Name: Rema Ann Dolor
BSIT2-A

Worksheet-2 in R

Worksheet for R Programming

Instructions:

- Use RStudio or the RStudio Cloud accomplish this worksheet. + Save the R script as *RWorksheet_lastname#2.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 Vectors

1. Create a vector using : operator

   a. Sequence from -5 to 5. Write the R code and its output. Describe its output.

R code:

```
y <- (-5:5)
y
```

Output:

```
[1] -5 -4 -3 -2 -1  0  1  2  3  4  5
```

   b. `x <- 1:7`. What will be the value of `x`?

answer:

```
x
[1] 1 2 3 4 5 6 7
```

2.* Create a vector using `seq()` function

   a. seq(1, 3, by=0.2) # specify step size
   Write the R code and its output. Describe the output.

R code:

```
l <- seq (1, 3, by = 0.2)
l
```

Output:

```
[1] 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0
```

3. A factory has a census of its workers. There are 50 workers in total. The following list shows their ages: 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, 43, 53, 41, 51, 35, 24, 33, 41, 53, 40, 18, 44, 38, 41, 48, 27, 39, 19, 30, 61, 54, 58, 26, 18.

a. Access 3rd element, what is the value?

```
workers_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, 43, 53, 41, 51, 35,
                 24,33, 41, 53, 40, 18, 44, 38, 41, 48, 27, 39, 19, 30, 61, 54, 58, 26,
                 18)
workers_age[3]
[1] 22
```

b. Access 2nd and 4th element, what are the values?

```
workers_age[2]
 [1] 28
workers_age[4]
 [1] 36
```

c. Access all but the 1st element is not included. Write the R code and its output.

```
workers_age[2:49]
 [1] 28 22 36 27 18 52 39 42 29 35 31 27 22 37 34 19 20
[18] 57 49 50 37 46 25 17 37 43 53 41 51 35 24 33 41 53
[35] 40 18 44 38 41 48 27 39 19 30 61 54 58 26
```

4. *Create a vector `x <- c("first"=3, "second"=0, "third"=9)`. Then named the vector, names(x).

a. Print the results. Then access `x[c("first", "third")]`.
Describe the output.

```
x <- c("first"=3, "second"=3, "third"=9)
 names(x)
 [1] "first"  "second" "third"
```

b. Write the code and its output.

5. Create a sequence x from -3:2.

```
x <- -3:2
x
[1] -3 -2 -1  0  1  2
```

a. Modify 2nd element and change it to 0;

```
x[2] <- 0
x
```

Describe the output.

Output:

```
[1] -3  0 -1  0  1  2
```

b. Write the code and its output.

6. *The following data shows the diesel fuel purchased by Mr. Cruz.

Month	Jan	Feb	March	Apr	May	June
Price per liter (PhP)	52.50	57.25	60.00	65.00	74.25	54.00
Purchase–quantity(Liters)	25	30	40	50	10	45

a. Create a data frame for month, price per liter (php) and purchase-quantity (liter). Write the codes.

R code:

```
month <- c("Jan", "Feb", "Mar", "Apr", "May", "June")
price_per_liter <- c(52.50, 57.25, 60.00, 65.00, 74.25, 54.00)
purchase_quantity <- c(25, 30, 40, 50, 10, 45)

frame <- data.frame(month, price_per_liter, purchase_quantity)
frame
```

b. What is the average fuel expenditure of Mr. Cruz from Jan to June? Note: Use `weighted.mean(liter, purchase)`

```
weighted.mean(price_per_liter, purchase_quantity)
[1] 59.2625
```

7. R has actually lots of built-in datasets. For example, the rivers data "gives the lengths (in miles) of 141 "major" rivers in North America, as compiled by the US Geological Survey".

a. Type "rivers" in your R console. Create a vector data with 7 elements, containing the number of elements (length) in rivers, their sum (sum), mean (mean), median (median), variance (var) standard deviation (sd), minimum (min) and maximum (max).

```
data <- c(length(rivers), sum(rivers), mean(rivers), median(rivers), var(rivers), sd(rivers), min(rivers), max(rivers))
```

```
data <- c(length(rivers), sum(rivers), mean(rivers), median(rivers), var(rivers),
      sd(rivers), min(rivers), max(rivers))
data
```

b. What are the results?

```
[1]   141.0000  83357.0000    591.1844    425.0000
[5] 243908.4086    493.8708    135.0000   3710.0000
```

c. Write the code and its outputs.

R code:

```
data <- c(length(rivers), sum(rivers), mean(rivers), median(rivers), var(rivers),
      sd(rivers), min(rivers), max(rivers))
data
```

output:

```
[1]   141.0000  83357.0000    591.1844    425.0000
[5] 243908.4086    493.8708    135.0000   3710.0000
```

8. The table below gives the 25 most powerful celebrities and their annual pay as ranked by the editions of Forbes magazine and as listed on the Forbes.com website.

Power Ranking	Celebrity Name	Pay	Power Ranking	Celebrity Name	Pay
1	Tom Cruise	67	14	Paul McCartney	40
2	Rolling Stones	90	15	George Lucas	233
3	Oprah Winfrey	225	16	Elton John	34
4	U2	110	17	David Letterman	40
5	Tiger Woods	90	18	Phil Mickelson	47
6	Steven Spielberg	332	19	J.K Rowling	75
7	Howard Stern	302	20	Bradd Pitt	25
8	50 Cent	41	21	Peter Jackson	39
9	Cast of the Sopranos	52	22	Dr. Phil McGraw	45
10	Dan Brown	88	23	Jay Lenon	32
11	Bruce Springsteen	55	24	Celine Dion	40
12	Donald Trump	44	25	Kobe Bryant	31
13	Muhammad Ali	55			

Figure 1: Forbes Ranking

a. Create vectors according to the above table. Write the codes.

R code:

```
power_ranking <- c(1:25)

celebrity_name <- c("Tom Cruise", "Rolling Stones", "Oprah Winfrey", "U2",
"Tiger Woods", "Steven Spielberg",

                    "Howard Stern", "50 Cent", "Cast  of the sopranos", "Dan Brown",
"Bruce Springsteen",

                    "Donal Trump", "Muhammad Ali", "Paul McCartney", "George Lucas",
"Elton John",

                    "David Letterman", "Phil Mickelson", "J.K Rowling", "Bradd Pitt",
"Peter Jackson",

                    "Dr. Phil McGrow", "J Lenon", "Celine Dion",

                    "Kobe Bryant")

pay <- c(67, 90, 225, 110, 90, 332, 302, 41, 52, 88, 55, 44, 55, 40, 233, 34, 40, 47,
75, 25, 39, 45, 32, 40, 31)
```

```
data_ranking <- data.frame(power_ranking, celebrity_name, pay)

data_ranking
```

b. Modify the power ranking and pay of J.K. Rowling. Change power ranking to 15 and pay to 90. Write the codes and its output.

R code:

```
 power_ranking [19] <- 15
power_ranking
pay [19] <- 90
pay
```

Output:

```
power_ranking [19] <- 15
 power_ranking
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
[18] 18 15 20 21 22 23 24 25
 pay [19] <- 90
 pay
 [1]  67  90 225 110  90 332 302  41  52  88  55  44  55
[14]  40 233  34  40  47  90  25  39  45  32  40  31

[5] 243908.4086    493.8708    135.0000   3710.0000
```

c. Interpret the data.