxas de homicídio em todos os 50 estados e DC.

   c. O nome do estado, a abreviatura do nome do estado, a região do estado e a população do estado e o número total de assassinatos em 2010.

   d. `str` não apresenta informações relevantes.

```
library(dslabs)
data("murders")

str(murders)
```

```
## 'data.frame':    51 obs. of  5 variables:
##  $ state     : chr  "Alabama" "Alaska" "Arizona" "Arkansas" ...
##  $ abb       : chr  "AL" "AK" "AZ" "AR" ...
##  $ region    : Factor w/ 4 levels "Northeast","South",..: 2 4 4 2 4 4 1 2 2 2 ...
##  $ population: num  4779736 710231 6392017 2915918 37253956 ...
##  $ total     : num  135 19 232 93 1257 ...
```

7. Quais são os nomes das colunas usadas pelo *data frame* para essas cinco variáveis?

```
colnames(murders)
```

```
## [1] "state"      "abb"        "region"     "population" "total"
```

8. Use o acessador `$` para extrair as abreviações de estado e atribuí-las ao objeto `a`. Qual é a classe deste objeto?

```
a <- murders$abb

class(a)
```

```
## [1] "character"
```

9. Agora use os colchetes para extrair as abreviações de estado e atribuí-las ao objeto `b`. Use a função `identical` para determinar se `a` e `b` são iguais.

```
b <- murders[['abb']]

identical(a, b)
```

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

10. Vimos que a coluna `region` armazena um fator. Você pode corroborar isso digitando:

```
class(murders$region)
```

```
## [1] "factor"
```

    Com uma linha de código, use as funções `levels` e `length` para determinar o número de regiões definidas por este conjunto de dados.

```
length(levels(murders$region))
```

```
## [1] 4
```

11. A função `table` pega um vetor e retorna a frequência de cada elemento. Você pode ver rapidamente quantos estados existem em cada região aplicando esta função. Use esta função em uma linha de código para criar uma tabela de estados por região.

```
table(murders$region)
```

```
##
##     Northeast         South North Central          West
##             9            17            12            13
```

12. Use a função `c` para criar um vetor com as altas temperaturas médias em janeiro para Pequim, Lagos, Paris, Rio de Janeiro, San Juan e Toronto, que são 35, 88, 42, 84, 81 e 30 graus Fahrenheit. Chame o objeto `temp`.

```
temp <- c(35, 88, 42, 84, 81, 30)
```

13. Agora crie um vetor com os nomes das cidades e chame o objeto `city`.

```
city <- c("Pequim", "Lagos", "Paris", "Rio de Janeiro", "San Juan", "Toronto")
```

14. Utilize a função `names` e os objetos definidos nos exercícios anteriores para associar os dados de temperatura à sua cidade correspondente.

```
names(temp) <- city

temp
```

```
##         Pequim          Lagos          Paris Rio de Janeiro       San Juan
##             35             88             42             84             81
##        Toronto
##             30
```

15. Utilize os operadores `[` e `:` para acessar a temperatura das três primeiras cidades da lista.

```
temp[1:3]
```

```
## Pequim  Lagos  Paris
##     35     88     42
```

16. Use o operador `[` para acessar a temperatura de Paris e San Juan.

```
temp[c("Paris", "San Juan")]
```

```
##    Paris San Juan
##       42       81
```

17. Use o operador `:` para criar a sequência de números $12, 13, 14, \cdots, 73$.

```
vec <- seq(from=12, to=73)

vec
```

```
##  [1] 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
## [26] 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
## [51] 62 63 64 65 66 67 68 69 70 71 72 73
```

18. Crie um vetor contendo todos os números ímpares positivos menores que 100.

```
impares_menores_que_100 <- seq(from=1, to=100, by=2)

impares_menores_que_100
```

```
##  [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49
## [26] 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

19. Crie um vetor de números que comece em 6, não passe de 55 e adicione números em incrementos de 4/7: 6, 6 + 4/7, 6 + 8/7 e assim por diante. Quantos números tem a lista? Dica: use `seq` e `length`.

```
vec <- seq(from=6, to=55, by=4/7)

length(vec)
```

```
## [1] 86
```

20. Qual é a classe do seguinte objeto `a <- seq(1, 10, 0.5)`?

```
a <- seq(1, 10, 0.5)

class(a)
```

```
## [1] "numeric"
```

21. Qual é a classe do seguinte objeto `a <- seq(1, 10)`?

```
a <- seq(1, 10)

class(a)
```

```
## [1] "integer"
```

22. A classe de `class(a<-1)` é numérica, não inteira. O padrão de R é numérico e para forçar um número inteiro, você precisa adicionar a letra `L`. Confirme se a classe de `1L` é inteira.

```
class(1)
```

```
## [1] "numeric"
```

```
class(1L)
```

```
## [1] "integer"
```

23. Defina o seguinte vetor:

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

e use coerção para obter números inteiros.

```
x <- as.numeric(x)

x
```

```
## [1] 1 3 5
```