Evolution of R

R was initially written by **Ross Ihaka** and **Robert Gentleman** at the Department of Statistics of the University of Auckland in Auckland, New Zealand. R made its first appearance in 1993.

- A large group of individuals has contributed to R by sending code and bug reports.

- Since mid-1997 there has been a core group (the "R Core Team") who can modify the R source code archive.

Features of R

- R is a well-developed, simple and effective programming language which includes conditionals, loops, user defined recursive functions and input and output facilities.

- R has an effective data handling and storage facility,

- R provides a suite of operators for calculations on arrays, lists, vectors and matrices.

- R provides a large, coherent and integrated collection of tools for data analysis.

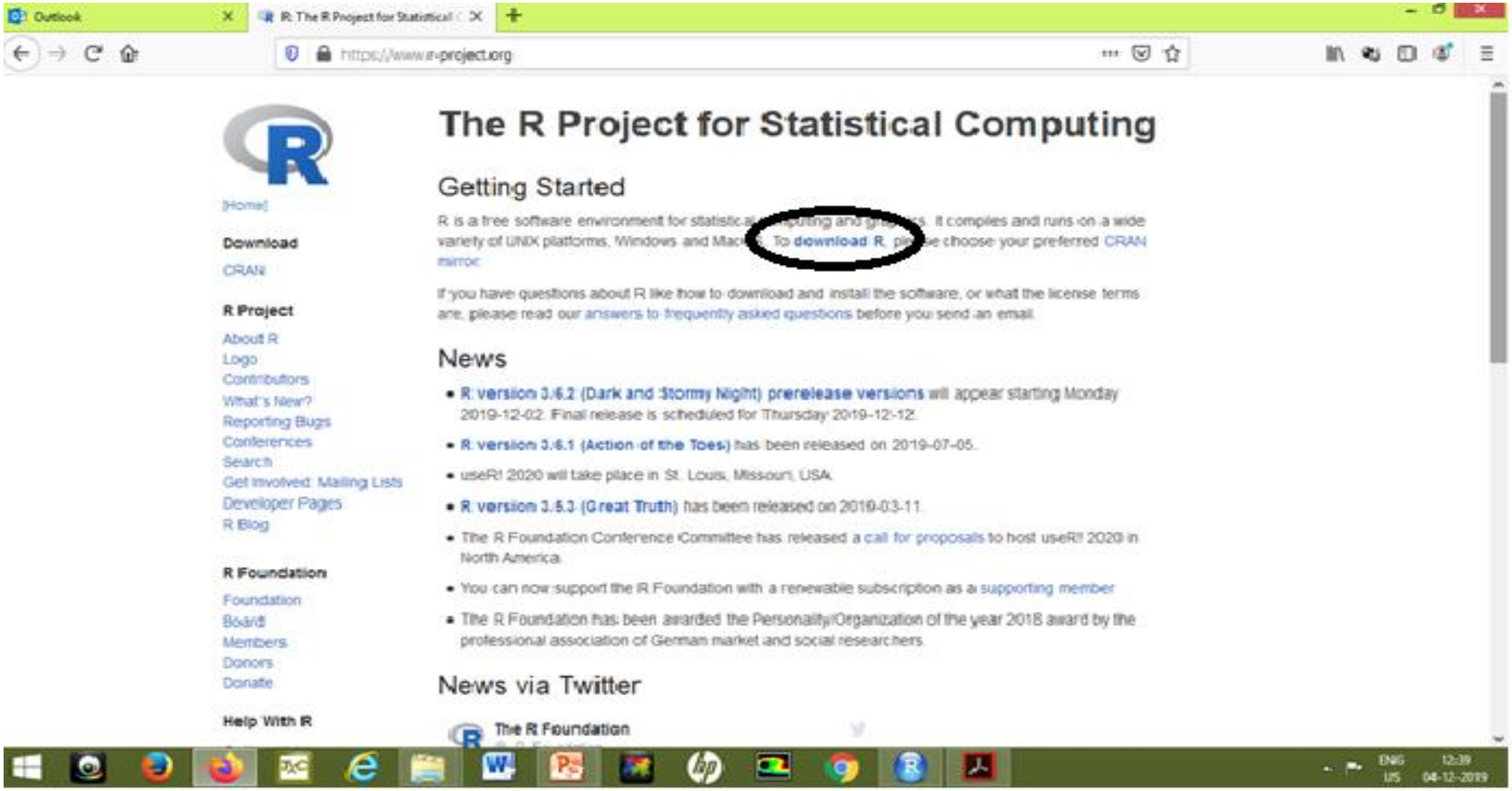
- R provides graphical facilities for data analysis and display either directly at the computer or printing at the papers.

Installing R :-

You may install R in a Windows or Apple Computer by downloading from

https://www.r-project.org

Click on download R



The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To download R, please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

News

- R version 3.6.2 (Dark and Stormy Night) prerelease versions will appear starting Monday 2019-12-02. Final release is scheduled for Thursday 2019-12-12.
- R version 3.6.1 (Action of the Toes) has been released on 2019-07-05.
- useR! 2020 will take place in St. Louis, Missouri, USA.
- R version 3.5.3 (Great Truth) has been released on 2019-03-11.
- The R Foundation Conference Committee has released a call for proposals to host useR! 2020 in North America.
- You can now support the R Foundation with a renewable subscription as a supporting member
- The R Foundation has been awarded the Personality/Organization of the year 2018 award by the professional association of German market and social researchers.

News via Twitter

The R Foundation

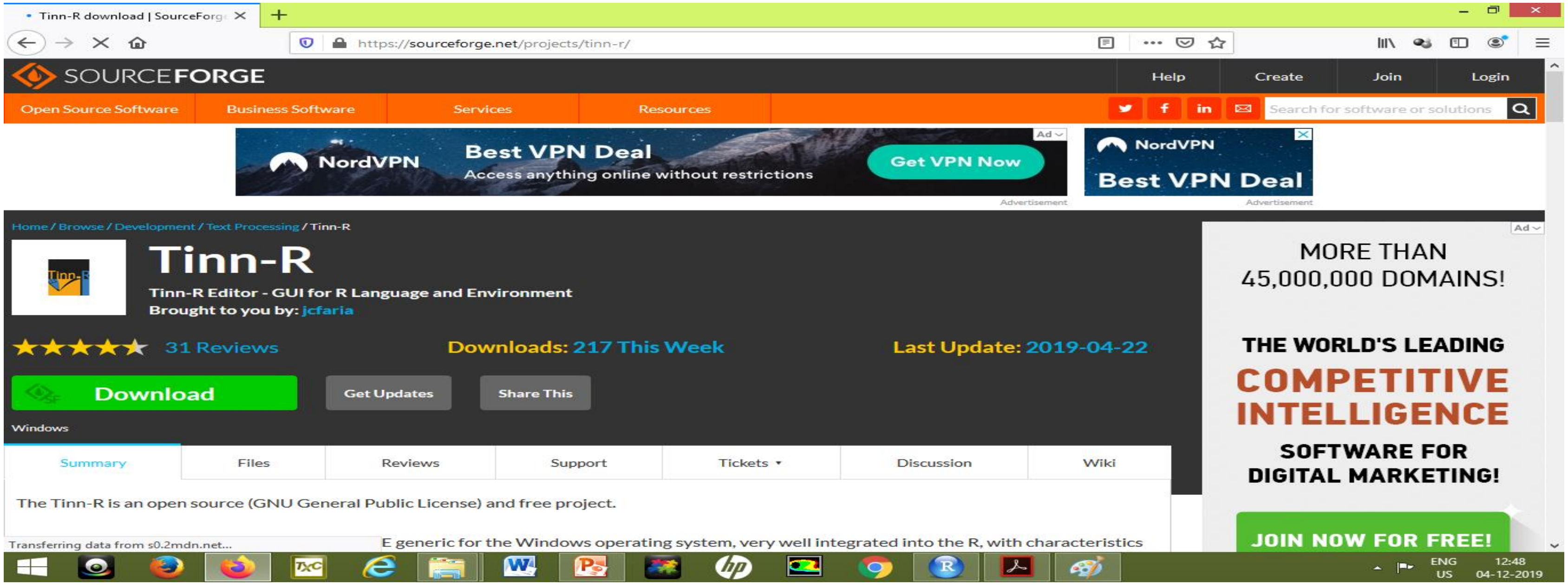
Installing RStudio:-

RStudio is a software which helps in running the R software

Note:-

Several such editors are available , e.g. Tinn R

(https://sourceforge.net/projects/tinn-r)



Tinn-R download | SourceForg
https://sourceforge.net/projects/tinn-r/
SOURCEFORGE
Help
Create
Join
Login
Open Source Software
Business Software
Services
Resources
Search for software or solutions

Home / Browse / Development / Text Processing / Tinn-R

Tinn-R

Tinn-R Editor - GUI for R Language and Environment

Brought to you by: jcfaria

★★★★★ 31 Reviews

Downloads: 217 This Week

Last Update: 2019-04-22

The Tinn-R is an open source (GNU General Public License) and free project.

Transferring data from s0.2mdn.net...

E generic for the Windows operating system, very well integrated into the R, with characteristics

- RStudio is written in c++ programming language.
- RStudio is a free and open-source Integrated Development Environment(IDE) for R.

Download Rstudio software from website

(https://www.rstudio.com/)

Experiments :Introduction; Understanding Data types; Importing/exporting data

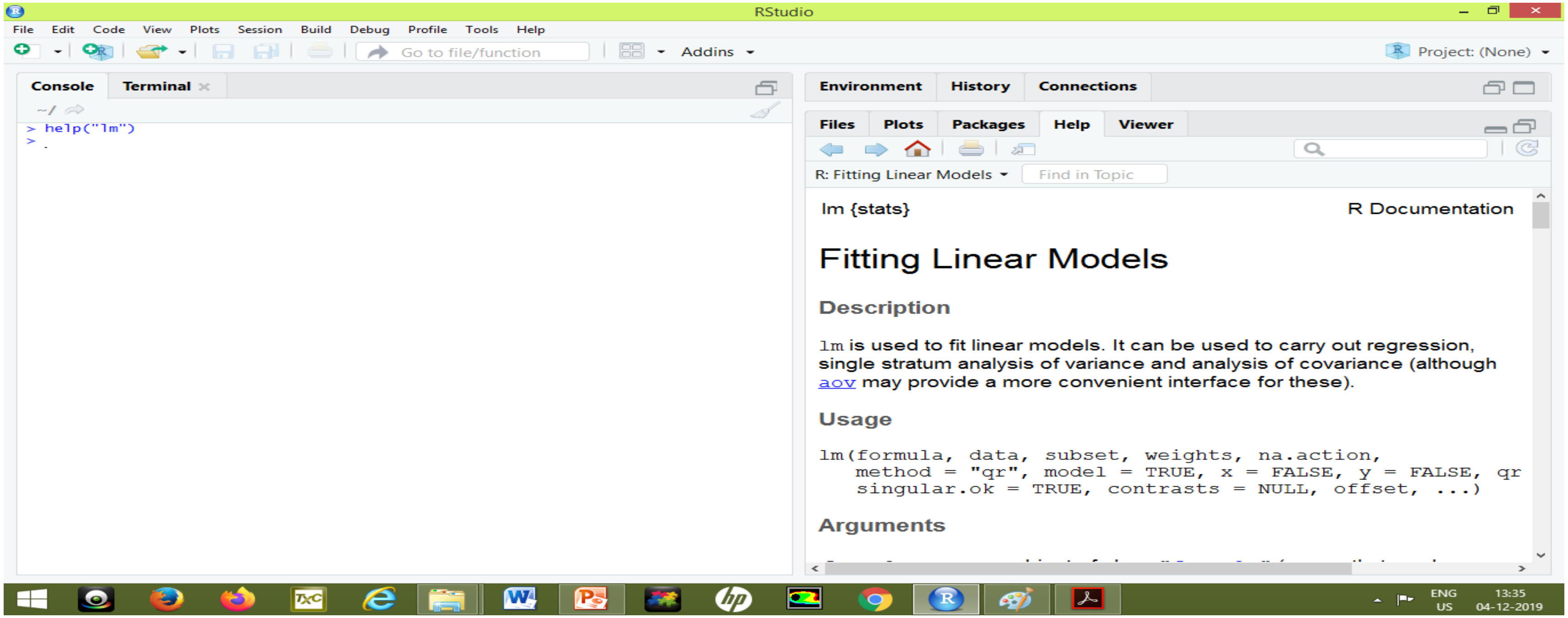
Introduction:- Help, Demonstration, Examples, Packages & Libraries

Help in R

1. help()- for on-line help.

Example:-

> help("lm")


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console
Terminal
~/
Environment History Connections
Files Plots Packages Help Viewer
R: Fitting Linear Models
Find in Topic
ENG
US
13:35
04-12-2019

```
> help("lm")
> 
```

lm {stats} R Documentation

Fitting Linear Models

Description

lm is used to fit linear models. It can be used to carry out regression, single stratum analysis of variance and analysis of covariance (although aov may provide a more convenient interface for these).

Usage

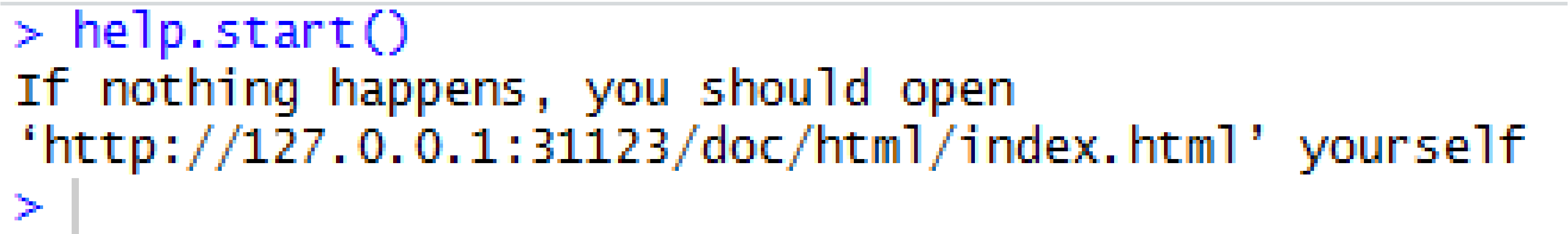
```
lm(formula, data, subset, weights, na.action,
   method = "qr", model = TRUE, x = FALSE, y = FALSE, qr
   singular.ok = TRUE, contrasts = NULL, offset, ...)
```

Arguments

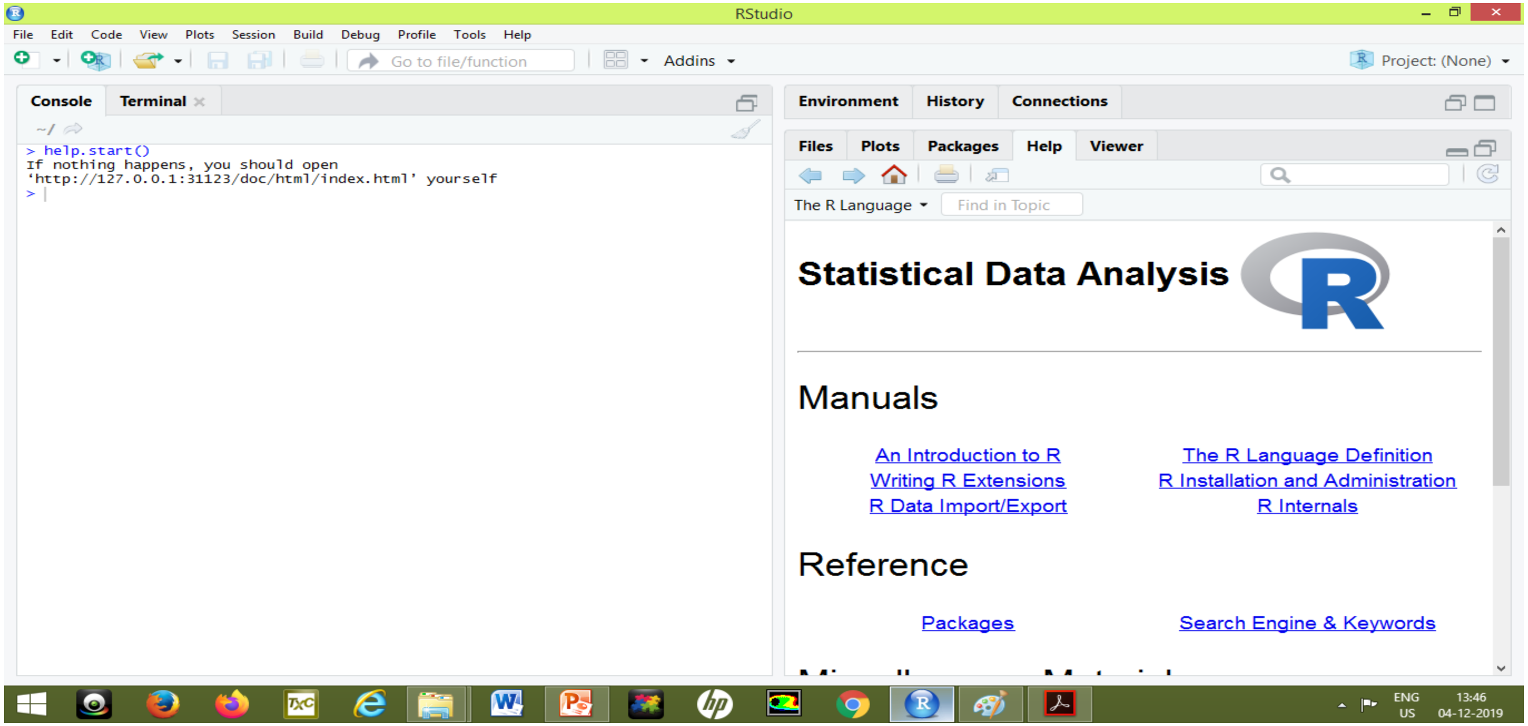
2. help.start() – For an HTML browser interface to help

> help.start()

Example:-



```
> help.start()
If nothing happens, you should open
'http://127.0.0.1:31123/doc/html/index.html' yourself
> 
```



```
> help.start()
If nothing happens, you should open
'http://127.0.0.1:31123/doc/html/index.html' yourself
> 
```

Statistical Data Analysis

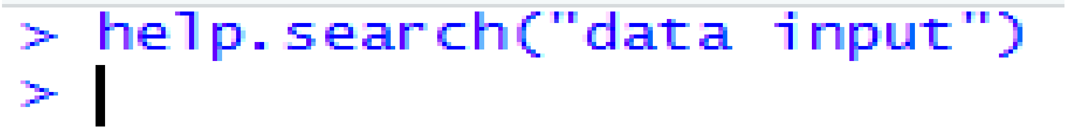
Manuals

An Introduction to R
The R Language Definition
Writing R Extensions
R Installation and Administration
R Data Import/Export
R Internals

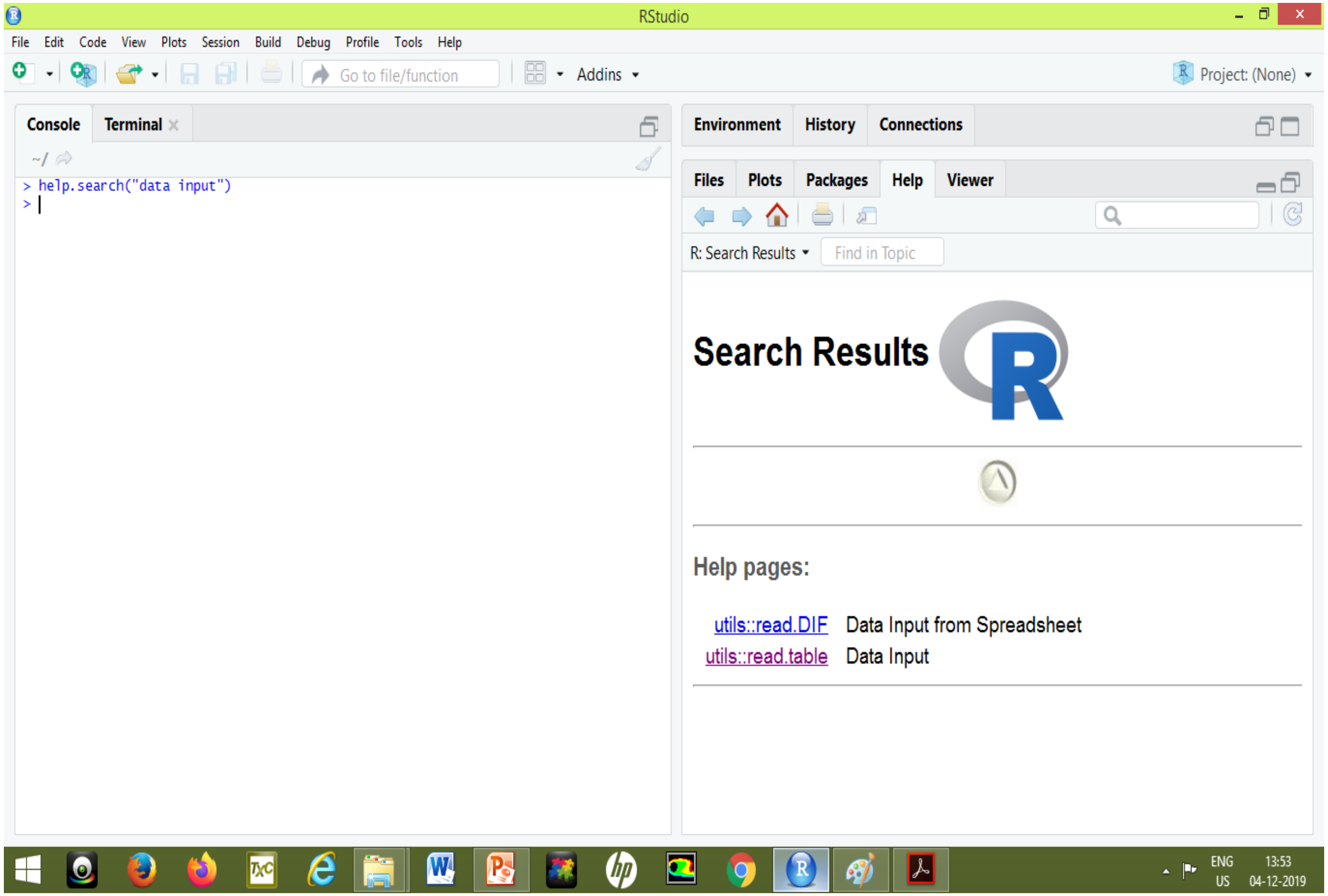
Reference

Packages
Search Engine & Keywords

3. help.search()-To search by the subject on which we want help.



```
> help.search("data input")
> |
```



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console
Terminal
~/
Environment History Connections
Files Plots Packages Help Viewer
R: Search Results
Find in Topic
ENG
US
13:53
04-12-2019

```
> help.search("data input")
> |
```

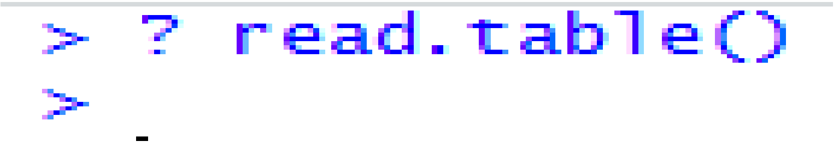
Search Results

Help pages:

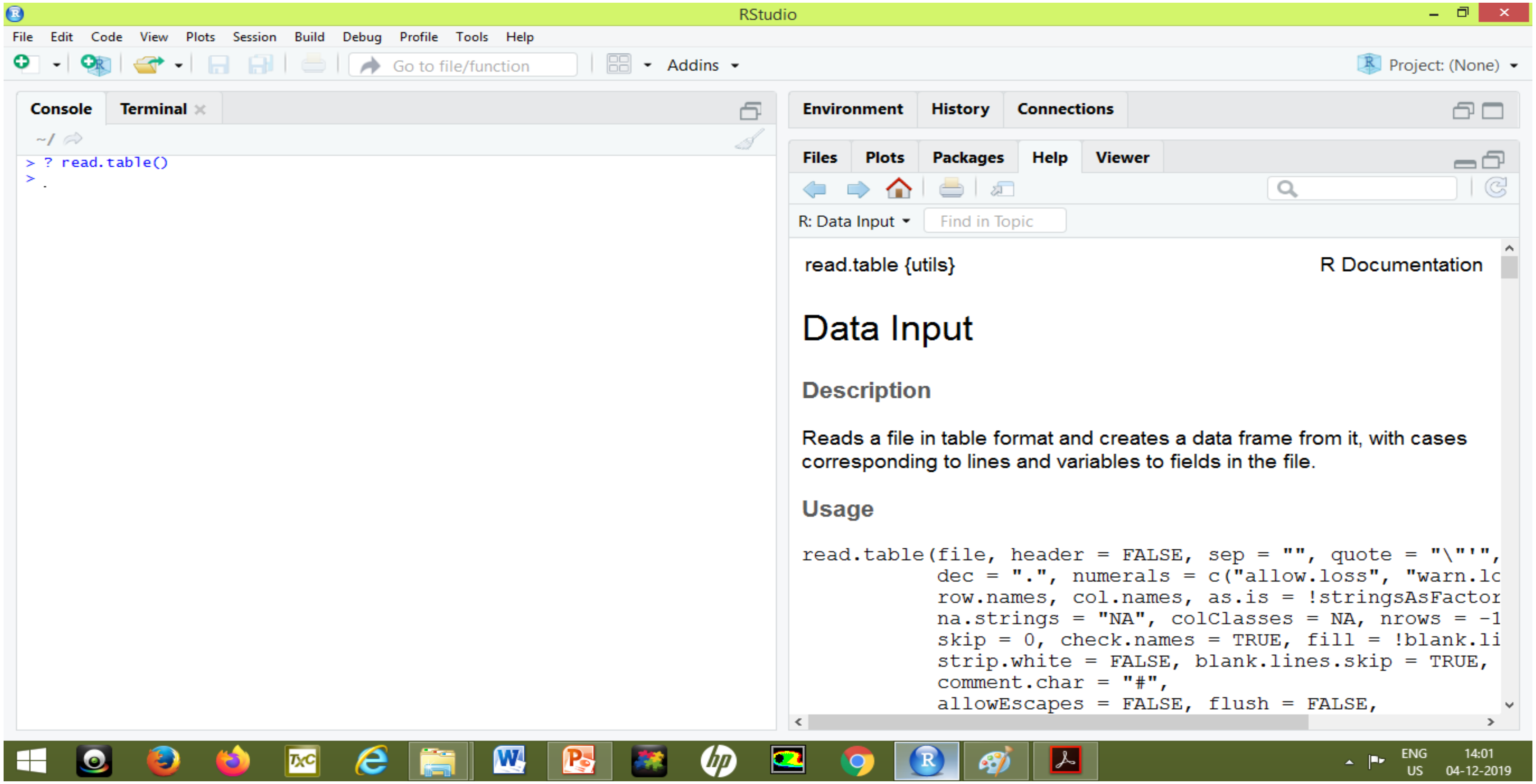
utils::read.DIF   Data Input from Spreadsheet
utils::read.table   Data Input

4. If you need help with a function, then type question mark followed by the name of the function.

Example:-



```
> ? read.table()
>
```



```
> ? read.table()
>
```

read.table {utils}　　　　R Documentation

Data Input

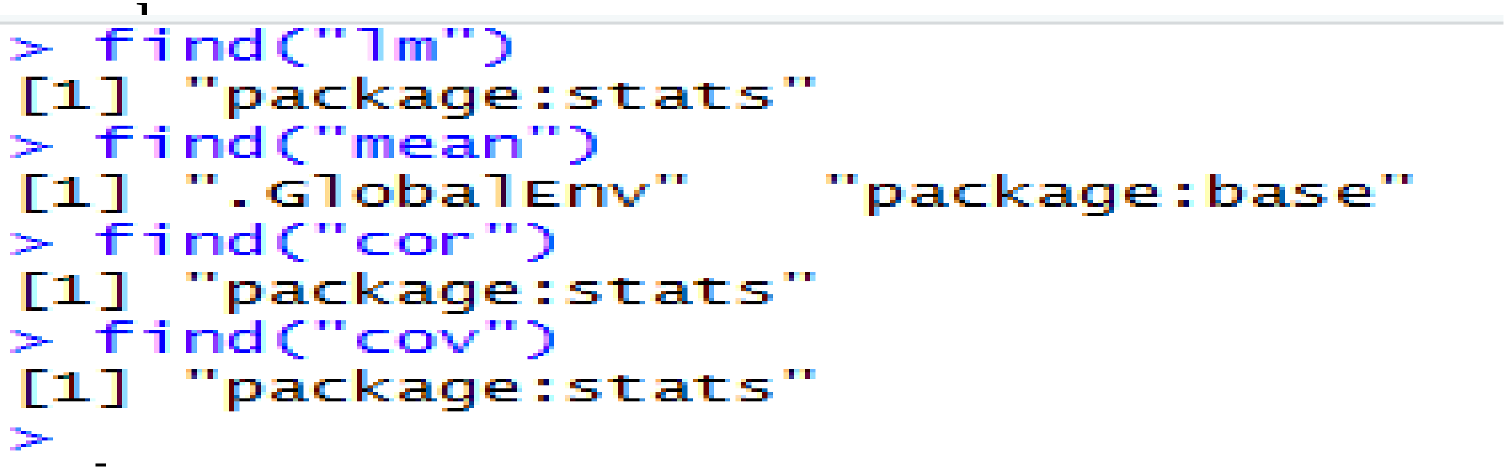
Description

Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file.

Usage

```
read.table(file, header = FALSE, sep = "", quote = "\"'",
           dec = ".", numerals = c("allow.loss", "warn.lc
           row.names, col.names, as.is = !stringsAsFactor
           na.strings = "NA", colClasses = NA, nrows = -1
           skip = 0, check.names = TRUE, fill = !blank.li
           strip.white = FALSE, blank.lines.skip = TRUE,
           comment.char = "#",
           allowEscapes = FALSE, flush = FALSE,
```

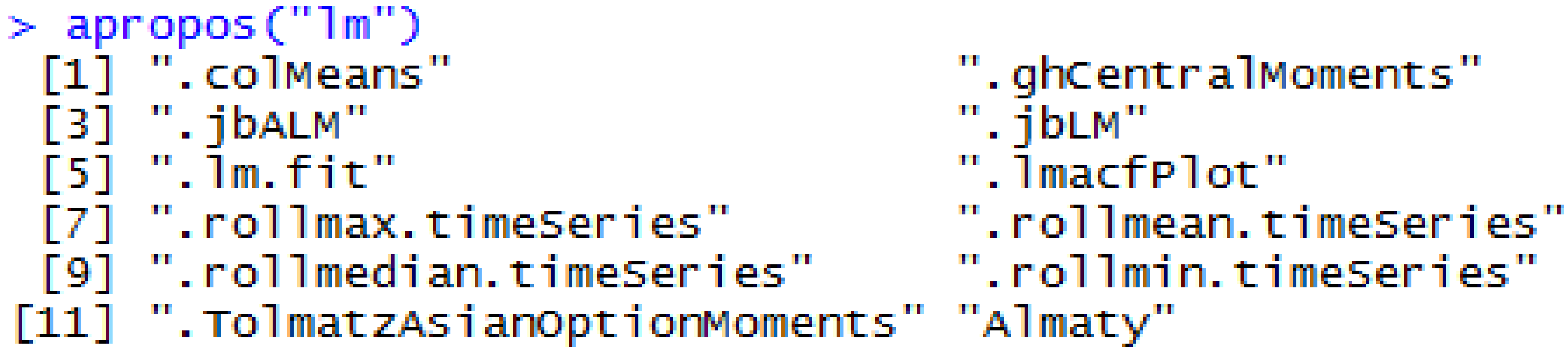
5. find() – Helps us to identify the package that contains the function.



```
> find("lm")
[1] "package:stats"
> find("mean")
[1] ".GlobalEnv"     "package:base"
> find("cor")
[1] "package:stats"
> find("cov")
[1] "package:stats"
>
```

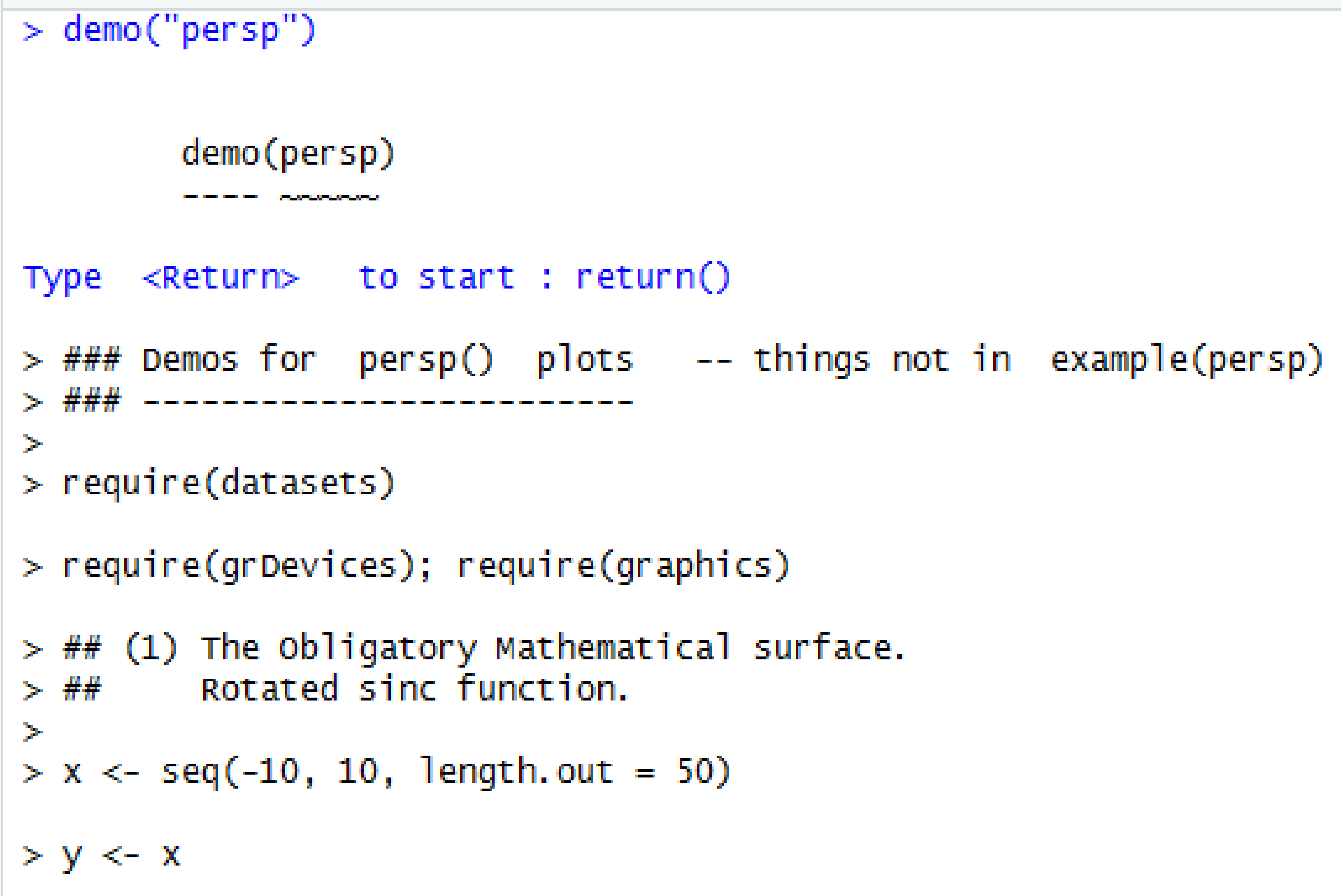
6. apropos() – The apropos returns a character vector giving the names of all objects in the search list that match your enquiry.

E



```
> apropos("lm")
 [1] ".colMeans"                    ".ghCentralMoments"
 [3] ".jbALM"                       ".jbLM"
 [5] ".lm.fit"                      ".lmacfPlot"
 [7] ".rollmax.timeSeries"          ".rollmean.timeSeries"
 [9] ".rollmedian.timeSeries"       ".rollmin.timeSeries"
[11] ".TolmatzAsianOptionMoments"   "Almaty"
```

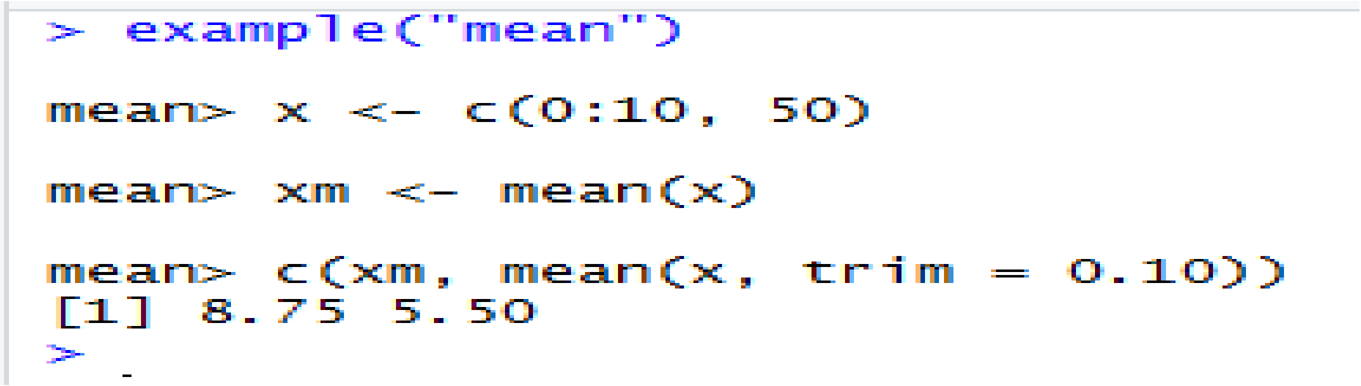
7. demo() – Useful for seeing the type of things R can do.



```
> demo("persp")


        demo(persp)
        ---- ~~~~~

Type  <Return>   to start : return()

> ### Demos for  persp()  plots   -- things not in  example(persp)
> ### -------------------------
>
> require(datasets)

> require(grDevices); require(graphics)

> ## (1) The Obligatory Mathematical surface.
> ##     Rotated sinc function.
>
> x <- seq(-10, 10, length.out = 50)

> y <- x
```

8. example() – To see a worked example of the function.

Example:-



```
> example("mean")

mean> x <- c(0:10, 50)

mean> xm <- mean(x)

mean> c(xm, mean(x, trim = 0.10))
[1] 8.75 5.50
> 
```

Libraries in R

R provided many functions and one also can write his/her own.

Functions and datasets are organised into libraries that come as a part of the base package in R

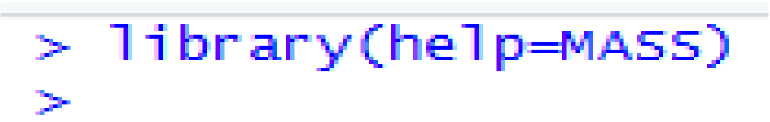
Example:-

To load the 'moments' library type :

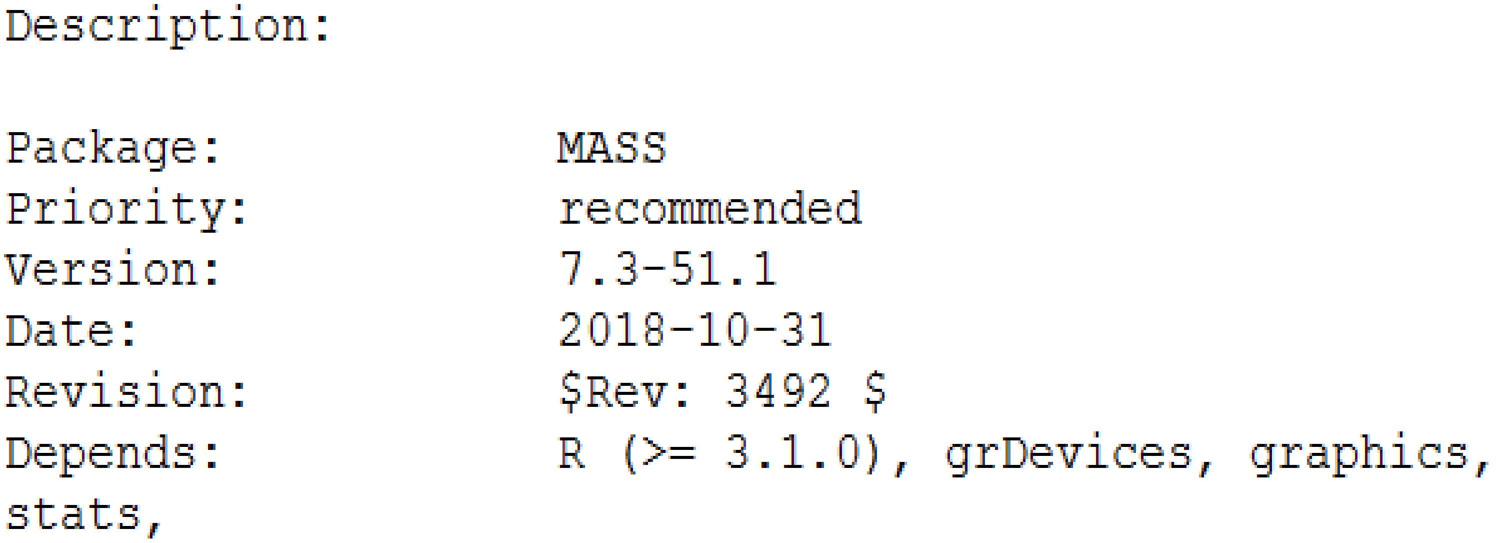
```
> library("moments")
> 
```

Contents of Libraries:-

help function is used discover the content of library packages.



```
> library(help=MASS)
>
```



```
Description:

Package:                    MASS
Priority:                   recommended
Version:                    7.3-51.1
Date:                       2018-10-31
Revision:                   $Rev: 3492 $
Depends:                    R (>= 3.1.0), grDevices, graphics,
stats,
```

followed by the list of all the functions and datasets.

Installing Packages and Libraries:-

Example:-

install.packages(“prob”)

Some libraries that come as a part of base package in R

MASS:-Related to a book Modern Applied Statistics using S-plus.

mgcv:-This library contains the details about the generalized additive model.

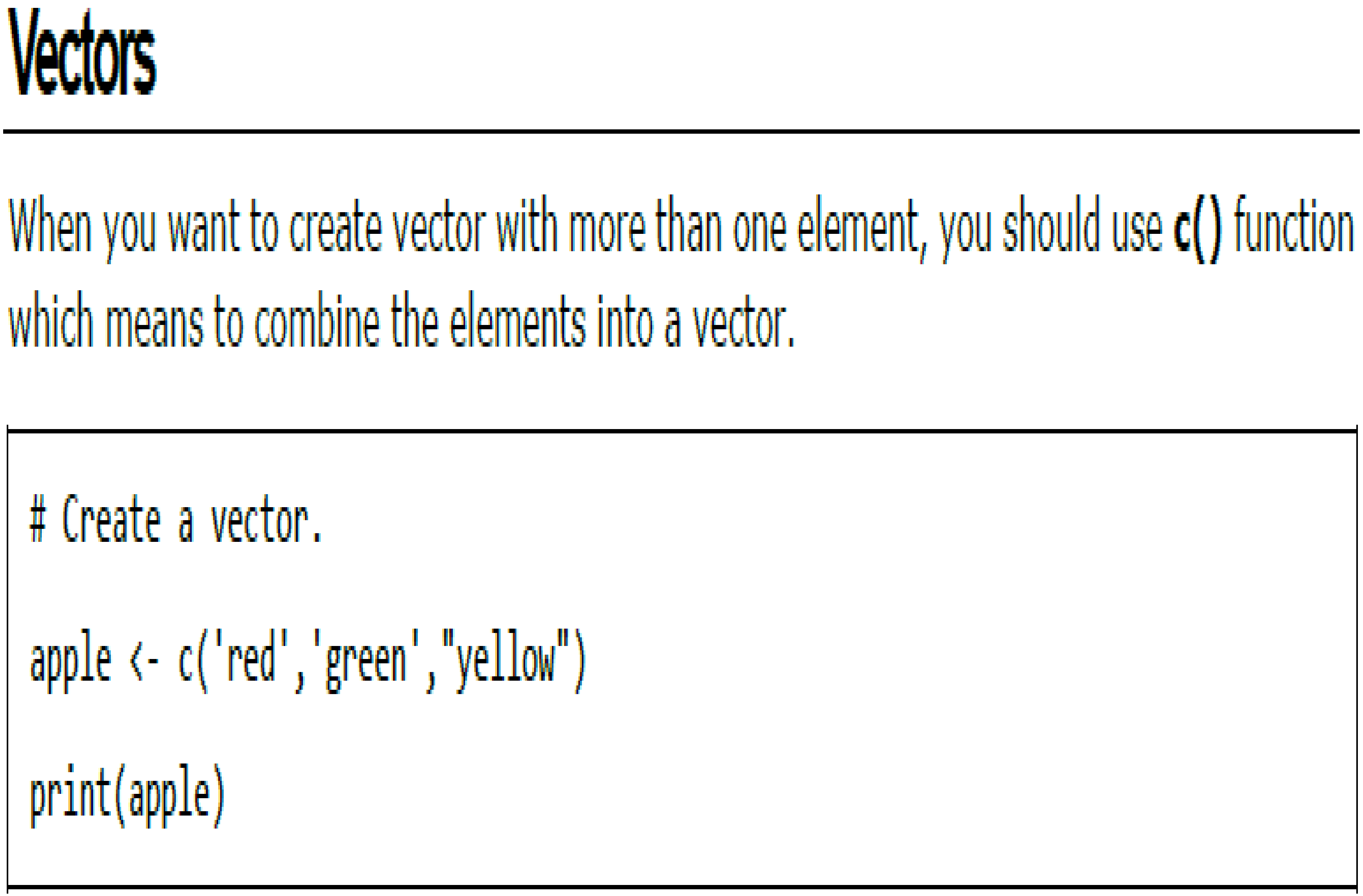
ctrl+l – Clears the RStudio console.

To quit R use q()

R – Data Types

The frequently used ones are:

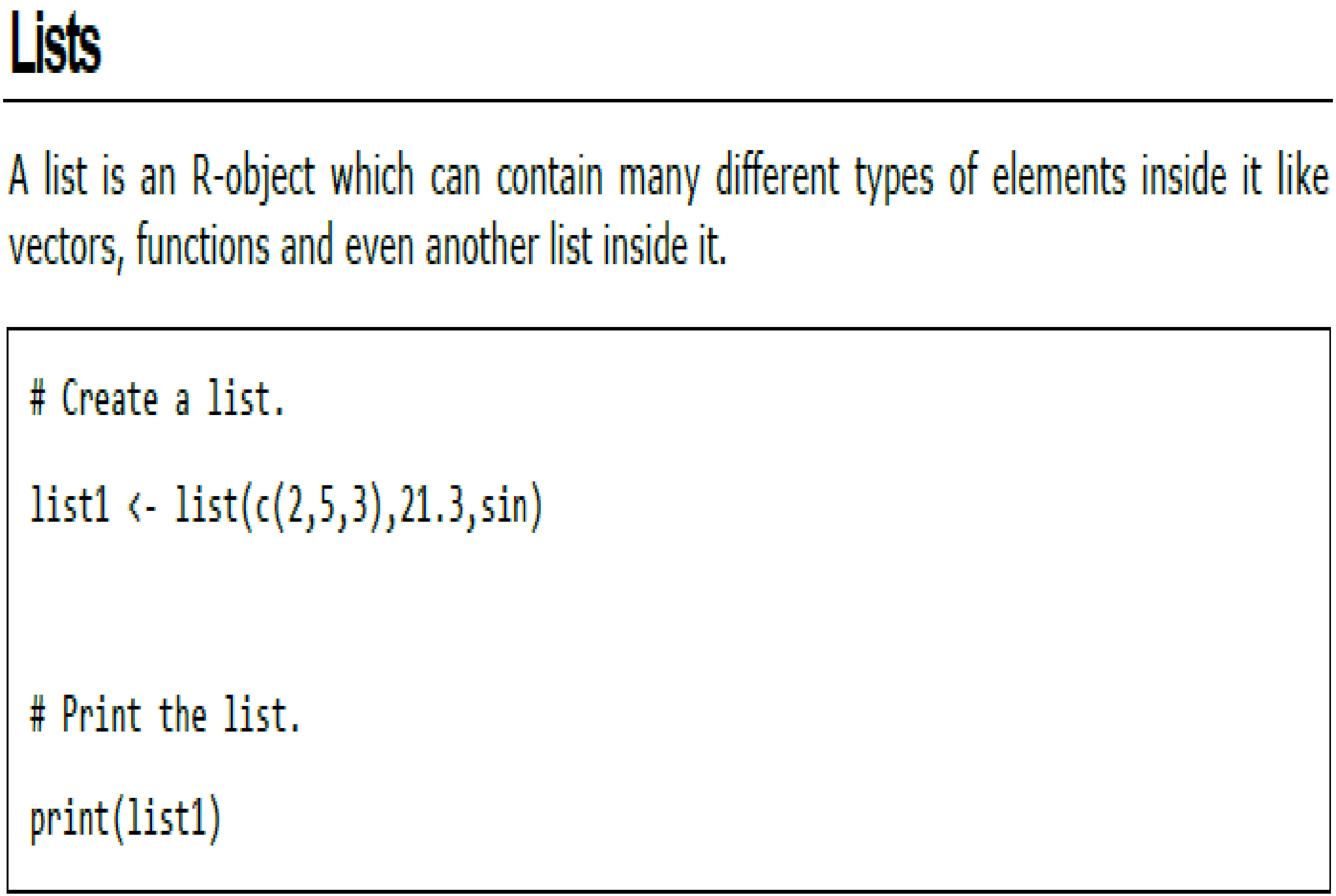
- **Vectors**
- **Lists**
- **Matrices**
- **Arrays**
- **Factors**
- **Data Frames**



Vectors

When you want to create vector with more than one element, you should use **c()** function which means to combine the elements into a vector.

```
# Create a vector.

apple <- c('red','green',"yellow")

print(apple)
```



Lists

A list is an R-object which can contain many different types of elements inside it like vectors, functions and even another list inside it.

```
# Create a list.

list1 <- list(c(2,5,3),21.3,sin)



# Print the list.

print(list1)
```

Matrices

A matrix is a two-dimensional rectangular data set. It can be created using a vector input to the matrix function.

```
# Create a matrix.

M = matrix( c('a','a','b','c','b','a'), nrow=2,ncol=3,byrow = TRUE)

print(M)
```

Arrays

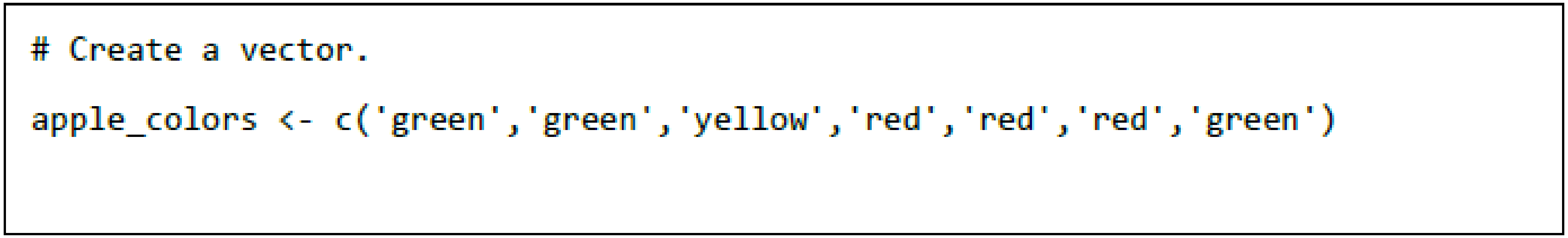
While matrices are confined to two dimensions, arrays can be of any number of dimensions. The array function takes a dim attribute which creates the required number of dimension. In the below example we create an array with two elements which are 3x3 matrices each.

```
# Create an array.

a <- array(c('green','yellow'),dim=c(3,3,2))

print(a)
```

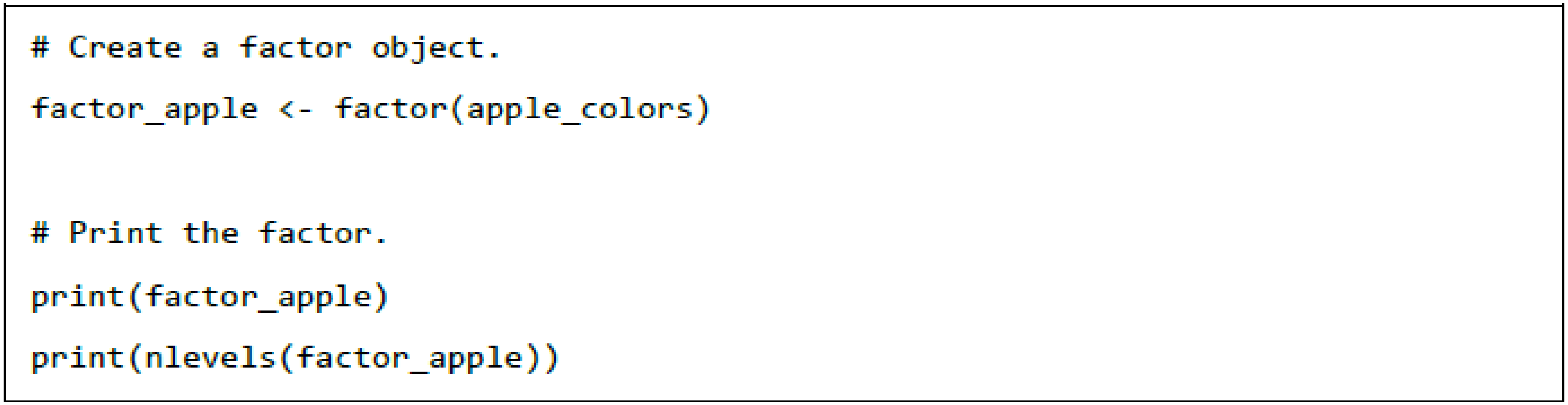
Factors

Factors are the r-objects which are created using a vector. It stores the vector along with the distinct values of the elements in the vector as labels. The labels are always character irrespective of whether it is numeric or character or Boolean etc. in the input vector. They are useful in statistical modeling.

Factors are created using the **factor()** function.The **nlevels** functions gives the count of levels.



```
# Create a vector.

apple_colors <- c('green','green','yellow','red','red','red','green')
```



```
# Create a factor object.

factor_apple <- factor(apple_colors)


# Print the factor.

print(factor_apple)

print(nlevels(factor_apple))
```

Data Frames

Data frames are tabular data objects. Unlike a matrix in data frame each column can contain different modes of data. The first column can be numeric while the second column can be character and third column can be logical. It is a list of vectors of equal length.

Data Frames are created using the **data.frame()** function.

```
# Create the data frame.

BMI <-      data.frame(

                  gender = c("Male", "Male","Female"),

                  height = c(152, 171.5, 165),

                  weight = c(81,93, 78),

                  Age =c(42,38,26)

                  )

print(BMI)
```

After R is started, there is a console awaiting for input. At the prompt (>), you can enter numbers and perform calculations.

```
 > 2+3
 [1] 5
> 100+200+300
[1] 600
```

All text after the pound sign "#" within the same line is considered a comment.

```
> 5                 # type 5 at the prompt

  [1] 5                    # here 5 is returned
  > 3 + 4          # adding two numbers
  [1] 7
  > 5^3                    # will compute 5^3
  [1] 125
  > pi             # pi value
  [1] 3.141593
  > 1 + 2 * 3              # Normal arithmetic rules apply
  [1] 7
```

The expression n1: n2, generates the sequence of integers from n1 to n2.

```
> 1:15    #print the numbers 1 to 15

[1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15

> 5:-5    # print the numbers 5 to -5

 [1]  5  4  3  2  1  0 -1 -2 -3 -4 -5
```

NA is used to indicate that a value is missing or not available. Any arithmetic expression which contains NA will produce NA as a result.

```
    > 1/0        > Inf-Inf              > sqrt(-1)
> 1+sin(NA)

    [1] Inf       [1] NaN          [1] NaN
[1] NA
```

- The following statements store the value 18 under the name x

```
>x = 18

>x <- 18

>18 -> x
```

- Variables can be used in expressions in the same way as numbers.

```
 > x=22
 > x=x+25
 > x
[1] 47
```

Character:-

(A character object is used to represent string values in R)

- as.character () function is used to convert objects into character values:

```
> x=as.character(4.58)
> x
 [1] "4.58"
```

- Strings can be concatenated by using paste function.

```
> paste("First", "Second", "Third")
[1] "First Second Third"

>paste("First", "Second", "Third", sep =
":")
[1] "First:Second:Third"

> fname = "Sri";  lname ="Ram"
> paste(fname)
> paste(fname,lname)
[1] "Sri Ram"
```

Vector Arithmetic:-

- Arithmetic operations of vectors are performed member wise.

```
> a = c(1, 3, 5, 7)
> b = c(1, 2, 4, 8)
```

If we add a and b, the sum would be a vector whose members are the sum of the corresponding members from a and b.

```
> a+b
[1]  2  5  9 15
```

If we multiply a by 5, we get a vector with each of its members multiplied by 5.

```
> 5*a

[1]  5 15 25 35
```

Similarly for subtraction, multiplication and division, we get new vectors via member wise operations.

```
> a-b

[1]  0  1  1 -1

> a*b

[1]  1  6 20 56

> a/b
```

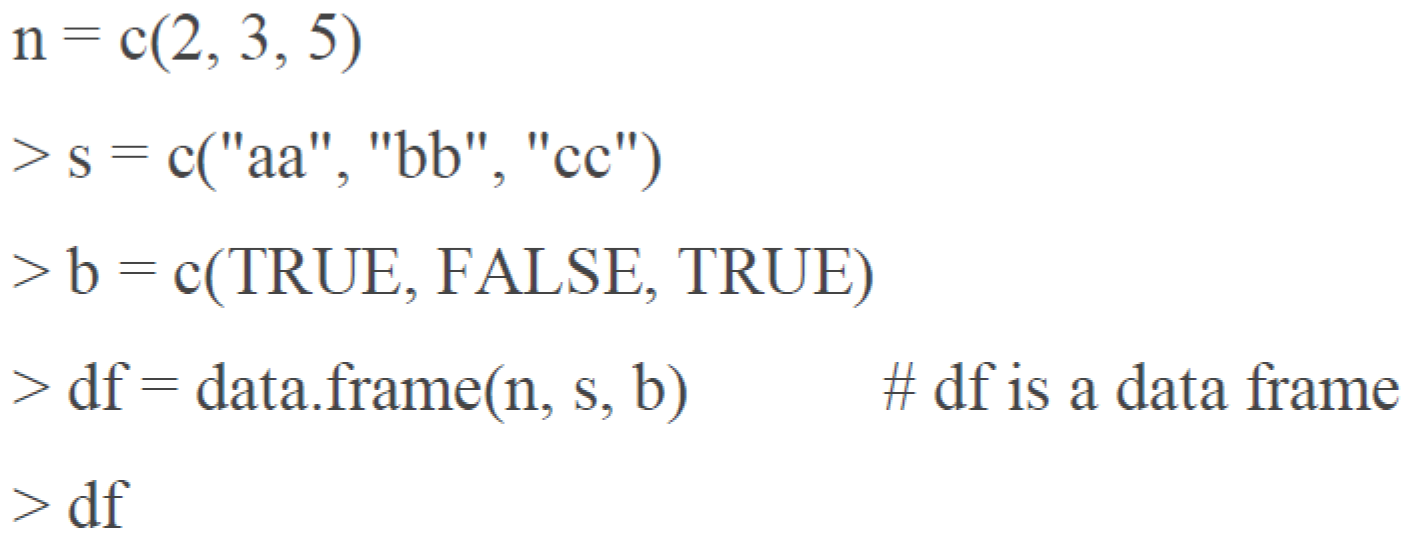
```
[1] 3 5 7 9
```

- If two vectors are of unequal length, the shorter one will be recycled in order to match the longer vector

```
> u=c(10,20,30)
> v=c(1,2,3,4,5,6,7,8,9)
> u+v
[1] 11 22 33 14 25 36 17 28 39
```

Data.frame

Example:



```
n = c(2, 3, 5)
> s = c("aa", "bb", "cc")
> b = c(TRUE, FALSE, TRUE)
> df = data.frame(n, s, b)          # df is a data frame
> df
```

- We use built-in data frames in R for our tutorials. For example, here is a built-in data frame in R, called **mtcars**.

```
> mtcars[1, 2]                  # first row, second column
[1] 6
> mtcars["Mazda
RX4", "cyl"]                # using the row and column names
[1] 6
> nrow(mtcars)                  # number of data rows
[1] 32
> ncol(mtcars)                  # number of columns
[1] 11
```

Important Note:

Enter the following data(or any data) in Excel sheet and save it as CSV file.That is Easy and fast for further analysis

col1	col2	col3
34	23	76
56	54	43
76	34	24
54	76	67
32	24	54

Code:

```
>mydata=read.csv
("C:\\Users\\admin\\Desktop\\mokesh\\workdata.csv")

# select your file based on your path
```