# Day 5 assessment

1. Write a R program to Create the following details

a. x= sample(-50:50, 10, replace=TRUE).and print the value of x

b. To create a sequence of numbers from 20 to 50 and find the mean of numbers from 20 to 50 and sum of numbers from 20 to 50.

CODE:

```
> # Part a
> x <- sample(-50:50, 10, replace = TRUE)
> print(x)
 [1]  27  14  38  31  13  -6  36  29 -28   5
> # Part b
> seq_numbers <- 20:50
> mean_numbers <- mean(seq_numbers)
> sum_numbers <- sum(seq_numbers)
> print(paste("Mean of numbers from 20 to 50:", mean_numbers))
[1] "Mean of numbers from 20 to 50: 35"
> print(paste("Sum of numbers from 20 to 50:", sum_numbers))
[1] "Sum of numbers from 20 to 50: 1085"
```

2. To create an array of two 3x3 matrices each with 3 rows and 3 columns from two given two

vectors.vector1 = c(1,3,4,5) and vector2 = c(10,11,12,13,14,15)

a. Print vector1, vector2

b. Print new array

CODE:

```
# Create the vectors
> vector1 <- c(1, 3, 4, 5)
> vector2 <- c(10, 11, 12, 13, 14, 15)
> # Part a
> print(vector1)
[1] 1 3 4 5
> print(vector2)
[1] 10 11 12 13 14 15
> # Part b
> matrix1 <- matrix(vector1, nrow = 3, ncol = 3, byrow = TRUE)
Warning message:
In matrix(vector1, nrow = 3, ncol = 3, byrow = TRUE) :
  data length [4] is not a sub-multiple or multiple of the number of rows [3]
> matrix2 <- matrix(vector2, nrow = 3, ncol = 3, byrow = TRUE)
Warning message:
In matrix(vector2, nrow = 3, ncol = 3, byrow = TRUE) :
  data length differs from size of matrix: [6 != 3 x 3]
> array12 <- array(c(matrix1, matrix2), dim = c(3, 3, 2))
> print(array12)
, , 1
```

```
     [,1] [,2] [,3]
[1,]    1    3    4
[2,]    5    1    3
[3,]    4    5    1

, , 2

     [,1] [,2] [,3]
[1,]   10   11   12
[2,]   13   14   15
[3,]   10   11   12
```

3. Write a R program to merge two given lists into one list. n1 = list (1,2,3) c1 = list("Raja",

"Rani", "Prince")

i) Write a R program to convert a given list to vector.n1 = list (1,2,3)c1 = list(4,5,6)

CODE:

```
> # Create the lists
> n1 <- list(1, 2, 3)
> c1 <- list("Raja", "Rani", "Prince")
> # Merge the lists
> merged_list <- c(n1, c1)
> print(merged_list)
[[1]]
[1] 1

[[2]]
[1] 2

[[3]]
[1] 3

[[4]]
[1] "Raja"

[[5]]
[1] "Rani"

[[6]]
[1] "Prince"

> # Create the list
> n1 <- list(1, 2, 3)
> c1 <- list(4, 5, 6)
> # Convert the list to a vector
> n1_vector <- unlist(n1)
> c1_vector <- unlist(c1)
> print(n1_vector)
[1] 1 2 3
> print(c1_vector)
[1] 4 5 6
```

4. Consider A=matrix(c(2,0,1,3),ncol=2) and B=matrix(c(5,2,4,-1), ncol=2).

a) Find A + B b) Find A – B c) Find A * B d) Find 3A + 3B

CODE:

```
# Create the matrices
> A <- matrix(c(2, 0, 1, 3), ncol = 2)
> B <- matrix(c(5, 2, 4, -1), ncol = 2)
> # Part a: A + B
> sum_AB <- A + B
> print(sum_AB)
     [,1] [,2]
[1,]    7    5
[2,]    2    2
> # Part b: A - B
> diff_AB <- A - B
> print(diff_AB)
     [,1] [,2]
[1,]   -3   -3
[2,]   -2    4
> # Part c: A * B
> prod_AB <- A %*% B
> print(prod_AB)
     [,1] [,2]
[1,]   12    7
[2,]    6   -3
> # Part d: 3A + 3B
> threeA_threeB <- 3 * A + 3 * B
> print(threeA_threeB)
     [,1] [,2]
[1,]   21   15
[2,]    6    6
```

5. Write a nested loop, where the outer for() loop increments “a” 3 times, and the inner for() loop

increments “b” 3 times. The break statement exits the inner for() loop after 2 incrementations. The

nested loop prints the values of variables, “a” and “b“.

CODE:

```
> for(a in 1:3){
+   for(b in 1:3){
+     if(b > 2) {
+       break
+     }
+     print(paste("a =", a, "b =", b))
+   }
+ }
[1] "a = 1 b = 1"
[1] "a = 1 b = 2"
[1] "a = 2 b = 1"
[1] "a = 2 b = 2"
[1] "a = 3 b = 1"
[1] "a = 3 b = 2"
```

6. (a) Suppose we have a fruit basket with 20 apples. Store the number of apples in a variable

my_apples.

(b) Every tasty fruit basket needs oranges, so we decide to add six oranges. As a data analyst , the

reflex is to immediately create a variable my_oranges and assign the value 6 to it. Next , calculate how

many pieces of fruit we have in total in the variable my_fruit.

CODE:

```
> # Part (a)
> my_apples <- 20
> # Part (b)
> my_oranges <- 6
> my_fruit <- my_apples + my_oranges
> # Printing the number of apples, oranges, and total fruits
> cat("Number of apples:", my_apples, "\n")
Number of apples: 20
> cat("Number of oranges:", my_oranges, "\n")
Number of oranges: 6
> cat("Total number of fruits:", my_fruit, "\n")
Total number of fruits: 26
```

7. Perform the following operations using R:

a. Initialize 3 character variables named age,employed and salary.

b. Transform age to numeric type and store in the variable age_clean.

c. Initialize employed_clean with the result obtained by converting employed to logical type.

d. Convert the respondent's salary to a numeric and store it in the variable salary_clean.

CODE:

```
> # Part (a)
> age <- "25"
> employed <- "TRUE"
> salary <- "$5000"
>
> # Part (b)
> age_clean <- as.numeric(age)
> age_clean
[1] 25
> # Part (c)
> employed_clean <- as.logical(employed)
> employed_clean
[1] TRUE
> # Part (d)
> salary_clean <- as.numeric(gsub("[^0-9]", "", salary))
> salary_clean
[1] 5000
```

8. Create the following vectors in R.

a = (5,10, 15, 20, ..., 160)

b = (87, 86, 85, ..., 56)

Use vector arithmetic to multiply these vectors and call the result d. Select subsets of d to identify the

following.

(a) What are the 19th, 20th, and 21st elements of d?

(b) What are all of the elements of d which are less than 2000?

(c) How many elements of d are greater than 6000?

CODE:

```
> # Create vector a
> a <- seq(from = 5, to = 160, by = 5)
> # Create vector b
> b <- rev(seq(from = 56, to = 87))
> # Multiply vectors a and b
> d <- a * b
> # (a) Subset of d for 19th, 20th, and 21st elements
> d[19:21]
[1] 6555 6800 7035
> # (b) Subset of d for elements less than 2000
> d[d < 2000]
[1]  435  860 1275 1680
> # (c) Count of elements in d greater than 6000
> sum(d > 6000)
[1] 16
```

9. You have an employee data-set, which comprises of two columns->”name” and designation”, add

a third column which would indicate the current date and time.

This is the employee data-set:

Output:

CODE:

```
> correct_answers <- 0
> while (correct_answers < 5) {
+   num1 <- sample(2:12, 1)
+   num2 <- sample(2:12, 1)
+   cat("What is", num1, "*", num2, "?\n")
+   answer <- as.integer(readline())
+   if (answer == num1 * num2) {
```

```
+    cat("Correct!\n")
+    correct_answers <- correct_answers + 1
+  } else {
+    cat("Incorrect. The correct answer is", num1 * num2, "\n")
+  }
+ }
What is 12 * 9 ?
cat("Congratulations, you answered 5 questions correctly!")
```

10. Implement a multiplication game. A while loop that gives the user two random numbers from 2 to

12 and asks the user to multiply them. Only exit the loop after five correct answers. Try using

as.integer(readline())

CODE:

```
> correct_answers <- 0
> while (correct_answers < 5) {
+   num1 <- sample(2:12, 1)
+   num2 <- sample(2:12, 1)
+   cat("What is", num1, "*", num2, "?\n")
+   answer <- as.integer(readline())
+   if (answer == num1 * num2) {
+     cat("Correct!\n")
+     correct_answers <- correct_answers + 1
+   } else {
+     cat("Incorrect. The correct answer is", num1 * num2, "\n")
+   }
+ }
What is 12 * 9 ?
cat("Congratulations, you answered 5 questions correctly!")
```

11. Create a Attendance sheet of the course “R Programming”.All are present for the course and

total strength of the students is 30. There are 15 male students register number from 191611258

to 191611272 and 15 female students of Register number from 191611273 to 191611287. Use

data frames to create the Attendance Sheet.(Refer the Sample attendance sheet for 6 students is

given below)

S ample Attendance Sheet

regno gender attendance

1 191611258 MALE PRESENT

2 191611259 MALE PRESENT

3 191611260 MALE PRESENT

4 191611261 FEMALE PRESENT

5 191611262 FEMALE PRESENT

6 191611263 FEMALE PRESENTK

CODE:

```
> # Create a data frame for male students
> male_regno <- 191611258:191611272
> male_attendance <- rep("PRESENT", 15)
> male_df <- data.frame(regno = male_regno, gender = "MALE", attendance = male_attendance)
> # Create a data frame for female students
> female_regno <- 191611273:191611287
> female_attendance <- rep("PRESENT", 15)
> female_df <- data.frame(regno = female_regno, gender = "FEMALE", attendance = female_attendanc
e)
> # Combine the two data frames
> attendance_sheet <- rbind(male_df, female_df)
> # Print the attendance sheet
> print(attendance_sheet)
      regno gender attendance
1  191611258   MALE    PRESENT
2  191611259   MALE    PRESENT
3  191611260   MALE    PRESENT
4  191611261   MALE    PRESENT
5  191611262   MALE    PRESENT
6  191611263   MALE    PRESENT
7  191611264   MALE    PRESENT
8  191611265   MALE    PRESENT
9  191611266   MALE    PRESENT
10 191611267   MALE    PRESENT
11 191611268   MALE    PRESENT
12 191611269   MALE    PRESENT
13 191611270   MALE    PRESENT
14 191611271   MALE    PRESENT
15 191611272   MALE    PRESENT
16 191611273 FEMALE    PRESENT
17 191611274 FEMALE    PRESENT
18 191611275 FEMALE    PRESENT
19 191611276 FEMALE    PRESENT
20 191611277 FEMALE    PRESENT
21 191611278 FEMALE    PRESENT
22 191611279 FEMALE    PRESENT
23 191611280 FEMALE    PRESENT
24 191611281 FEMALE    PRESENT
25 191611282 FEMALE    PRESENT
26 191611283 FEMALE    PRESENT
27 191611284 FEMALE    PRESENT
28 191611285 FEMALE    PRESENT
29 191611286 FEMALE    PRESENT
30 191611287 FEMALE    PRESENT
```

12. Create two vectors named v and w with the following contents:

v :21,55,84,12,13,15

w : 9,44,22,33,14,35

A) Print the length of the vectors B) Print all elements of the vectors

C) Print the sum of the elements in each vector. D)Find the mean of each vector. (Use R's mean() function)

E) Add vectors v and w. F) Multiply vectors v and w.

G) In vector v select all elements that are greater than 2.

H) In vector w select all elements that are less than 20.

CODE:

```
# Create the vectors v and w
> v <- c(21, 55, 84, 12, 13, 15)
> w <- c(9, 44, 22, 33, 14, 35)
> # A) Print the length of the vectors
> cat("Length of v:", length(v), "\n")
Length of v: 6
> cat("Length of w:", length(w), "\n\n")
Length of w: 6

> # B) Print all elements of the vectors
> cat("Elements of v:", v, "\n")
Elements of v: 21 55 84 12 13 15
> cat("Elements of w:", w, "\n\n")
Elements of w: 9 44 22 33 14 35

> # C) Print the sum of the elements in each vector
> cat("Sum of v:", sum(v), "\n")
Sum of v: 200
> cat("Sum of w:", sum(w), "\n\n")
Sum of w: 157

> # D) Find the mean of each vector
> cat("Mean of v:", mean(v), "\n")
Mean of v: 33.33333
> cat("Mean of w:", mean(w), "\n\n")
Mean of w: 26.16667

> # E) Add vectors v and w
> cat("Sum of v and w:", v + w, "\n\n")
Sum of v and w: 30 99 106 45 27 50

> # F) Multiply vectors v and w
> cat("Product of v and w:", v * w, "\n\n")
Product of v and w: 189 2420 1848 396 182 525

> # G) In vector v select all elements that are greater than 2
> cat("Elements of v greater than 2:", v[v > 2], "\n\n")
```

Elements of v greater than 2: 21 55 84 12 13 15

```
> # H) In vector w select all elements that are less than 20
> cat("Elements of w less than 20:", w[w < 20], "\n\n")
Elements of w less than 20: 9 14
```

13. lapply function is applied to all elements of the input and it returns a list and saaply function is

applied to all elements of the input and it returns a vector. Demonstrate the use of sapply and

lapply with the following vector.

movies<-c("SPYDERMAN","BATMAN","VERTIGO","CHINATOWN")

Convert these elements of vector into lowercase letters.

CODE:

```
movies <- c("SPYDERMAN", "BATMAN", "VERTIGO", "CHINATOWN")
> # Using lapply to convert elements to lowercase and return a list
> movies_lower_l <- lapply(movies, tolower)
> # Using sapply to convert elements to lowercase and return a vector
> movies_lower_s <- sapply(movies, tolower)
> # Print the results
> cat("Using lapply:", movies_lower_l, "\n")
Using lapply: Error in cat("Using lapply:", movies_lower_l, "\n") :
  argument 2 (type 'list') cannot be handled by 'cat'
> cat("Using sapply:", movies_lower_s, "\n")
Using sapply: spyderman batman vertigo chinatown
> Using lapply: spyderman batman vertigo chinatown
Error: unexpected symbol in "Using lapply"
> Using sapply: spyderman batman vertigo chinatown
Error: unexpected symbol in "Using sapply"
```

14. Create dataframe dataframe1 with the following vectors,

Mark1=c(35,45,67)

Mark2=c(56,89,99)

Mark3=c(78,75,83)

Use sapply and lapply function to find minimum marks ,maximum mark and average of all marks

CODE:

```
> # Create the dataframe
> dataframe1 <- data.frame(Mark1 = c(35, 45, 67),
+                          Mark2 = c(56, 89, 99),
+                          Mark3 = c(78, 75, 83))
> # Using lapply to find minimum, maximum, and average of all marks
> lapply(dataframe1, function(x) c(min = min(x), max = max(x), mean = mean(x)))
$Mark1
 min  max mean
  35   67   49

$Mark2
    min      max     mean
56.00000 99.00000 81.33333

$Mark3
    min      max     mean
75.00000 83.00000 78.66667

> # Using sapply to find minimum, maximum, and average of all marks
> sapply(dataframe1, function(x) c(min = min(x), max = max(x), mean = mean(x)))
     Mark1    Mark2    Mark3
min     35 56.00000 75.00000
max     67 99.00000 83.00000
mean    49 81.33333 78.66667
> # Using lapply
> $Mark1
Error: unexpected '$' in "$"
> min   max  mean
Error: unexpected symbol in "min   max"
> 35.00 67.00 49.00
Error: unexpected numeric constant in "35.00 67.00"
> $Mark2
Error: unexpected '$' in "$"
> min   max  mean
Error: unexpected symbol in "min   max"
> 56.00 99.00 81.33
Error: unexpected numeric constant in "56.00 99.00"
> $Mark3
Error: unexpected '$' in "$"
> min   max  mean
Error: unexpected symbol in "min   max"
> 75.00 83.00 78.67
Error: unexpected numeric constant in "75.00 83.00"
> # Using sapply
> Mark1 Mark2 Mark3
Error: unexpected symbol in "Mark1 Mark2"
> min  35.00 56.00 75.00
Error: unexpected numeric constant in "min  35.00"
> max  67.00 99.00 83.00
Error: unexpected numeric constant in "max  67.00"
> mean 49.00 81.33 78.67
Error: unexpected numeric constant in "mean 49.00"
```

15. Write a R Program :

a. To find the multiplication table (from 1 to 10)

b. To find factorial of number

c. To check if the input number is odd or even

d. To check if the input number is prime or not

e. To find sum of natural numbers up-to 10, without formula using loop statement

CODE:

```
# Using a loop to calculate the sum of natural numbers
> sum <- 0
> for(i in 1:10) {
+   sum <- sum + i
+ }
> # Print the result
> cat("The sum of natural numbers up to 10 is:", sum)
The sum of natural numbers up to 10 is: 55


# Using a loop to calculate the factorial
> factorial <- function(n) {
+   result <- 1
+   for(i in 1:n) {
+     result <- result * i
+   }
+   return(result)
+ }
> # Example usage:
> factorial(5) # Returns 120
[1] 120
> # Using modulo operator to check if number is even or odd
> is_even <- function(n) {
+   if(n %% 2 == 0) {
+     return(TRUE)
+   } else {
+     return(FALSE)
+   }
+ }
> # Example usage:
> is_even(4) # Returns TRUE
[1] TRUE
> is_even(5) # Returns FALSE
[1] FALSE
> # Using a loop to check if number is prime
> is_prime <- function(n) {
+   if(n == 1) {
+     return(FALSE)
+   }
+   for(i in 2:(n-1)) {
+     if(n %% i == 0) {
+       return(FALSE)
+     }
+   }
+   return(TRUE)
+ }
> # Example usage:
> is_prime(7) # Returns TRUE
[1] TRUE
> is_prime(8) # Returns FALSE
```

```
[1] FALSE
> # Using a loop to calculate the sum of natural numbers
> sum <- 0
> for(i in 1:10) {
+   sum <- sum + i
+ }
> # Print the result
> cat("The sum of natural numbers up to 10 is:", sum)
The sum of natural numbers up to 10 is: 55
```

16. a. Create a data frame from four given vectors.

name =c ('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin',

'Jonas')

score = c (12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19)

attempts =c (1, 3, 2, 3, 2, 3, 1, 1, 2, 1)

qualify = c ('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

b. Write a R program to extract first two rows from a given data frame.

c. Write a R program to extract 3rd and 5th rows with 1st and 3rd columns from a given data frame

d. Find the average score with respect to first, second, and third attempts. Don't use any special in

build function for this task.

e. Write a R program to create a list containing a vector, a matrix and a list and give names to the

elements in the list. Access and print the first and second element of the list

CODE:

```
a)name <- c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas')

score <- c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19)

attempts <- c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1)

qualify <- c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')


dataframe1 <- data.frame(name, score, attempts, qualify)

b)first_two_rows <- dataframe1[1:2,]
```

```
c)third_fifth_rows <- dataframe1[c(3,5), c(1,3)]
d)# Using for loop
avg_score <- numeric(3)
for(i in 1:3) {
  avg_score[i] <- mean(dataframe1$score[dataframe1$attempts == i])
}

# Using sapply function
avg_score <- sapply(1:3, function(x) mean(dataframe1$score[dataframe1$attempts == x]))
e)my_list <- list(vec = c(1,2,3), mat = matrix(1:6, nrow = 2), lst = list("a", "b", "c"))
names(my_list) <- c("vector", "matrix", "list")

# Accessing the first and second elements
print(my_list$vector)
print(my_list$matrix)
```