Introduction

What is R?

R is a very powerful programming environment for statistical research and data analysis, including the ability to easily generate numbers, manipulate arrays of various dimensions, and to produce very quality graphics.

Features:-

- An interactive programmed, an effective data handling and storage facility
- An array oriented. Can generate, manipulate, and operate on large array using simple commands.
- It is very graphical. A large number of high level graphics commands are available to produce publication quality graphics both on your screen and on a printer.
- An interpreted language, in which individual language expressions are read and then immediately executed.
- Well developed, simple and effective programming language which includes conditionals, loops, user defined recursive functions and input and output facilities.

General instruction:-

- A name is any combination of letters, numbers, and periods (.), and if it starts with '.' the second character must not be a digit, and can not start with number.
- File name and variables can be more than 8 characters in length.
- It is case sensitive: the object X is not the same as object x.
- Any comment after # on a given line not execute.
- Commands are separated either by semi-colon(';'), or by a new line.
- If a command is not complete at the end line, R will give a different prompt, by default + .
- Data are stored in _data. Subdirectory.

Help Facilities

R has online help system. To start the help system you have many choices:

- For general help:
  1) **>help()**
  2) Click on [Help]

- For a specific command or function:
  1) **>help** (command name), for example, **>help(mean)** or
  2) **>? Function name**, for example, **>? mean**

- For help on characters: the argument must be enclosed in double quotes,
  >help("[[")

- For searching for entries
  The **help.search** command, for example,

>help.search("linear models")

- The examples on a help topic can normally be run by
  > example(topic),

Data objects

Data modes:

In R, data object is a collection of values. The modes of values are as follows:

- Logical: the values T(or TRUE) and F(FALSE).
- Numeric: real numbers, integers, decimal or scientific notation.
- Complex: complex numbers of the form a+bi (3+1.23 i), (a and b) are numeric.
- Character: enclosed by double quotes (“) or apostrophes (‘), such a “Sara” or ‘Sara’.

❖ If you want to know the mode of any object use **mode ( )** function

Types of data objects:

There are seven basic types of data objects in R:

1) Vector (an ordered set of values) – one way array of ordered data.
2) Matrix (two dimensions).
3) Array (a matrix with more than two dimensions)
4) Data frame (generalized matrices that allow a mix of columns with different data modes).
5) List (a list of components, where each component can be a data object of different data types).
6) Factor (categorical data)
7) Time series.

Operators in R

I. Names and Assignment:

The assignment operator (<- or =) used to associate names and values.
For example

x <- 7 or **x =7** # stores the value 7 in an object named x
You can check of the object x either by typing x or **print (x)**.

Note:

All assignments in R remain until removed or overwritten. The **rm()** command used to remove a variable.
Example:

```
>Print(x)
[1] 7
>rm(x)        # remove x
>x
Error: object "x" not found.
```

To display the names of the objects which are currently stored within R,

```
> objects()
```

II. Arithmetic operators :

Operator	Description	Priority
( )	parentheses	1
** or ^	Exponentiation	2
:	Sequences of numbers	3
* /	Multiply, divide	4
+ -	Add, subtract	5

III. Logical and comparison operators:

Operator	Description	Operator	Description
<	Smaller than	&	Factorized And
>	Larger than	|	Factorized Or
= =	Equal to	!	Not
<=	Smaller than or equal to	! =	Not equal to
>=	Larger than or equal to		

Use of Brackets

Name of bracket	bracket	Function
Round brackets	( )	For function calls like in **mean(x)**, and to set priorities
Square brackets	[ ]	Index brackets in **x[3]** used to access or extracts data
Curly brackets	{ }	Block delimiter for grouping sequences of commands as in functions or if statements

Missing values

When an element or value is “not available” or a ”missing value ” the data values are represented by such special symbols NA. when a value (missing data, square root or logarithm of negative number). For these cases, any operation on NA becomes NA.

The function **is.na(x)** gives a logical vector of the same size as x with value TRUE if and only if the corresponding element in x is NA.

```
>x<-c(1:3,NA) ; x
 >is.na(x)     # is TRUE both for NA and NAN values.
[1] FALSE FALSE FALSE  TRUE
>x= =NA
[1] NA NA NA NA
>sum(x)
NA
```

There is a second kind of “missing” values which are produced by numerical computation; it is called Not a Number, NAN, values. Examples are

```
>0/0           # give NAN
>Inf – Inf    # give NAN
>xx=Inf/Inf
> is.nan(xx)     # is TRUE  only for NAN values.
   > x<-c(1,2,3,NAN,4,5,NAN,7)
   > sum(x)
[1] NaN
   > log(-2)
   [1] NaN
   Warning message:
   NaNs produced in: log(x)

   > x<-c(1,2,3,NaN,4,5,NaN,7)
   > is.na(x)
   [1] FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE
   FALSE
```

To remove missing values from x:

```
>x= x[!is.na(x)]
[1] 1 2 3 4 5 7
```