JAN IVAN TAMONAN BSIT 2-B

R Programming Worksheet

1. Set up a vector named age, consisting of the following values:

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?

34

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)

length(age)
Output: 35
```

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

```
reciprocal_age <- 1 / age
reciprocal_age
Output:
[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
```

3. Assign new_age <- c(age, 0, age). What happened to the new_age?

New age’s output became:

```
[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
[25] 17 37 42 53 41 51 35 24 33 41  0 34 28 22 36 27 18 52 39 42 29 35 31 27
[49] 22 37 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.

```
sorted_age <- sort(age)
sorted_age
Output:
[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
[25] 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.

```
min_age <- min(age)
max_age <- max(age)
min_age
Output: 17
max_age
Output: 57
```

6. Set up a vector named data, consisting of the following values:

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?

12

b. Write the R code and its output.

```
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)
length(data)  # Output: 12
```

7. Generate a new vector for data where you double every value of the data. What happened to the data?

Each value in `data` is multiplied by 2.

8. Generate a sequence for the following scenarios:
8.1 Integers from 1 to 100.
8.2 Numbers from 20 to 60.
8.3 Mean of numbers from 20 to 60.
8.4 Sum of numbers from 51 to 91.
8.5 Integers from 1 to 1,000 (first 10).
a. How many data points from 8.1 to 8.4?
223

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

```
seq_1_to_100 <- seq(1, 100)
seq_1_to_100
seq_20_to_60 <- seq(20, 60)
seq_20_to_60
mean_20_to_60 <- mean(seq_20_to_60)
mean_20_to_60
sum_51_to_91 <- sum(51:91)
sum_51_to_91
Output:
> seq_1_to_100 <- seq(1, 100)
> seq_1_to_100
  [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
> seq_20_to_60 <- seq(20, 60)
> seq_20_to_60
 [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
[25] 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
>
>
> mean_20_to_60 <- mean(seq_20_to_60)
> mean_20_to_60
[1] 40
>
> sum_51_to_91 <- sum(51:91)
> sum_51_to_91
[1] 2911
```

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

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

```
not_div_3_5_7 <- Filter(function(i) all(i %% c(3, 5, 7) != 0), seq(100))
not_div_3_5_7
Output:
[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
[25] 53 58 59 61 62 64 67 68 71 73 74 76 79 82 83 86 88 89 92 94 97
```

10. Generate a sequence backward of the integers from 1 to 100. Write the R code and its output.

```
seq_backwards <- seq(100, 1)
seq_backwards
Output:
[1] 100  99  98  97  96  95  94  93  92  91  90  89  88  87  86  85  8483
[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, and find their sum.
a. How many data points from 10 to 11?
10 has 100 data points
11 has 11 data points

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

```
multiples_3_5 <- Filter(function(x) x %% 3 == 0 | x %% 5 == 0, 1:24)
multiples_3_5
Output:
```

```
[1]  3  5  6  9 10 12 15 18 20 21 24
sum_multiples_3_5 <- sum(multiples_3_5)
sum_multiples_3_5
output:
[1] 143
```

12. Statements grouped using braces.

```
x <- 0
x <- {x + 5}
x  # Output: 5
```

Explanation: The block adds 5 to `x` (which was initially 0) and assigns the result back to `x`.

13. Set up a vector named score, consisting of:

72, 86, 92, 63, 88, 89, 91, 92, 75, 75, 77.

a. 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]  # Output: 86
score[3]  # Output: 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]    1    2 -999    4 -999    6    7
```

The NA values in the vector are replaced by -999 when printed.

15. Use the following codes to input your name and age:

```
name = readline(prompt="Input your name: ")
age = readline(prompt="Input your age: ")
print(paste("My name is", name, "and I am", age, "years old."))
# Example output: "My name is John and I am 25 years old."
print(R.version.string)
```

Output: R version details (e.g., "R version 4.3.1 (2023-06-16)")