R

The Programming Language

➔ R is a programming language and software environment for statistical analysis, graphics representation and reporting. R was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, and is currently developed by the R Development Core Team.

➔ R is freely available under the GNU General Public License, and pre-compiled binary versions are provided for various operating systems like Linux, Windows and Mac.

- The core of R is an interpreted computer language which allows branching and looping as well as modular programming using functions. R allows integration with the procedures written in the C, C++, .Net, Python or FORTRAN languages for efficiency.

Basic Syntax

To launch R prompt:

```
$ R
```

Print Hello World:

```
> print(“Hello World”)
```

R Script File

➔ Usually, you will do your programming by writing your programs in script files and then you execute those scripts at your command prompt with the help of R interpreter called **Rscript**. So let's start with writing following code in a text file called test.

```
# My first program in R Programming
myString <- "Hello, World!"

print ( myString)
```

➔ Save the above code in a file test.R and execute it at Linux command prompt as given below.

```
$ Rscript test.R

#Output
[1] "Hello, World!"
```

Data Types

- ➔ In contrast to other programming languages like C and java in R, the variables are not declared as some data type. The variables are assigned with R-Objects and the data type of the R-object becomes the datatype of the variable. There are many types of R-objects. 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)

# Output
[1] "red"    "green"  "yellow"
```

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.

```r
# Create a list.
list1 <- list(c(2,5,3),21.3,sin)
```

```
[[1]]
[1] 2 5 3


[[2]]
[1] 21.3


[[3]]
function (x)  .Primitive("sin")
```

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)

# Output

     [,1] [,2] [,3]
[1,] "a"  "a"  "b"
[2,] "c"  "b"  "a"
```

Arrays

- While Matrices are 2 dimensional, arrays can have any number of dimension. It takes dim as an attribute which creates the required number of dimension.

```
# Create an array.
a <- array(c('green','yellow'),dim = c(3,3,2))
print(a)
```

```
, , 1

     [,1]     [,2]     [,3]
[1,] "green"  "yellow" "green"
[2,] "yellow" "green"  "yellow"
[3,] "green"  "yellow" "green"

, , 2

     [,1]     [,2]     [,3]
[1,] "yellow" "green"  "yellow"
[2,] "green"  "yellow" "green"
[3,] "yellow" "green"  "yellow"
```

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.

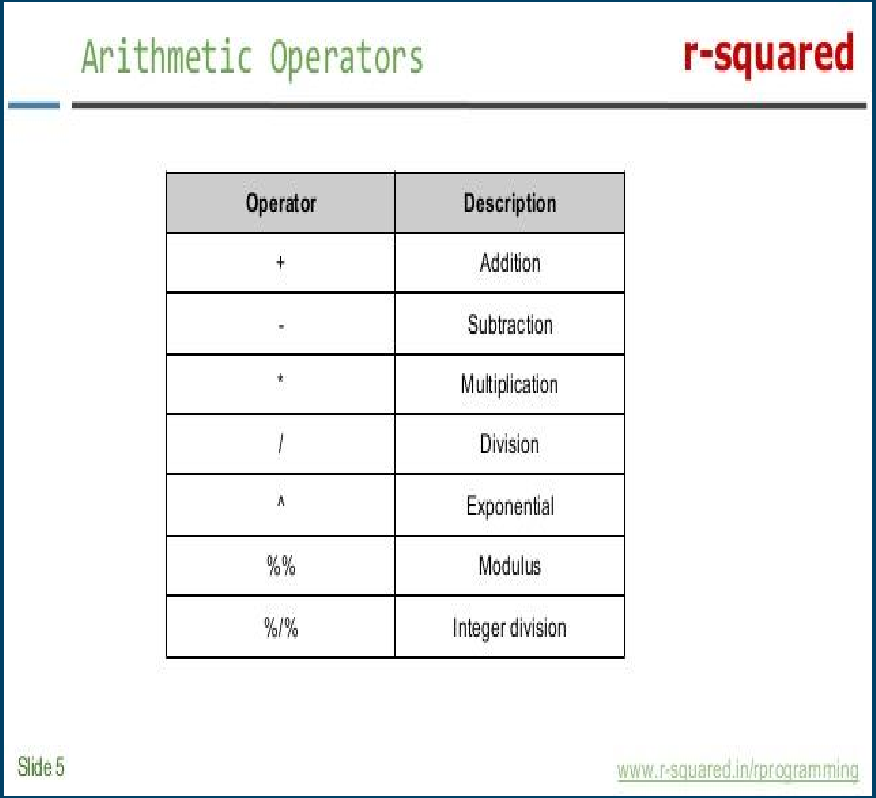
```
# Create the data frame.
BMI <-  data.frame(
  gender = c("Male", "Male","Female"),
  height = c(152, 171.5, 165),
  weight = c(81,93, 78),
  Age = c(42,38,26)
)
print(BMI)
```

```
  gender height weight Age
1   Male  152.0     81  42
2   Male  171.5     93  38
3 Female  165.0     78  26
```

Types of Operators

- Types of operators are:
  - Arithmetic Operators
  - Relational Operators
  - Logical Operators
  - Assignment Operators

Arithmetic Operators



Arithmetic Operators

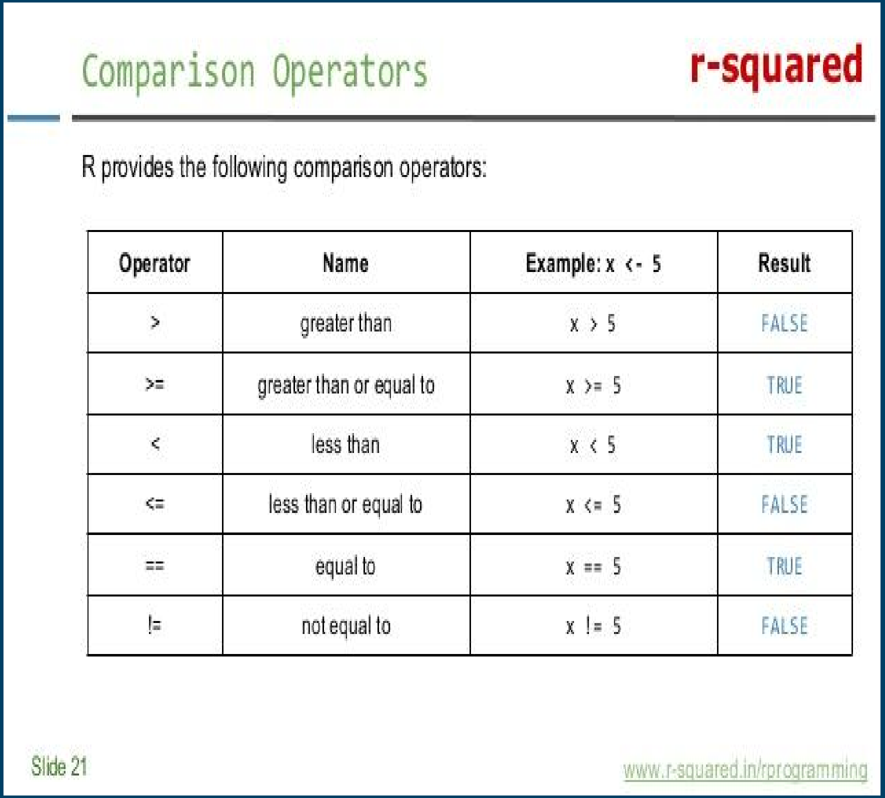
r-squared

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponential
%%	Modulus
%/%	Integer division

Slide 5

www.r-squared.in/rprogramming

Comparison Operators



Comparison Operators

r-squared

R provides the following comparison operators:

Operator	Name	Example: x <- 5	Result
>	greater than	x > 5	FALSE
>=	greater than or equal to	x >= 5	TRUE
<	less than	x < 5	TRUE
<=	less than or equal to	x <= 5	FALSE
==	equal to	x == 5	TRUE
!=	not equal to	x != 5	FALSE

Logical Operators

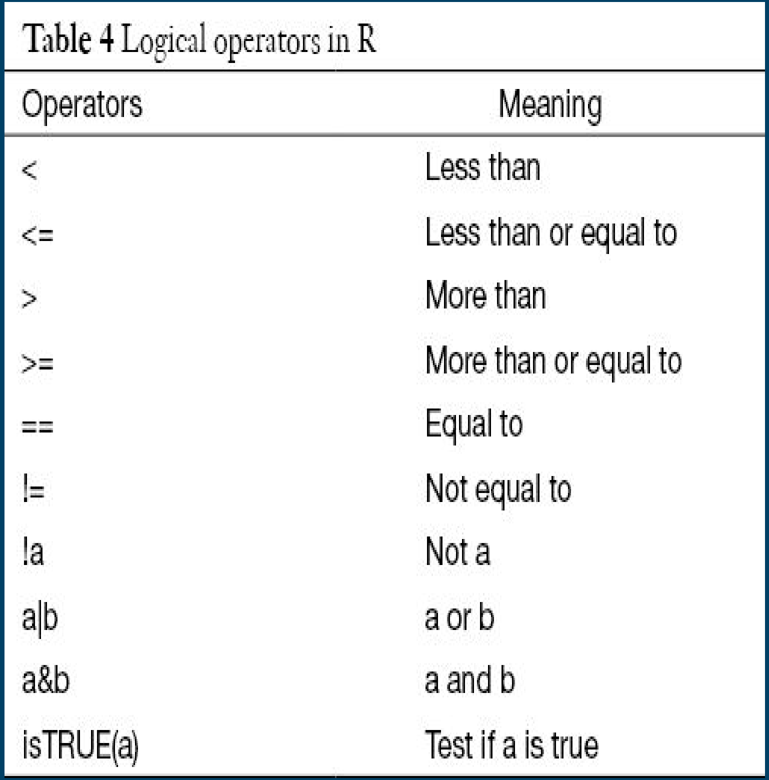
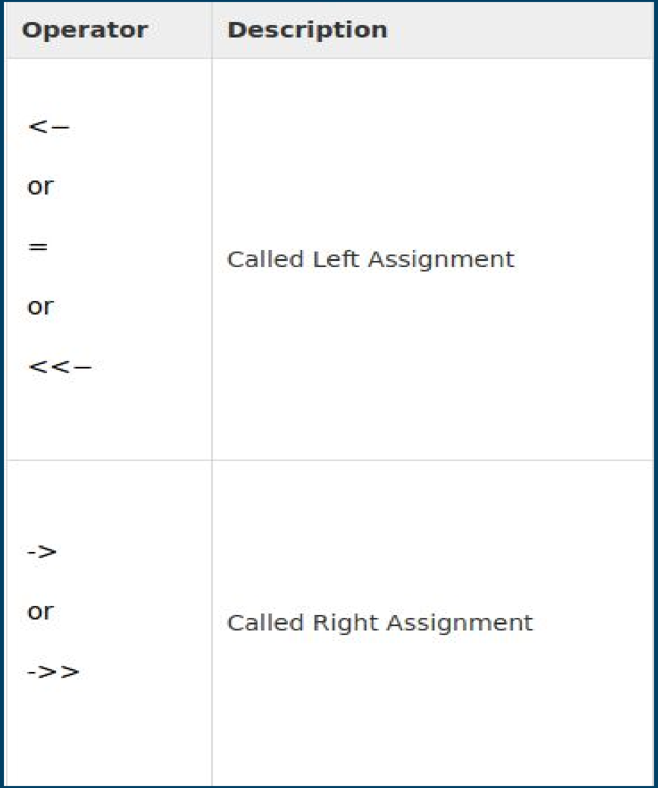


Table 4 Logical operators in R

Operators	Meaning
<	Less than
<=	Less than or equal to
>	More than
>=	More than or equal to
==	Equal to
!=	Not equal to
!a	Not a
a|b	a or b
a&b	a and b
isTRUE(a)	Test if a is true

Assignment Operators



Operator	Description
<− or = or <<−	Called Left Assignment
-> or ->>	Called Right Assignment

Decision Making

→ if...else Statement

```
if(boolean_expression) {
   // statement(s) will execute if the boolean expression is true.
} else {
   // statement(s) will execute if the boolean expression is
false.
}
```

→ Switch statement

```
switch(expression, case1, case2, case3....)
```

```
x <- switch(
   3,
   "first",
   "second",
   "third",
   "fourth"
)
print(x)
```

Loops

- ➔ for loop

```
for (value in vector) {
  statements
}
```

- ➔ while loop

```
while (test_expression) {
  statement
}
```

- ➔ repeat loop
  - ◆ The **Repeat loop** executes the same code again and again until a stop condition is met.

```
repeat {
  commands
  if(condition) {
    break
  }
}
```

Functions

➔ 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
}
```

➔ A function is called using the function name along with the arguments given in parentheses.

```
function_name(arg_1, arg_2, …)
```

Functions

➔ Calling a function with argument values by position:

```
# Call the function by position of arguments.
new.function(5,3,11)
```

➔ Calling a function with argument values by name:

```
# Call the function by names of the arguments.
new.function(a = 11, b = 5, c = 3)
```

R Charts and Graphs

- Pie Chart
  - In R the pie chart is created using the **pie()** function which takes positive numbers as a vector input. The additional parameters are used to control labels, color, title etc.
- The basic syntax for creating a pie-chart using the R is –

```
pie(x, labels, main, col)
```

x is a vector containing the numeric values use in the pie chart.

labels is used to give description to the slices.

main indicates title of the chart.

col indicates color palette.

R Charts and Graphs

- Bar Charts
  - R uses the function **barplot()** to create bar charts. R can draw both vertical and horizontal bars in the bar chart. In bar chart each of the bars can be given different colors.
- The basic syntax to create a bar-chart in R is –

```
barplot(H, xlab, ylab, main, names.arg, col)
```

H is a vector containing numeric values in bar chart.

xlab and **ylab** are labels for x axis and y axis respectively.

names.arg is a vector of names appearing under each bar.

R Charts and Graphs

- Scatterplots
  - Scatterplots show many points plotted in the Cartesian plane. Each point represents the values of two variables. One variable is chosen in the horizontal axis and another in the vertical axis.
- The basic syntax for creating scatterplot in R is –

```
plot(x, y, main, xlab, ylab, xlim, ylim)
```

x and **y** are the data set whose values are the horizontal and vertical coordinates.

main is the title of the graph.

xlim and **ylim** are values of x and y for plotting.

R Charts and Graphs

- Line Graphs
  - A line chart is a graph that connects a series of points by drawing line segments between them. These points are ordered in one of their coordinate (usually the x-coordinate) value.
- The basic syntax to create a line chart in R is –

```
plot(v,type,col,xlab,ylab)
```

v is a vector containing the numeric values.

type takes the value "p" to draw only the points, "l" to draw only the lines and "o" to draw both points and lines.

xlab and **ylab** are labels for x and y axis respectively.

main is the title of the chart

Thank You