```
#1.Try to write a code for printing sequence of numbers from 1 to 50 with the differences of 3, 5, 10
> Seqby3 <<- seq(1, 50, by=3)
> Seqby5 = seq(1, 50, by=5)
> Seqby10 = seq(1, 50, by=10)
> print(Seqby3)
 [1]  1  4  7 10 13 16 19 22 25 28 31 34 37 40 43 46 49
> print(Seqby5)
 [1]  1  6 11 16 21 26 31 36 41 46
> print(Seqby10)
[1]  1 11 21 31 41
>
>
> #2. What are the different data objects in R? and write syntax and example for each and every object
>
> # Data Types/Object in R - Vector, List, Matrix, Array, Factor, Data Frame
>
> #vector : syntax: vector_name <<- c(element1,element2,...elementn)
> Vect <<- c(1,2,3,4,5)
> print(Vect)
[1] 1 2 3 4 5
> class(Vect)
[1] "numeric"
>
> #list : syntax: list_name <<- list(element1,element2,....elementn)
> Fruit <<- list(c("Apple","Mango","Banana","Grape"))
> print(Fruit)
[[1]]
[1] "Apple"  "Mango"  "Banana" "Grape"

> class(Fruit)
[1] "list"
>
> #Matrix: syntax: matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
> m <- matrix(c(1,2,3,4,5,6,7,8,9), nrow = 3, ncol = 3, byrow = TRUE,
+             dimnames = list(c('r1', 'r2','r3'), c('c1', 'c2','c3')))
> print(m)
   c1 c2 c3
r1  1  2  3
r2  4  5  6
r3  7  8  9
> class(m)
[1] "matrix" "array"
>
> #Array: syntax: array_name <<- array(data=file,vector, dim = c(dem1,dem2), dimnames = null )
> ar <- array(c(1,2), dim = c(3, 3, 2))
> print(ar)
, , 1

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

, , 2
```

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

> class(ar)
[1] "array"
>
>
> # Factor: syntax: factor(vect_name)
> k <- c("red", "green", "blue", "blue", "red")
> class(k)
[1] "character"
> k
[1] "red"   "green" "blue"  "blue"  "red"
> factor_k <- factor(k) # creating vector using factor()
> class(factor_k)
[1] "factor"
> factor_k
[1] red   green blue  blue  red
Levels: blue green red
>
>
> # Data Frame: syntax: data.frame(coln1,coln2,...colnN)
> df <- data.frame(gender = c("Male", "Male","Female","Male","Female"), height = c(152, 171.5, 165,155,175),
+                  weight = c(81,93,78,65,95), Age = c(42,38,64,45,40))
> print(df)
  gender height weight Age
1   Male  152.0     81  42
2   Male  171.5     93  38
3 Female  165.0     78  64
4   Male  155.0     65  45
5 Female  175.0     95  40
> class(df)
[1] "data.frame"
>
> #3.Create Data frame with 3 columns and 5 rows and write a code to fetch and delete row and a column using inde
x and add new column and row to the existed data frame
> #Creating Data frame
> df1 <- data.frame(gender = c("Male", "Female","Female","Male","Female"), height = c(152, 171.5, 165,145,170),

+                   weight = c(81,55,58,70,65))
> print(df1)
  gender height weight
1   Male  152.0     81
2 Female  171.5     55
3 Female  165.0     58
4   Male  145.0     70
5 Female  170.0     65
>
> #Fetching data from df1
> df1[2,1:3]  #fetching 2nd row and all 3 coln
  gender height weight
2 Female  171.5     55
>
```

```
> #Deleting - Drop rows & coln using "-" operator in R
> df1 = df1[-c(5),-c(3)]
> df1
  gender height
1   Male  152.0
2 Female  171.5
3 Female  165.0
4   Male  145.0
>
> #adding New row using rbind()
> newrow = c("Female",130) # Two value because of two column are there in df
> df1 = rbind(df1,newrow)
> df1
  gender height
1   Male    152
2 Female  171.5
3 Female    165
4   Male    145
5 Female    130
>
> #adding New Column using cbind()
> age = c(40,39,38,27,45) # 5 value because of 5 observations
> df1 = cbind(df1,age)
> df1
  gender height age
1   Male    152  40
2 Female  171.5  39
3 Female    165  38
4   Male    145  27
5 Female    130  45
>
> # 4.Write nested if else statements to print whether the given number is negative, positive or Zero
> x <- 0
> if (x>0) {
+   print("Positive Number")
+ } else if (x<0) {
+   print("Negative Number")
+ } else if (x==0){
+   print("Nmberis Zero")
+ } else print("Not a Number")
[1] "Nmberis Zero"
>
> # 5.write a program to input any value and check whether it is character, numeric or special character
>
> checking <- function(read){
+   if (grepl("[[:alpha:]]",read)){
+     print("Input is a character")
+   }
+   else if (grepl("^[0-9]+$",read)){
+     print("Input is numeric")
+   }
+   else if (grepl("[[:punct:]]",read)){
+     print("Input is a Special character")
+   }
+   else
+     print("Wrong input")
+ }
>
> # Calling the function 'Checking'
> checking("Analyts")
[1] "Input is a character"
> checking(5)
[1] "Input is numeric"
> checking("#")
[1] "Input is a Special character"
>
>
> #6.write difference between break and next also write examples for both
>
> # Break :
> #1.terminates the loop statement then and there if the break conditions Satisfies.
> #2.Break statement basically terminates/stop the loop resulting in no further iterations of the loop.
> #3.Keyword for Break Statement is: break
> #4.Example:
> for (b in seq(10)){
+   if (b==5){
+     break      # will break at 4 since, b==5
+   }
+   print(b)
+ }
[1] 1
[1] 2
[1] 3
[1] 4
>
> #Next :
> #1.Skip a iteration then and there if the next condition satisfies.
> #2.Next statements basically skip a step/iteration and continues with further iterations
> #3.keyword for Next statement is: next
> #4.Example:
> for (n in seq(7)){
+   if (n==4){
+     next    # will next and skip 4
+   }
+   print(n)
+ }
[1] 1
[1] 2
[1] 3
[1] 5
[1] 6
[1] 7
>
>
> #7.write a program to print a given vector in reverse format x= c(1,5.6,3,10,3.5,5)
>
> x= c(1,5.6,3,10,3.5,5)
> print(rev(x))       # rev() prints the arguments in reverse order.
[1]  5.0  3.5 10.0  3.0  5.6  1.0
>
```

```
>
> #8.write a program to get the mode value of the given vector ('a','b','c','t','a','c','r','a','c','t','z','r','v','t','u','e','t')
>
> vect1 = c("a","b","c","t","a","c","r","a","c","t","z","r","v","t","u","e","t")
> #Creating a function for mode
> mode <- function(vec) {         #naming the function as mode and parameter
+   uni <- unique(vec)            #unique() - will gives unique value and storing it in 'uni'
+   uni[which.max(tabulate(match(vec, uni)))] #which.max() - gives the index of max element, tabulate() - counts th
e elements/values.
+ }
> #Calling the Function mode for vect1
> mode(vect1)
[1] "t"
>
>
> #9.Write a function to filter only data belongs to 'setosa' in species of Iris dataset.( using dplyr package)
> library(dplyr)         # importing the libary
> irisdf<-datasets::iris  # Loading iris dataset in irisdf variable
>
> #Creating a function/method to filter only data of setosa
> iris_setosa <-filter(irisdf,Species == "setosa")
> # calling the variable in which data of setosa species is stored
> iris_setosa
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1           5.1         3.5          1.4         0.2  setosa
2           4.9         3.0          1.4         0.2  setosa
3           4.7         3.2          1.3         0.2  setosa
4           4.6         3.1          1.5         0.2  setosa
5           5.0         3.6          1.4         0.2  setosa
6           5.4         3.9          1.7         0.4  setosa
7           4.6         3.4          1.4         0.3  setosa
8           5.0         3.4          1.5         0.2  setosa
9           4.4         2.9          1.4         0.2  setosa
10          4.9         3.1          1.5         0.1  setosa
11          5.4         3.7          1.5         0.2  setosa
12          4.8         3.4          1.6         0.2  setosa
13          4.8         3.0          1.4         0.1  setosa
14          4.3         3.0          1.1         0.1  setosa
15          5.8         4.0          1.2         0.2  setosa
16          5.7         4.4          1.5         0.4  setosa
17          5.4         3.9          1.3         0.4  setosa
18          5.1         3.5          1.4         0.3  setosa
19          5.7         3.8          1.7         0.3  setosa
20          5.1         3.8          1.5         0.3  setosa
21          5.4         3.4          1.7         0.2  setosa
22          5.1         3.7          1.5         0.4  setosa
23          4.6         3.6          1.0         0.2  setosa
24          5.1         3.3          1.7         0.5  setosa
25          4.8         3.4          1.9         0.2  setosa
26          5.0         3.0          1.6         0.2  setosa
27          5.0         3.4          1.6         0.4  setosa
28          5.2         3.5          1.5         0.2  setosa
29          5.2         3.4          1.4         0.2  setosa
30          4.7         3.2          1.6         0.2  setosa
31          4.8         3.1          1.6         0.2  setosa
```

```
32          5.4         3.4         1.5         0.4  setosa
33          5.2         4.1         1.5         0.1  setosa
34          5.5         4.2         1.4         0.2  setosa
35          4.9         3.1         1.5         0.2  setosa
36          5.0         3.2         1.2         0.2  setosa
37          5.5         3.5         1.3         0.2  setosa
38          4.9         3.6         1.4         0.1  setosa
39          4.4         3.0         1.3         0.2  setosa
40          5.1         3.4         1.5         0.2  setosa
41          5.0         3.5         1.3         0.3  setosa
42          4.5         2.3         1.3         0.3  setosa
43          4.4         3.2         1.3         0.2  setosa
44          5.0         3.5         1.6         0.6  setosa
45          5.1         3.8         1.9         0.4  setosa
46          4.8         3.0         1.4         0.3  setosa
47          5.1         3.8         1.6         0.2  setosa
48          4.6         3.2         1.4         0.2  setosa
49          5.3         3.7         1.5         0.2  setosa
50          5.0         3.3         1.4         0.2  setosa
>
>
>
> #10.Create a new column for iris dataset with the name of Means_of_obs, which contains mean value of each row.
( using dplyr package)
> #importing Dataset
> df_iris<-datasets::iris
> #Creating an array of means of first 4 colums by using apply & mean Function
> means = apply(df_iris[1:4],1,mean)
> #appending the array of mean at the end of the dataframe
> df_new_column = mutate(df_iris,Means_of_obs=means)
> #Creating View instant
> View(df_new_column)
>
>
>
> #11.Filter data for the "versicolor" and get only 'sepel_length' and Sepel _width' columns.( using dplyr package)
> # Loading iris dataset in irisdf variable
> irisdf<-datasets::iris
> #filtering using subset function
> df_versicolor = subset(irisdf, Species == "versicolor")[,1:2]
> #Creating View instant
> View(df_versicolor)
>
> #12.create below plots for the mtcars also write your inferences for each and every plot (use ggplot package) Use
Different ( Size , Colour )
> #Creating DataFrame of mtcars dataset
> df_mtcars<-datasets::mtcars
> #getting a list of columns names
> colnames(df_mtcars)
 [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear" "carb"
> library("ggplot2")
> #1.create a scatter plot
> ggplot(data = mtcars,aes(x = mpg,y = hp)) + geom_point(color ="Red", size=2)
>
> # inference: As the mpg of the Cars increase, hp(horsePower) decrease it is Strongly negatively correlated
```

```
>
> #2.create a box plot
> ggplot(data = mtcars,aes(x = wt)) + geom_boxplot(color ="Green", size = 2) + coord_flip()
> # inference: There are 3 outliers (Extreme Observation) in the wt category
>
> #3.create a Histogram
> ggplot(mtcars,aes(mpg)) + geom_histogram(fill = "Red",color ="Black", bins = 5,size = 1)
> # inference: Histogram of MPG Doesn't follow Normal Distribution, it is also Postively Skewed
>
> #4.create a line Graph
> ggplot(data = mtcars,aes(x = mpg,y = hp)) + geom_line(color = "Red", size = 2)
> # inference: As the mpg of the Cars increase, hp(horsePower) decrease it is Strongly negatively correlated
>
> #5.create a Bar Graph
> ggplot(data = mtcars,aes(x = gear)) + geom_bar(color ="Black",fill = "NavyBlue", size = 1)
> # inference: Through the bar graph it can be there are 15, 12 and 5 cars with 3, 4 and 5 gears respectively
>
```