RWorksheet_Alpanghe#1.Rmd

Jersey Gabriel E. Alpanghe

2024-09-04

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.

a. How many data points?

There are 34 data points in total.

b. Write the R code and its output.

```
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, 17, 37, 42, 53, 41, 51,
         35, 24, 33, 41)

print(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
```

```
num_data_points <- length(age)

print(paste("Number of data points: ", num_data_points))
```

```
## [1] "Number of data points:  34"
```

2.Find the reciprocal of the values for age. Write the R code and its output.

```
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, 17, 37, 42, 53, 41, 51,
         35, 24, 33, 41)

reciprocal_age <- 1 / age

print(reciprocal_age)
```

```
##  [1] 0.02941176 0.03571429 0.04545455 0.02777778 0.03703704 0.05555556
##  [7] 0.01923077 0.02564103 0.02380952 0.03448276 0.02857143 0.03225806
## [13] 0.03703704 0.04545455 0.02702703 0.02941176 0.05263158 0.05000000
## [19] 0.01754386 0.02040816 0.02000000 0.02702703 0.02173913 0.04000000
## [25] 0.05882353 0.02702703 0.02380952 0.01886792 0.02439024 0.01960784
## [31] 0.02857143 0.04166667 0.03030303 0.02439024
```

```
if (!require(MASS)) install.packages("MASS")
```

```
## Loading required package: MASS
```

```r
library(MASS)

fraction_reciprocal_age <- fractions(reciprocal_age)

print(fraction_reciprocal_age)
```

```
##  [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).

```r
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, 17, 37, 42, 53, 41, 51,
         35, 24, 33, 41)


new_age <- c(age, 0, age)


print(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
```

4.Sort the values for age. Write the R code and its output.

```r
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, 17, 37, 42, 53, 41, 51,
         35, 24, 33, 41)
sorted_age <- sort(age)


print(sorted_age)
```

```
##  [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. Write the R code and its output.

```r
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, 17, 37, 42, 53, 41, 51,
         35, 24, 33, 41)

min_age <- min(age)
max_age <- max(age)

print(paste("Minimum age:", min_age))
```

```
## [1] "Minimum age: 17"
```

```r
print(paste("Maximum age:", max_age))
```

```
## [1] "Maximum age: 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.

a. How many data points?

There are 12 data points in total.

b.

```
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)

num_data_points <- length(data)

print(num_data_points)
```

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

7. Generates a new vector for data where you double every value of the data.

What happened to the data?

The values within the sequence of data have been doubled.

```
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)


doubled_data <- data * 2

print(doubled_data)
```

```
##  [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
```

8. Generate a sequence for the following scenario:

8.1 Integers from 1 to 100.

```
sequence_8.1 <- 1:100
print(sequence_8.1)
```

```
##   [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

```
sequence_8.2 <- 20:60
print(sequence_8.2)
```

```
##  [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

```
mean_8.3 <- mean(20:60)
print(mean_8.3)
```

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

8.4 Sum of numbers from 51 to 91

```r
sum_8.4 <- sum(51:91)
print(sum_8.4)
```

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

8.5 Integers from 1 to 1,000

```r
sequence_8.5 <- 1:1000
max_8.5 <- max(sequence_8.5[sequence_8.5 <= 10])
print(max_8.5)
```

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

a. How many data points from 8.1 to 8.4?

There are 183 data points from 8.1 to 8.4 in total.

b. Write the R code and its output from 8.1 to 8.4. 8.1 Integers from 1 to 100.

```r
sequence_8.1 <- 1:100
print(sequence_8.1)
```

```
##   [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

```r
sequence_8.2 <- 20:60
print(sequence_8.2)
```

```
##  [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

```r
mean_8.3 <- mean(20:60)
print(mean_8.3)
```

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

8.4 Sum of numbers from 51 to 91

```r
sum_8.4 <- sum(51:91)
print(sum_8.4)
```

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

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

```r
sequence_8.5 <- 1:1000

max_value_8.5 <- max(sequence_8.5[sequence_8.5 <= 10])
print(max_value_8.5)
```

```
## [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.

```r
sequence <- seq(100)

filtered_numbers <- Filter(function(i) { all(i %% c(3, 5, 7) != 0) }, sequence)

print(filtered_numbers)
```

```
##  [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. Write the R code and its output.

```r
backwards_sequence <- seq(100, 1, by = -1)

print(backwards_sequence)
```

```
##   [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. Find the sum of these multiples.

   a. How many data points from 10 to 11?

   b. Write the R code and its output from 10 and 11.

Number10:

```r
backwards_sequence <- seq(100, 1, by = -1)

print(backwards_sequence)
```

```
##   [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
```

Number11:

```r
upper_limit <- 25

numbers <- 1:(upper_limit - 1)

multiples <- numbers[numbers %% 3 == 0 | numbers %% 5 == 0]


sum_multiples <- sum(multiples)

print(multiples)
```

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

```r
print(sum_multiples)
```

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

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.

This code will caused a syntax error because the + operator expected another operand but none is provided. Since the block is incomplete, the interpreter will not evaluate the code until it is syntactically correct.

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.

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

second_element <- score[2]
third_element <- score[3]

print(second_element)
```

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

```r
print(third_element)
```

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

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

a. Change the NA to 999 using the codes print(a,na.print="-999").

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

```r
a <- c(1, 2, NA, 4, NA, 6, 7)

print(a, na.print = "-999")
```

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

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:

name = readline(prompt="Input your name:") age = readline(prompt="Input your age:") print(paste("My name is",name, "and I am",age ,"years old.")) print(R.version.string) What is the output of the above code?

The output shows the name, and age the user will input and pastes them into the sentence of "My name is,________, and I am ___ years old." along with the R version 4.3.1 (2024-05-20) provided.