Experiment - 1

Introduction to R Computing

A. Installation of R :-

1. Open any browser and search for R Programming.
2. Go to R Programming site and download R Programming With Compactable Operating Systems.
3. Download Specific executable file.
4. Install R Programming in your system.

Installation of R With RStudio :-

1. Open any browser and Search for RStudio Programming.
2. Go to RStudio programming Website and downloa Specific executable file.
3. Make Sure that file is Compactable to your Systems Operating System.
4. Install the RStudio Software in your Systems.

B. The basics of R syntax and Work Space

R Syntax :-

To Output any text in R we can use Single codes or double codes.

Ex :- "Hello World".

To Output any numbers just tag pack the number where no codes or required.

Simple Calculations:-

5+5

Output:- (1) 10

5-5

Output:- (2) 0

Print function():- We can use print function() (or) might not, as like same as python use print function() {}.

Ex:- for (x in 1:10)
{ print (x)}

Comments:-

Mainly used for documentation of code with symbol hash. Whenever the newline starts with hash they are programming will ignores that line and hence it will be not treated for execution.

Ex:- # Hello R Programming.

Multi line Comments:-

We can use number of comments one after other with starting hash by any newline.

Ex:- # Hello World
R Programming

Variables :—

In R Programming Variables are containers for storing data values. We need not to have any specific command for declaring available. When you asign a value to a variable then you created a container to a sign value make use of <- sign. We can also use = sign, but in some cases it will be for done.

Name, Symbol, John.

ace <- 40.

Data type :— As like in python, need not to declare with any particular type but in or we can change even after they declared.

Ex :- ROM, <- 30, #, Numeric.

ROM, <- "Salt" # character.

Basic Operator :—

* Numeric - (10.5, 45, 78).
* Integer - (1L, 20L, 77L) - L declares as integer in specific data type.
* Complex - (9+3i) - Where i is imaginary part.
* Character - ("Hello") - also called as String in other programming languages.
* Logical - TRUE (or) FALSE - boolean type.

We can use class( ) to check the data type.

Ex:- x <- 10.5
class(x)

Type Conversion:-

As.Numeric()
As. Complex()
As. Integer()

Ex:- x <- 1L
Y <- 2
a <- as numeric (x)
b <- as. Numeric (Y)

Built in Mathfunctions:-

* Min() - Min (5, 10, 15)
* Max() - Max (5, 10, 15)
* Sqrt() - Sqrt (16)
* ABS() - ABS (-4.7)
* Sealling() - It is used to round the number upwards to nearest integer.

Ex:- Sealling (1.4) = 2.

* Floor() - Round the number downwards to nearest Integer.

Ex:- Floor (1.4) = 1

Multi line Strings:-

A <- ("loren is a good person, he is very active").

In R we can add any string at middle by using \n. \ also called as escape character.

If we need to break the line you have to insert as same as. In code, use cat ( ).

String length :— Use n char ( ) used to return the length of string.

Ex:— N char (a).

Check a String :— GREPL ( ) - to check character (or) Sequence of character in string.

Ex:— Str <- "Hello" "SVIT"

Grepl ("s", Str).

To Combine two strings - In R We have special function called paste function.

Ex:— STR <- "Hello".

STR1 <- "Hai".

paste (str, Str1).

output :-

Hello Hai

Escape characters :- In R we use \ followed with characters.

Ex:— STR <- "we are good \"str"\ we are in SVIT".

→ CAT (STR) :—

We use cat ( ) whenever using an escape character.

Code	Description
\\	backslash
\n	newline
\R	Carries Return
\t	tap
\b	backspace

-Arithmetic Operators:-

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
÷	Division
%	Modulo Division
^	Exponential
%%%	Modulo.
%/%	Integer division.

-Assignment Operators-

Operator	Description
<-	-Assignment Operator- used for local declaration of variables.
<<-	-Assignment Operator- we use for declaring

Variables globaling:

Comparison Operators:-

Operator	Description
==	equal
!=	Not equal
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to.

logical Operators:-

Operator	Description
&&	logical-And - Returns true if both the elements are true.
&&	logical And - Returns true when both statements are true.
|	logical OR - If one of the element is true then returns true.
||	logical OR - If returns true any one of the statement Contains true.
!	logical Not - Returns true whenever the statement is false.

Miscellaneous Operator :-

Operator	Description
:	Creates a series of numbe Sequence. Ex:- X<-1:10
%in%	Find out if an element belongs to a vector. Ex:- X %in% Y
%*%	Used for matrix multiplication Ex:- X %*% Y.

loops in R

If else loop :-

Ex:-
```
If (condition)
{
Print(x)
}
else
{
Print(x)
}
```

Else if loop :-

```
if (condition)
{
Print(x)
}
else if (condition)
{
Print()
}
else
{
Print()
}
```

While loop :-

```
While(Condition)
{
Statements
}
```

While if loop :-

```
While (condition)
{
statements
if (condition)
{
break
}}
```

For loop :-

```
For (x in 1:10) {
Print(x) }
```

Functions :- Is a block of code only runs whenever it is called. Function returns Data As a result.

Ex:-

```
my fun <- function ( ) {
          Print ("Hello")
          }.
```

Parameter :- Variable listed inside the parenthesis in function defination.

Argument :- The value that is sent to to the function when it is called.

Return Function( ) :- It is used to return value within a function.

Ex:-

```
My function <- function (x)
{
return ( 5 * x)
}
```

Print (My function (4))

Nested Functions():-

Ex:- Nested function <- function (x, y)
{
A <- x+y
return (A)
}
Nested function (Nested function (2,2), Nested function (3,3))

Recursion :-

Ex:- Try Recursion <- function (K)
{
If (K>0)
{
result <- K + try recursion K-1
Print (result)
}
else
{
result = 0
return (result)
}}

**Vectors :- A List of items that are of the same type; denoted with C ( ).

Ex:- Fruits <- C ("Apple", "Banana", "Orange").

Vector length:- Used to Calculate length of the vector using length function.

Ex:- Fruits length(Fruits)

Sort( ):- It is used to sort a vector.

Ex:- Sort (Fruit)

To access Vectors in ~~or programming~~ :-

In R-Programming to access the vectors we use[].

Ex:- Fruit[2].

Change item :- To change item of a vector we use vector index number has to be assigned with same time.

Ex:- Fruit[1] <- "Mango.

Repeat Vector :- In R we have a special function called REP{}

Ex:- repeat <- Rep{ c(1,2,3), each=3).

Sequence function :- In R We use sequence function to get starting and ending values by time of interval (or) Sequence.

Ex:- Number <- Seq (From = 0, 2=100, By = 20).

Work Space

part 1 Work space :- It is alloted for writing Source code to get a source Space in RStodio :

* Click on File Option We will get a few options like new file, new project, recent files, Open project, Save, Save as,

Compile report, print and close options.

* After clicking on new file you will get some options like or Script (ctrl+4, shift+), or notebook, shiny web app, text file, C++ file, R-HTML R-presentation & R-documentation options.

* Click on R-script, it will opens a new window. It consists of few options like Save, Source on save, Search button, code tools, Run button and Re-run source code of previous one.

* To save the file simply click on save button and redirect to a folder to where source files stored.

Part 2 Workspace:—

* It is alloted for giving/viewing source code variables, functions and objects. It consists few options like environment, Window, History Window and connections Window.

Environment Window:—

* It is basically used to view files, saved files and to import data sets.

* In this Window we have a special feature called global environment.

* In global environment, its maintanance values, functions and objects. Everything is detailed

according to Source code execution.

History Window :-

* It maintance all the code that you have return. It is mainly used for checking previous code that have been run.

Connections Windows :-

In Connection Window we have two more options. like ODBC and Spark. ODBC is used to store the data and we can retrieve it back. Spark is also used to store large amount of data as seen in big data (hadoop).

Part 3 Workspace :-

In this Workspace we have two options like Console Window and terminal Window. We mostly use Console Window for code execution of R.

Part 4 Workspace :-

In this Workspace we have few options like files, plots, packages, help and Viewer

File :- When you click on File option it will opens a window with fewer options like new folder, delete, rename and more.

Plot :- plots are graphical representation of data. When we use plots related functions in the Source code after execution the plots are visible in any graph manner.

Packages :- When you click on packages option a Window will be open with options like install and update. In this Window we have a system library with library name, description and its

Version. We can install required packages at any time before using in source code. The same can also updated.

Help:- Help is a function which is used to no about usage of functions, libraries and Objects. In this window we have R. resources, R Studio, manuals and references.

*In this window we have a search bar, simply type any function or about any library. you will get all the details about specific search.

Viewer:-

It is used to view necessary graphs, charts.

In this window we have few options like delete, clear and show in new window.

Each of the workspace can be managed one by one with operations provided.

6. List and Matrices:-

List:- list is a collection of data which is ordered and changable. To create a list use list function.

Ex:- list()

l1 ← list("Apple", "Banana").

To Access list:- In order to access the list items refer to its index number.

Ex:- l1[1].

Change Item Value :- To change any item refer to the index number.

Ex:- $l_1[1] \leftarrow$ "Grape"

List length function :- To find how many times items in list used.

Ex:- length [l_1].

To check if item exist :- To find specific item in list, we use % in % Operator and it will be return 'True' or 'False'.

Ex:- "apple" % in % l_1

Add items in list :- In R to append any item in to the list use append function( ).

Ex:- append (l_1, "Banana").

(or)

append (l_1, "Banana", after = 1)

Remove list items :- To remove list items refer for negative value of index number.

Ex:- $l_2 \leftarrow l_1[-1]$

Range of indexes in list :- We can also specified a range of indexes using : Operator.

Ex:- $l_1[2:5]$

Loop through a list :-

```
Ex:- for (x in l1)
     {
     (Print(l1)
     }
```

Join to a list :- To Command (or) join two lists, We need to use C().

Ex:- list 1 = l1 ("A", "B", "C")

list 2 = l2 ("M", "N", "D")

list 3 = c (list 1, list 2)

Matrices :- Matrices is a two-dimensional data set with Columns and rows. A Column is a verlical representation of data, A row is a horizontal representation of data. In R a matrix can be created by using matrix() with specifying n rows and n Columns.

m1 ← matrix (c(1,2,3,4,5,6), n row = 3, n col = 2).

Matrix either may be a string.

Ex:- m2 ← matrix (c("a","b","c","d" n row = 2, ncol = 2)

Access Matrix :-

We can access items by using [ ] by specifying row and Column position.

m1 [1,1]

To access whole row just give the row number and leave a comma after.

Ex:- m1 [1,1]

m1 [1,]

To access whole Column gives the Column number and leave a Comma before.

We use Square brackets to select the Column.

Ex:- $m_1[\ , 1]$.

Access more than One row :-

Ex:- $M_1[c(1,2),]$

Access more than One Column :-

Ex:- $M_1[\ , c(1,2)]$

Adds Rows and Columns a matrix :- We use c bined ( ) to add columns in matrix.

$M_3 \leftarrow$ Cbind (M_1, c("A", "B")

Use or bind to add row in matrix :-

$M_3 \leftarrow$ c (bind (M_1, R("A", "B")

Remove rows and columns :- To remove rows and Columns use c( ) with specifying rows and columns with index numbers.

Ex:- $M_1 \leftarrow m[-c(1), -c(2)]$

Each of row and columns are specified with - to remove from matrix.

Check if an item exist :- We use percentage in %.

Operator to check an item exist in matrix or.

Ex:- "Apple" %in% M_1.

To find number of rows and columns :- In R we have a special function called dim( ). It represents no. of rows and Columns.

Ex:- dim(m_1).

Matrix length :- To find length of matrix we use length function.

Ex:- length (m_1).

loop-through matrix:-

In R we can pass matrix to a loop.

Ex:-
```
For(rows in 1:nrow(M1))
{
For (Columns in 1:ncolumn(M1))
{
print (M1 [rows, Columns])
}
}
```

Combine two matrix:- We use R bind function for rows cbind function for columns.

Ex:- M1 <- Matrix(c("a","b","c","d"), Nrow = 2, Ncol = 2)

M2 <- M2(c("e","f","g","h"), Nrow = 2, Ncol = 2)

M3 <- R bind (M1, M2)

M3 <- C bind (M1, M2)

Experiment:- D

Sub Settings:-

R Provides usage of Work space Window as required to user. We can change any Work space Window to Comportable positions. By placing cursor on third window or 4th window you will get a moove icon which is meant to perform operations like drag and lower the size of Window.

Ex:- X <- C(4, 8, 12, 14, 9, 7)

choose what we keep

Y <- X[c(2,3)]

choose what we draw use negative sign to the index number.

Y <- X[c(-1,-2)].

Drop the last value of vector.

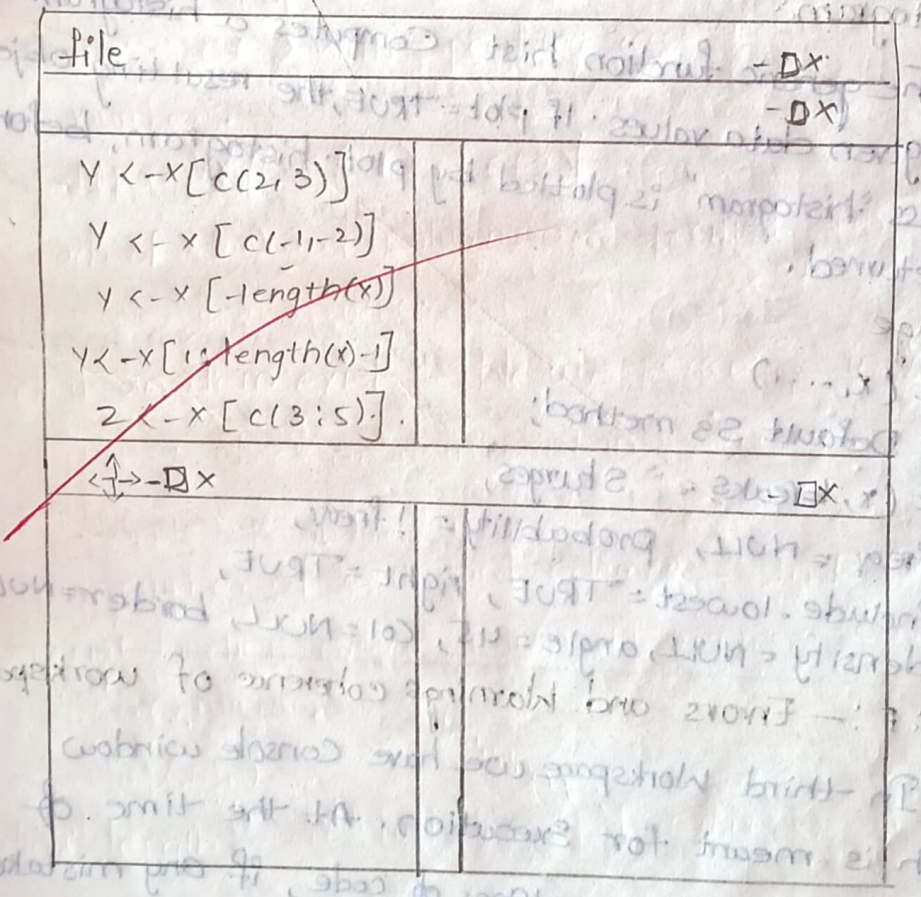
Y <- X[-length(X)]

Drop the last value, and key remaining values.

Y <- X[1:length(X)-1]

use of : operator

Z <- X[c(3:5)]



Experiment-E:-

System defined functions, help System.

We use many system defined functions for various Operations. If we don't know how to use the System defined function. Go to Console type as:

-Help start().

It will navigate to a 4th Workspace Containing help System provided with a Search bar. Go to that Search bar and type.

hist and hit Enter button.

It will returns either details about histogram function.

R: Histograms ▾ (Find in TOPIC)

-Histogram:

The generic function hist Computes a histogram of the given data values. If plot=TRUE, the resulting object of class "histogram" is plotted by plot.histogram, before it is returned.

Usage

```
hist(x, ...)
## Default S3 method:
hist(x, breaks = "Sturges",
  freq = NULL, probability = !freq,
  include.lowest = TRUE, right = TRUE,
  density = NULL, angle = 45, col = NULL, border = NULL,
```

Exp: F:- Errors and Warnings coherence of workspace.

In third Workspace we have Console window which is meant for Execution. At the time of Execution or any source lines of code, if any mistakes as underground or System automatically provides you Errors and Warnings with description.

Console | Terminal x

Warning: You are configured to use the CRAN mirror at https://cran.rstudio.com/. This mirror supports secure (HTTPS) downloads however your system is unable to communicate securely with the server (possibly due to out of date certificates files on your system). Falling back to using insecure URL for this mirror.

To learn more and/or disable this warning message see the "Use secure download method for HTTP" option in Tools -> Global options -> Packages.

- Error: unexpected symbol in "plot (1:10, main= "graph" x label" ~~x-axis", x label=~~

> plot (1:10, main = "graph" x label = "x-axis", Y ~~label~~ = "y-axis")

> plot (1:10, main = "graph" x lab = "x-axis", Y lab = "y-axis").

31/5/22