1. Write a R program to Create the following details

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

code:

```
v = sample(-50:50, 10, replace=TRUE)
print("Content of the vector:")
print("10 random integer values between -50 and +50:")
print(v)
```

OUTPUT

```
[1] "Content of the vector:"
[1] "10 random integer values between -50 and +50:"
 [1]  31 -13 -21  42  49 -39  20  12  39  -2
```

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:

```
print("Sequence of numbers from 20 to 50:")
print(seq(20,50))
print("Mean of numbers from 20 to 50:")
print(mean(20:50))
print("Sum of numbers from 20 to 50:")
print(sum(20:50))
```

OUTPUT

```
[1] "Sequence of numbers from 20 to 50:"
```

```
[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 44 45 46 47 48 49 50

[1] "Mean of numbers from 20 to 50:"

[1] 35

[1] "Sum of numbers from 20 to 50:"

[1] 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:

```
print("Two vectors of different lengths:")

v1 =  c(1,3,4,5)

v2 =  c(10,11,12,13,14,15)

print(v1)

print(v2)

result = array(c(v1,v2),dim = c(3,3,2))

print("New array:")

print(result)
```

OUTPUT

```
[1] "Two vectors of different lengths:"

[1] 1 3 4 5

[1] 10 11 12 13 14 15

[1] "New array:"

, , 1
```

```
     [,1] [,2] [,3]
[1,]    1    5   12
[2,]    3   10   13
[3,]    4   11   14

, , 2

     [,1] [,2] [,3]
[1,]   15    4   11
[2,]    1    5   12
[3,]    3   10   13
```

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

&quot;Rani&quot;, &quot;Prince&quot;)

code:

```
n1 = list(1,2,3)
c1 = list("Raja", "Rani", "Prince")
print("Original lists:")
print(n1)
print(c1)
print("Merge the said lists:")
mlist =  c(n1, c1)
print("New merged list:")
print(mlist)
```

OUTPUT

```
[1] "Merge the said lists:"
[1] "New merged list:"
[[1]]
[1] 1

[[2]]
[1] 2

[[3]]
[1] 3

[[4]]
[1] "Raja"

[[5]]
[1] "Rani"

[[6]]
[1] "Prince"
```

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

code:

```
n1 = list(1,2,3)
c1 = list(4,5,6)
print("Original lists:")
print(n1)
print(c1)
print("Convert the lists to vectors:")
```

```
v1 = unlist(n1)

v2 = unlist(c1)

print(v1)

print(v2)

print("Add two vectors:")

v = v1 + v2

print("New vector:")

print(v)
```

OUTPUT

```
[1] "Original lists:"

[[1]]

[1] 1


[[2]]

[1] 2


[[3]]

[1] 3


[[1]]

[1] 4


[[2]]

[1] 5


[[3]]

[1] 6


[1] "Convert the lists to vectors:"
```

```
[1] 1 2 3

[1] 4 5 6

[1] "Add two vectors:"

[1] "New vector:"

[1] 5 7 9
```

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

a) code:

```
A = matrix(c(2,0,1,3), ncol=2)

B = matrix(c(5,2,4,-1), ncol=2)

A+B
```

OUTPUT

```
##      [,1] [,2]

## [1,]    7    5

## [2,]    2    2
```

b) code:

```
A = matrix(c(2,0,1,3), ncol=2)

B = matrix(c(5,2,4,-1), ncol=2)

A-B
```

OUTPUT

```
##      [,1] [,2]
## [1,]   -3   -3
## [2,]   -2    4
```

c) code:

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

A*B
```

OUTPUT

```
[,1] [,2]
[1,]   10    4
[2,]    0   -3
```

d) code:

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

3A + 3B
```

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))
   }
}
```

OUTPUT

```
[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:

```
# Assign a value to the variables my_apples and my_oranges
my_apples <- 5
```

```
my_oranges <- 6

# Add these two variables together
my_apples + my_oranges

# Create the variable my_fruit
my_fruit <- my_apples + my_oranges
```

OUTPUT

```
> my_apples <- 5
> my_oranges <- 6
>
> my_apples + my_oranges
[1] 11
>
> my_fruit <- my_apples + my_oranges
> my_apples <- 5
> my_oranges <- 6
>
> my_apples + my_oranges
[1] 11
>
> my_fruit <- my_apples + my_oranges
>

> my_fruit
[1] 11
```

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
>  Part (c)
> employed_clean <- as.logical(employed)
> employed_clean
> # Part (d)
> salary_clean <- as.numeric(gsub("[^0-9]", "", salary))
> salary_clean
```

OUTPUT

```
[1] 25
[1] TRUE
[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:

```
a<-seq(from=5 ,to=160, by=5) # Create a vector a
print(a)
length(a)
b<-seq(from=87,to=56,by=-1)# Create a vector b
print(b)
length(b)
d<-a*b # Use vector arithmetic to multiply these vectors and call the result 'd'.
print(d)
d<-a*b # Use vector arithmetic to multiply these vectors and call the result 'd'.
print(d)
j<-d<2000 #What are all of the elements of d which are less than 2000?
d[j]
k<-d>6000 # How many elements of d are greater than 6000?
length(d[k])
```

OUTPUT

```
[1]   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80  85
[18]  90  95 100 105 110 115 120 125 130 135 140 145 150 155 160
[1] 32
```

```
[1] 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65

[24] 64 63 62 61 60 59 58 57 56

[1] 32

[1]  435  860 1275 1680 2075 2460 2835 3200 3555 3900 4235 4560 4875 5180

[15] 5475 5760 6035 6300 6555 6800 7035 7260 7475 7680 7875 8060 8235 8400

[29] 8555 8700 8835 8960

[1] 6555 6800 7035

[1]  435  860 1275 1680

[1] 16
```

9. You have an employee data-set, which comprises of two columns-&gt;"name" and designation", add

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

This is the employee data-set:

code:

```
employee <- data.frame(

  name = c("john", "sam", "raj", "amy", "anne"),

  designation = c("ceo", "ceo", "sde", "coo", "analyst")

)


employee$datetime <- Sys.time()


print(employee)
```

ouput:

```
name designation           datetime

1  john         ceo 2023-03-24 14:25:12
```

```
2   sam        ceo 2023-03-24 14:25:12
3   raj        sde 2023-03-24 14:25:12
4   amy        coo 2023-03-24 14:25:12
5  anne    analyst 2023-03-24 14:25:12
```

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, "x", num2, "? ")
  answer <- as.integer(readline())
  if (answer == num1 * num2) {
    correct_answers <- correct_answers + 1
    cat("Correct!\n")
  } else {
    cat("Incorrect. The correct answer is", num1 * num2, "\n")
  }
}
cat("Congratulations, you got 5 correct answers!\n")
```

output:

```
What is 3 x 7? 21
```

```
Correct!
What is 10 x 9? 90
Correct!
What is 6 x 6? 35
Incorrect. The correct answer is 36
What is 2 x 11? 22
Correct!
What is 8 x 4? 32
Correct!
Congratulations, you got 5 correct answers!
```

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)

CODE:

```
> male_regno <- 191611258:191611272
> male_attendance <- rep("PRESENT", 15)
> male_df <- data.frame(regno = male_regno, gender = "MALE", attendance = male_attendance)

> female_regno <- 191611273:191611287
> female_attendance <- rep("PRESENT", 15)
> female_df <- data.frame(regno = female_regno, gender = "FEMALE", attendance = female_attendance)

> attendance_sheet <- rbind(male_df, female_df)
> print(attendance_sheet)
```

output:

```
       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

```
v <- c(21, 55, 84, 12, 13, 15)
w <- c(9, 44, 22, 33, 14, 35)
```

B) Print all elements of the vectors

```
cat("Length of v:", length(v), "\n")
cat("Length of w:", length(w), "\n\n")
```

C) Print the sum of the elements in each vector.

D)Find the mean of each vector. (Use R&#39;s mean() function)

```
cat("Elements of v:", v, "\n")
cat("Elements of w:", w, "\n\n")
```

E) Add vectors v and w.

```
cat("Sum of elements in v:", sum(v), "\n")
cat("Sum of elements in w:", sum(w), "\n\n")

cat("Mean of v:", mean(v), "\n")
cat("Mean of w:", mean(w), "\n\n")
```

F) Multiply vectors v and w.

```
cat("v + w:", v + w, "\n\n")



cat("v * w:", v * w, "\n\n")
```

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

```
cat("Elements in v greater than 2:", v[v > 2], "\n\n")
```

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

```
cat("Elements in w less than 20:", w[w < 20], "\n\n")
```

output:

```
Length of v: 6
Length of w: 6

Elements of v: 21 55 84 12 13 15
Elements of w: 9 44 22 33 14 35

Sum of elements in v: 200
Sum of elements in w: 157

Mean of v: 33.33333
Mean of w: 26.16667

v + w: 30 99 106 45 27 50

v * w: 189 2420 1848 396 182 525

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

Elements in 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&lt;-
c(&quot;SPYDERMAN&quot;,&quot;BATMAN&quot;,&quot;VERTIGO&quot;,&quot;CHINATOWN&qu
ot;)

Convert these elements of vector into lowercase letters.

CODE:

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



lowercase_movies1 <- lapply(movies, tolower)

print(lowercase_movies1)



lowercase_movies2 <- sapply(movies, tolower)

print(lowercase_movies2)
```

output:

```
[[1]]
[1] "spyderman"

[[2]]
[1] "batman"

[[3]]
[1] "vertigo"
```

```
[[4]]
[1] "chinatown"
```

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:

```
dataframe1 <- data.frame(
  Mark1 = c(35, 45, 67),
  Mark2 = c(56, 89, 99),
  Mark3 = c(78, 75, 83)
)

sapply(dataframe1, min)  # minimum marks for each subject
sapply(dataframe1, max)  # maximum marks for each subject
sapply(dataframe1, mean) # average marks for each subject



lapply(dataframe1, min)  # minimum marks for each subject
lapply(dataframe1, max)  # maximum marks for each subject
lapply(dataframe1, mean) # average marks for each subjec
```

output:

```
[[1]]
[1] "spyderman"

[[2]]
[1] "batman"
```

```
[[3]]
[1] "vertigo"


[[4]]
[1] "chinatown"
```

15. Write a R Program :

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

```
rows <- 1:10
cols <- 1:10



for (i in rows) {
  for (j in cols) {
    result <- i * j
    cat(result, "\t")
  }
  cat("\n")
}
```

output:

```
1       2       3       4       5       6       7       8       9       10
2       4       6       8       10      12      14      16      18      20
3       6       9       12      15      18      21      24      27      30
4       8       12      16      20      24      28      32      36      40
5       10      15      20      25      30      35      40      45      50
6       12      18      24      30      36      42      48      54      60
7       14      21      28      35      42      49      56      63      70
8       16      24      32      40      48      56      64      72      80
9       18      27      36      45      54      63      72      81      90
```

```
10      20      30      40      50      60      70      80      90      100
```

b. To find factorial of number

```
num <- 5

fact <- 1

for (i in 1:num) {
  fact <- fact * i
}

cat("The factorial of", num, "is", fact)
```

output:

```
The factorial of 5 is 120
```

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

```
num <- 7
if (num %% 2 == 0) {
  cat(num, "is even")
} else {
  cat(num, "is odd")
}
```

output:

```
7 is odd
```

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

```
num <- 17


count <- 0


for (i in 2:num-1) {
  if (num %% i == 0) {
    count <- count + 1
  }
}
if (count > 0) {
  cat(num, "is not a prime number")
} else {
  cat(num, "is a prime number")
}
```

output:

```
17 is a prime number
```

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

```
num <- 10

sum <- 0


for (i in 1:num) {
  sum <- sum + i
}
```

```
cat("The sum of natural numbers up to", num, "is", sum)
```

output:

```
The sum of natural numbers up to 10 is 55
```

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

```
name =c (&#39;Anastasia&#39;, &#39;Dima&#39;, &#39;Katherine&#39;, &#39;James&#39;,
&#39;Emily&#39;, &#39;Michael&#39;, &#39;Matthew&#39;, &#39;Laura’, &#39;Kevin&#39;,
&#39;Jonas&#39;)
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 (&#39;yes&#39;, &#39;no&#39;, &#39;yes&#39;, &#39;no&#39;, &#39;no&#39;,
&#39;yes&#39;, &#39;yes&#39;, &#39;no&#39;, &#39;no&#39;, &#39;yes&#39;)
```

code:

```
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')

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

output:

```
       name score attempts qualify
1  Anastasia  12.5        1     yes
2       Dima   9.0        3      no
3  Katherine  16.5        2     yes
```

```
4     James  12.0        3      no
5     Emily   9.0        2      no
6    Michael  20.0        3     yes
7    Matthew  14.5        1     yes
8     Laura  13.5        1      no
9     Kevin   8.0        2      no
10    Jonas  19.0        1     yes
```

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

code:

```
first_two <- data[1:2, ]
print(first_two)
```

output:

```
      name score attempts qualify
1 Anastasia  12.5        1     yes
2      Dima   9.0        3      no
```

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

code:

```
subset_data <- data[c(3, 5), c(1, 3)]
print(subset_data)
```

output:

```
      name attempts
3 Katherine        2
5     Emily        2
```

d. Find the average score with respect to first, second, and third attempts. Don’t use any special in build function for this task.

code:

```
first_attempt_scores <- data[attempts == 1, "score"]
```

```
first_attempt_avg <- sum(first_attempt_scores) / length(first_attempt_scores)
cat("Average score for first attempts:", first_attempt_avg, "\n")



second_attempt_scores <- data[attempts == 2, "score"]
second_attempt_avg <- sum(second_attempt_scores) / length(second_attempt_scores)
cat("Average score for second attempts:", second_attempt_avg, "\n")



third_attempt_scores <- data[attempts == 3, "score"]
third_attempt_avg <- sum(third_attempt_scores) / length(third_attempt_scores)
cat("Average score for third attempts:", third_attempt_avg)
```

output:

```
Average score for first attempts: 13.5
Average score for second attempts: 10.25
Average score for third attempts: 15.0
```

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:

```
create a vector
my_vector <- c(10, 20, 30, 40, 50)
my_matrix <- matrix(1:9, nrow = 3, ncol = 3)
my_list <- list(
  my_vector = my_vector,
  my_matrix = my_matrix,
  my_inner_list = list("John", "Doe", c("johndoe@example.com", "johndoe@gmail.com"))
)
print(my_list[[1]])
print(my_list[[2]])output:
```

Output:

```
$my_vector

[1] 10 20 30 40 50


$my_matrix

[,1] [,2] [,3]

[1,] 1 4 7

[2,] 2 5 8

[3,] 3 6 9


$my_list

$my_list[[1]]

[1] "John" "Doe"


$my_list[[2]]

[1] "johndoe@example.com" "johndoe@gmail.com"
```

Accessing first and second elements of the list:

```
$my_vector

[1] 10 20 30 40 50


$my_matrix

[,1] [,2] [,3]

[1,] 1 4 7

[2,] 2 5 8

[3,] 3 6 9
```