15TH MAY ASSIGNMENET

NAME:SAKSHEE SACHIN SAWANT

COLLEGE:VESIT CHEMBUR

Q2.EXPLAIN HOW TO IMPORT EXCEL FILE AND PERFORM FUNCTION ON THAT

```
> getwd()#setwd("path" )
[1] "C:/Users/SAKSHEE/Documents"
> z=read.csv("Book1.csv",TRUE,",")
> print(z)
  rollno name gemder height   city
1      1  sak      M    5.0    mum
2      2  dfa      M    5.2 nashik
3      3  ddv      F    5.2   pune
4      4   vc      M    6.0    mum
5      5  hng      M    4.0   pune
6      6  ghj      F    6.0    mum
7      7  ety      F    5.0    mum
8      8   ty      M    3.0   pune
> hist(z$height,main="student",ylab="age",xlab="height",)

>
```

Reading a CSV File

Following is a simple example of **read.csv()** function to read a CSV file available in your current working directory −

```
data <- read.csv("input.csv")
print(data)
```

When we execute the above code, it produces the following result −

```
     id,   name,    salary,   start_date,     dept
1      1    Rick     623.30    2012-01-01      IT
2      2    Dan      515.20    2013-09-23      Operations
3      3    Michelle 611.00    2014-11-15      IT
4      4    Ryan     729.00    2014-05-11      HR
5     NA    Gary     843.25    2015-03-27      Finance
6      6    Nina     578.00    2013-05-21      IT
7      7    Simon    632.80    2013-07-30      Operations
8      8    Guru     722.50    2014-06-17      Finance
```

Analyzing the CSV File

By default the **read.csv()** function gives the output as a data frame. This can be easily checked as follows. Also we can check the number of columns and rows.

```
data <- read.csv("input.csv")

print(is.data.frame(data))
print(ncol(data))
print(nrow(data))
```

When we execute the above code, it produces the following result −

```
[1] TRUE
[1] 5
[1] 8
```

Once we read data in a data frame, we can apply all the functions applicable to data frames as explained in subsequent section.

Get the maximum salary

```
# Create a data frame.
data <- read.csv("input.csv")

# Get the max salary from data frame.
sal <- max(data$salary)
print(sal)
```

When we execute the above code, it produces the following result −

```
[1] 843.25
```

Get the details of the person with max salary

We can fetch rows meeting specific filter criteria similar to a SQL where clause.

```
# Create a data frame.
data <- read.csv("input.csv")

# Get the max salary from data frame.
sal <- max(data$salary)
```

```
> a=4
> abs(a)
[1] 4
> exp(2)
[1] 7.389056
> sqrt(a)
[1] 2
> factorial(a)
[1] 24
> log(a)
```

```
[1] 1.386294
> log(a,bas=10)
[1] 0.60206
> pi
[1] 3.141593
> v=c(2,7,8,-9,4,3,10)
> sum(v)
[1] 25
> mean(v)
[1] 3.571429
> median(v)
[1] 4
> max(v)
[1] 10
> min(v)
[1] -9
> mode(v)
[1] "numeric"

>
```

Q2.EXPLAIN FUNCTIONS IN R WITH EXAMPLE.

Function Definition

An R function is created by using the keyword **function**. The basic syntax of an R function definition is as follows −

```
function_name <- function(arg_1, arg_2, ...) {
  Function body
}
```

Function Components

The different parts of a function are −

- **Function Name** − This is the actual name of the function. It is stored in R environment as an object with this name.
- **Arguments** − An argument is a placeholder. When a function is invoked, you pass a value to the argument. Arguments are optional; that is, a function may contain no arguments. Also arguments can have default values.
- **Function Body** − The function body contains a collection of statements that defines what the function does.
- **Return Value** − The return value of a function is the last expression in the function body to be evaluated.

R has many **in-built** functions which can be directly called in the program without defining them first. We can also create and use our own functions referred as **user defined** functions.

```
#how to create fxn and call
> vfun<-function(){
+   print("Welcome you all")
+ }
> vfun()
[1] "Welcome you all"
> #how to create fxn and call
> vfun1<-function(a){
+   print("Welcome you all")
+   print(a)
+ }
> vfun1(7)
[1] "Welcome you all"
[1] 7
> #how to create fxn and call
> vfun2<-function(a,b){
+   print(a+b)
+ }
> vfun2(2,3)
[1] 5
```