RWorksheet.Sorenio#1.Rmd

2024-09-18

1. Set up a vector named age, cosnsisting 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, 17, 37, 42, 53, 41, 51, 35, 24, 33, 41)
```

a. How many data points?

b. Write the R code and its output.

```
num_data_points <- length(age)
print(num_data_points)
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)
```

2. Find the reciprocal of the values for age.

Write the R code and its output.

```
reciprocalage <- 1 / age
print(reciprocalage)
```

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

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

4. Sort the values for age.

```
age_sorted <- sort(age)
print(age_sorted)
```

5. Find the minimum and maximum value for age.

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

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

```
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

b. Write the R code and its output.

```
numdatapoints <- length(data)
print(numdatapoints)
```

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

```
doubled_data <- data * 2
print(doubled_data)
```

8. Generate a sequence for the following scenario:

8.1 Integers from 1 to 100.

```
int_1_to_100 <- 1:100
print(int_1_to_100)
```

8.2 Numbers from 20 to 60

```
num_20_to_60 <- 20:60
print(num_20_to_60)
```

8.3 Mean of numbers from 20 to 60

```
mean_20_to_60 <- mean(numbers_20_to_60)
print(mean_20_to_60)
```

8.4 Sum of numbers from 51 to 91

```
sum_51_to_91 <- sum(51:91)
print(sum_51_to_91)
```

8.5 Integers from 1 to 1,000

```
ints_1_to_1000 <- 1:1000
print(ints_1_to_1000)
```

a. How many data points from 8.1 to 8.4?_____

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

```
variable8_1 <- length(int_1_to_100)
variable8_2 <- length(num_20_to_60)
variable8_3 <- 1
variable8_4 <- 1
totaldatapoints <- variable8_1 + variable8_2 + variable8_3 + variable8_4
print(totaldatapoints)
```

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

max value = 10

```
ints_up_to_10 <-
ints_1_to_1000[ints_1_to_1000 <= 10]
maxvalue <- max(ints_up_to_10)
print(maxvalue)
```

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

Write the R code and its output.

```
not_divisible <- Filter(function(i) { all(i %% c(3, 5, 7) != 0) }, seq(1, 100))
print(not_divisible)
```

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

Write the R code and its output.

```
backwards_sequence <- rev(1:100)
print(backwards_sequence)
```

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

```
backwards_sequence <- rev(1:100)
print(backwards_sequence)
multsof3to5 <- seq(1, 24)[seq(1, 24) %% 3 == 0 | seq(1, 24) %% 5 == 0]
print(multsof3to5)
SumOfMultiples <- sum(multsof3to5)
SumOfMultiples
numbers <- 1:24
multiples <- numbers[numbers %% 3 == 0 | numbers %% 5 == 0]
sum_of_multiples <- sum(multiples)
print(multiples)
print(sum_of_multiples)
```

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 + }
```

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 <- score[2]
score_3 <- score[3]
print(score_2)
print(score_3)
```

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").

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

```
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)
```