Worksheet-1 in R

Name: Junmar Mahipus BSIT-2A

Worksheet for R Programming

Instructions:

- Use RStudio or the RStudio Cloud accomplish this worksheet. + Save the R script as *RWorksheet_lastname#1.R*.
- Create your own *GitHub repository* and push the R script as well as this pdf worksheet to your own repo.

Accomplish this worksheet by answering the questions being asked and writing the code manually.

Using functions:

seq(), assign(), min(), max(), c(), sort(), sum(), filter()

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?
   Answer: 12 data points

b. Write the R code and its output
age
#Output : [1] 34 28 22 36 27 18 52 39 42 29 35 31 27 22 37 34 19 20 57 49 50[22] 37 46 25 17 37 42 53 41 51 35 24 33 41

2. Find the reciprocal of the values for age. Write the R code and its output. reciprocal <- function(age) vec <- 1/age rage <- reciprocal(age) rage

#[1] 0.02941176 0.03571429 0.04545455 0.02777778
#[5] 0.03703704 0.05555556 0.01923077 0.02564103
#[9] 0.02380952 0.03448276 0.02857143 0.03225806
#[13] 0.03703704 0.04545455 0.02702703 0.02941176
#[17] 0.05263158 0.05000000 0.01754386 0.02040816
#[21] 0.02000000 0.02702703 0.02173913 0.04000000
#[25] 0.05882353 0.02702703 0.02380952 0.01886792
#[29] 0.02439024 0.01960784 0.02857143 0.04166667
#[33] 0.03030303 0.02439024

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

What happen to the new_age?
new_age <- c(age, 0, age)
new_age
#Answer: it display the numbers given

4. Sort the values for age.
sort(age)
Write the R code and its output.
#17 18 19 20 22 22 24 25 27 27 28 29 31 33 34 34 35 35 36 37 37
#[22] 37 39 41 41 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.
max(age) # the output is 57
min(age) # the output is 17

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?
# Answer: 12
b. Write the R code and its output.
Data
#Output : 2.4 2.8 2.1 2.5 2.4 2.2 2.5 2.3 2.5 2.3 2.4 2.7

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

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)

2*Data

8. Generate a sequence for the following scenario:

8.1 Integers from 1 to 100.
#Answer: seq(1:100) #Output is number sequence
from 1-100.
8.2 Numbers from 20 to 60
x <- 20:60 print(seq(x))#Output is
numbers 1 -41. 8.3 Mean of numbers
from 20 to 60

print(mean(x)) #output is 40 8.4 Sum
of numbers from 51 to 91

print(sum(51:91)) # output is 2911

#8.3 Mean of numbers from 20 to 60
print(mean(x))

#*8.4 Sum of numbers from 51 to 91
print(sum(51:91))

8.5 Integers from 1 to 1,000
seq(1:1000) #Output is number sequence from 1-1000.

a. How many data points from 8.1 to 8.4?_______
#Answer: 143 data points are in 8.1 to 8.4
b. Write the R code and its output from 8.1 to 8.4. seq(1:100) #Output is number sequence from 1-100.

x <- 20:60 print(seq(x)) #Output is
numbers 1 -41.

print(mean(x))#output is 40
print(sum(51:91))# output is 2911
c. For 8.5 find only maximum data points until 10. m <- seq(1:10) max(m)
#Answer is 10

9. *Print a vector with the integers between 1 and 100 that are not divisible by 3, 5 and 7 using filter option. filter(function(i) { all(i %% c(3,5,7) != 0) }, seq(100)) Write the R code and its output.

Filter(function(i) { all(i %% c(3,5,7) != 0) }, seq(100)) #output it displays random numbers
10. Generate a sequence backwards of the integers from 1 to 100.
Write the R code and its output.
seq(from = 100, to = 1)

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

sum((1:25)[((1:25)%%3 == 0) | ((1:25)%%5 == 0)])

a. How many data points from 10 to 11?
1 data points from 10 to 11
b. Write the R code and its output from 10 and 11.
seq(from = 100, to = 1) #output is numbers from 100 to 1
sum((1:25)[((1:25)%%3 == 0) | ((1:25)%%5 == 0)])#output is 168

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.

The output is error because it declares the closing brace after executing the program's as a result the output is error.

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.

Answer: X[2] = 86 x[3] = 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.

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

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

Output:

```
  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?

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

```
> age = readline(prompt="Input your age: ")
Input your age: 12
> print(paste("My name is",name, "and I am",age ,"years old."))
[1] "My name is Gabby and I am 12 years old."
```