RWorksheet_Vicinte#1

Anthony Dave Vicinte

2024-09-20

#1. Set up a vector named age, consisting of 34, 28, 22, 36, 27, 18, 52, 39, 42, 29, 35, 31, 27, 22, 37, 34, 19, 20, 57, 49, 50, 37, 46, 25, 17, 37, 42, 53, 41, 51, 35, 24, 33, 41.

```
age <- c(34, 28, 22, 36, 27, 18, 52, 39, 42, 29, 35, 31, 27, 22, 37, 34, 19, 20, 57, 49, 50, 37, 46, 25
```

a. How many data points?

```
length(age)
```

```
## [1] 34
```

2. Find the reciprocal of the values for age.

```
library(MASS)
reciprocalOfage <- 1 / age
fractions(reciprocalOfage)
```

```
##  [1] 1/34 1/28 1/22 1/36 1/27 1/18 1/52 1/39 1/42 1/29 1/35 1/31 1/27 1/22 1/37
## [16] 1/34 1/19 1/20 1/57 1/49 1/50 1/37 1/46 1/25 1/17 1/37 1/42 1/53 1/41 1/51
## [31] 1/35 1/24 1/33 1/41
```

3. Assign also new_age <- c(age, 0, age).

```
new_age <- c(age, 0, age)
new_age
```

```
##  [1] 34 28 22 36 27 18 52 39 42 29 35 31 27 22 37 34 19 20 57 49 50 37 46 25 17
## [26] 37 42 53 41 51 35 24 33 41  0 34 28 22 36 27 18 52 39 42 29 35 31 27 22 37
## [51] 34 19 20 57 49 50 37 46 25 17 37 42 53 41 51 35 24 33 41
```

What happen to the new_age?

-The new_age vector is a combination of the original age values, 0 and then repeats the age values again.

4. Sort the values for age.

```r
sortedOfage <- sort(age)
sortedOfage
```

```
##  [1] 17 18 19 20 22 22 24 25 27 27 28 29 31 33 34 34 35 35 36 37 37 37 39 41 41
## [26] 42 42 46 49 50 51 52 53 57
```

5. Find the minimum and maximum value for age.

```r
minOage <- min(age)
minOage
```

```
## [1] 17
```

```r
maxOfage <- max(age)
maxOfage
```

```
## [1] 57
```

6. Set up a vector named data, consisting of 2.4, 2.8, 2.1, 2.5, 2.4, 2.2, 2.5, 2.3, 2.5, 2.3, 2.4, and 2.7.

```r
data <- c(2.4, 2.8, 2.1, 2.5, 2.4, 2.2, 2.5, 2.3, 2.5, 2.3, 2.4, 2.7)
```

a. How many data points?

```r
length(data)
```

```
## [1] 12
```

7. Generates a new vector for data where you double every value of the data. | What happen to the data?

```r
doubledThedata <- data * 2
doubledThedata
```

```
##  [1] 4.8 5.6 4.2 5.0 4.8 4.4 5.0 4.6 5.0 4.6 4.8 5.4
```

What happen to the data?

- Every value in the vector data multiplies to two.

8. Generate a sequence for the following scenario:

8.1 Integers from 1 to 100.

```
seq1To100 <- seq(1, 100)
seq1To100
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100
```

8.2 Numbers from 20 to 60

```
seq20To60 <- seq(20, 60)
seq20To60
```

```
##  [1] 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44
## [26] 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
```

*8.3 Mean of numbers from 20 to 60

```
mean20To60 <- mean(seq20To60)
mean20To60
```

```
## [1] 40
```

*8.4 Sum of numbers from 51 to 91

```
sum51To91 <- sum(51:91)
sum51To91
```

```
## [1] 2911
```

*8.5 Integers from 1 to 1,000

```
seq_1_1000 <- seq(1, 1000)
```

a. How many data points from 8.1 to 8.4?

```
TotalDataPoints <- length(seq1To100) + length(seq20To60) + length(mean20To60) + length(sum51To91)
TotalDataPoints
```

```
## [1] 143
```

-There are 143 data points from 8.1 to 8.4.

c. For 8.5 find only maximum data points until 10.

```
maxTo10 <- max(seq_1_1000[1:10])
maxTo10
```

```
## [1] 10
```

9. *Print a vector with the integers between 1 and 100 that are not divisible by 3, 5 and 7 using filter option.

```
filteredValues <- Filter(function(i) { all(i %% c(3, 5, 7) != 0) }, seq(100))
filteredValues
```

```
##  [1]  1  2  4  8 11 13 16 17 19 22 23 26 29 31 32 34 37 38 41 43 44 46 47 52 53
## [26] 58 59 61 62 64 67 68 71 73 74 76 79 82 83 86 88 89 92 94 97
```

10. Generate a sequence backwards of the integers from 1 to 100.

```
backwardSeq <- seq(100,1)
backwardSeq
```

```
##   [1] 100  99  98  97  96  95  94  93  92  91  90  89  88  87  86  85  84  83
##  [19]  82  81  80  79  78  77  76  75  74  73  72  71  70  69  68  67  66  65
##  [37]  64  63  62  61  60  59  58  57  56  55  54  53  52  51  50  49  48  47
##  [55]  46  45  44  43  42  41  40  39  38  37  36  35  34  33  32  31  30  29
##  [73]  28  27  26  25  24  23  22  21  20  19  18  17  16  15  14  13  12  11
##  [91]  10   9   8   7   6   5   4   3   2   1
```

11. List all the natural numbers below 25 that are multiples of 3 or 5.

```
multiplesOf3_5 <- seq(1, 24)[seq(1, 24) %% 3 == 0 | seq(1, 24) %% 5 == 0]
multiplesOf3_5
```

```
##  [1]  3  5  6  9 10 12 15 18 20 21 24
```

a. How many data points from 10 to 11?

```
DataPoints11 <- length(backwardSeq) + length(multiplesOf3_5)
DataPoints11
```

```
## [1] 111
```

-There are 111 data points from 10 to 11.

12. Statements can be grouped together using braces '{' and '}'. A group of statements is sometimes called a block. Single statements are evaluated when a new line is typed at the end of the syntactically complete statement. Blocks are not evaluated until a new line is entered after the closing brace.

Enter this statement:

x <- {0 + x + 5 + }

Describe the output. -It outputs error because the x inside the brakcet is not intialized with a value.

13. *Set up a vector named score, consisting of 72, 86, 92, 63, 88, 89, 91, 92, 75, 75 and 77. To access individual elements of an atomic vector, one generally uses the x[i] construction.

Find x[2] and x[3]. Write the R code and its output.

```
score <- c(72, 86, 92, 63, 88, 89, 91, 92, 75, 75, 77)
```

```
score[2]
```

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

```
score[3]
```

```
## [1] 92
```

14. *Create a vector a = c(1,2,NA,4,NA,6,7).

```
a <- c(1,2,NA,4,NA,6,7)
print(a,na.print="-999")
```

```
## [1]    1    2 -999    4 -999    6    7
```

b. Write the R code and its output. Describe the output.

-All the NA in the vector a changed to 999.

15. A special type of function calls can appear on the left hand side of the assignment operator as in > class(x) <- "foo".

#Follow the codes below:

```r
name = readline(prompt="Input your name: ")
```

```
## Input your name:
```

```r
age = readline(prompt="Input your age: ")
```

```
## Input your age:
```

```r
print(paste("My name is",name, "and I am",age ,"years old."))
```

```
## [1] "My name is  and I am  years old."
```

```r
print(R.version.string)
```

```
## [1] "R version 4.4.1 (2024-06-14)"
```