1. Create a vector in R and perform the following operations:
    a. maximum and minimum
    b. levels of factor
    c. sort a vector

```
#Q1
temp <- c(3,2,1,7,4,5,6)

#a
print(max(temp))
print(min(temp))

#b
directions <- factor(c("North","South","East","West"))
print(directions)

#c
sort(temp)
```

2. Create a vector in R and perform the following operations:
    a. sort in ascending and descending order
    b. sum and mean
    c. count total number of elements in vector

```
#Q2
temp <- c(1,2,5,3,6,4,2,8,3,1)

#a
print(sort(temp))
print(sort(temp, decreasing = TRUE))

#b
print(sum(temp))
print(mean(temp))

#c
print(length(temp))
```

3. R Program to create a simple calculator

```
add <- function(x,y) {
  x+y
}

subtract <- function(x,y) {
  x-y
}
```

```
multiply <- function(x,y) {
  x*y
}

divide <- function(x,y) {
  x/y
}

print("Select Operation.")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")
choice = as.integer(readline(prompt="Enter choice[1/2/3/4]: "))
num1 = as.integer(readline(prompt="Enter first number: "))
num2 = as.integer(readline(prompt="Enter second number: "))
operator <- switch(choice,"+","-","*","/")
result <- switch(choice, add(num1,num2), subtract(num1,num2), multiply(num1,num2),
divide(num1,num2))
print(paste(num1, operator, num2, "=", result))
```

4. R Program to find the factorial of a number using recursion

```
    factorial <- function(x) {
 if(x==0 || x==1){
  return(1)
 }


 return(x*factorial(x-1))
}


num <- readline(prompt="Enter a number: ")
print(paste("Factorial of",num,"is",factorial(as.integer(num))))
```

5. R Program to add two matrices

```
a <- matrix(1:8, nrow=4)
b <- matrix(1:8, nrow=4)
```

```
c <- a+b

print(c)
```

6. R program to multiply two matrices

   ```
   a <- matrix(1:8, nrow=4)
   b <- matrix(1:8, nrow=4)
   c <- a%%b
   print(c)
   ```

7. Create a data frame of 5 students containing name, roll, department and perform the following
   a. Find the summary
   b. Add division column
   c. Delete a row.

   ```
   students <- data.frame(
     name=c("alice","bob","cathie","donald","eren"),
     roll=c(1,2,3,4,5),
     department=c("IT","CS","EEE","EXTC","IN")
   )

   #summary
   print(summary(students))

   #Add divison column
   students$division <- c("A","B","C","D","E")
   print(students)

   #Delete a row
   # by condition
   students <- subset(students, roll!=2)
   # by index
   students <- students[-c(1),]
   print(students)
   ```

8. Create a data frame of 3 students containing name, roll, department and perform the following
   a. Find the structure
   b. Add two more rows
   c. Delete a column

   ```
   students <- data.frame(
     name=c("alice","bob","cathie","donald","eren"),
     roll=c(1,2,3,4,5),
     department=c("IT","CS","EEE","EXTC","IN")
   ```

```
)

#structure
print(str(students))

#Add two rows
students[nrow(students)+1,] <- c("frank", 6, "MECH")
students[nrow(students)+1,] <- c("george", 7, "CHEM")
print(students)

#Delete a column
students <- subset(students, select=-department)
print(students)
```

9. Create two data frames of students having name, roll, department and perform the following
    a. Natural Join or Inner Join
    b. Left Outer Join
    c. Full join

```
s1 <- data.frame(
  name=c("alice","bob","cathie","donald","eren","x"),
  roll=c(1,2,3,4,5,11),
  department=c("IT","CS","EEE","EXTC","IN","BSC")
)

s2 <- data.frame(
  id=c(2,3,4,5,6,7),
  roll=c(6,7,8,9,10,11),
  college=c("RAIT","IIT","NIT","DTU","NSIT","NIIT")
)

#Natural Join or Inner Join
print(merge(s1,s2,by="roll"))

#Full Join
print(merge(s1,s2,by="roll",all=TRUE))

#Left Outer Join
print(merge(s1,s2,by="roll",all.x=TRUE))

#Right Outer Join
print(merge(s1,s2,by="roll",all.y=TRUE))

#Cross Join
print(merge(s1,s2,by=NULL))
```

10. Create two data frames of students having name, roll, department and perform the following
    a. Inner join
    b. Right outer join
    c. Cross join

11. R Program to print the Fibonacci Sequence Using Recursive Function

```
fibonacci <- function(n) {
 if(n==0)
  return(0)
 else if(n==1)
  return(1)

 return(fibonacci(n-1)+fibonacci(n-2))
}

for(i in 0:10){
 print(fibonacci(i))
}
```

12. R Program to extract first two rows from a given data frame student having name, roll and department.

```
students <- data.frame(
  name=c("alice","bob","cathie","donald","eren"),
  roll=c(1,2,3,4,5),
  department=c("IT","CS","EEE","EXTC","IN")
)

students <- students[c(1,2),]
print(students)
```

13. R program to show scatterplot, box plot and histogram (using cars dataset)

```
#ScatterPlot

plot(x=mtcars$mpg,
    y=mtcars$wt,

    xlab="Miles Per Gallon",

    ylab="Weight",

    main="Mileage vs Weight")


#Boxplot

boxplot(mpg~wt,data=mtcars,xlab="Weight",ylab="Mileage",main="Weight vs Mileage")


#Histogram

weights <- mtcars$wt

hist(weights,main="Weights of Cars",col="yellow",border="blue")
```

14. R program to plot cars dataset using ggplot() and also plot histogram.

```
library(ggplot2)

#first
mtcarsPlot <- ggplot(mtcars, aes(wt,mpg)) + geom_point()
print(mtcarsPlot)

#second
mtcarsPlot <- ggplot(mtcars, aes(x=wt,y=mpg,size=disp,color=disp)) + geom_point()
print(mtcarsPlot)

#Histogram
histogram <- ggplot(data=mtcars, aes(x=mpg)) +
geom_histogram(col="red",fill="green",alpha =.2,binwidth = 5)
print(histogram)
```

15. R program to implement simple linear regression train model and predict results using test model

16. R program to implement decision tree train model and predict results using test model

```
install.packages('party')

library(party)

#create the input dataframe

input.dat<-readingSkills[c(1:105),]
```

```
#create the tree
output.tree<-ctree(nativeSpeaker~age+shoeSize+score,data = input.dat)
#plot the tree
plot(output.tree)
print(input.dat)
#save the file
dev.off()
```