**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 programming language and software environment for statistical analysis, graphics representation and reporting. The following are the important 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.

> myString <- "Hello, World!" > print ( myString)

Comments

# My first program in R Programming

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 data type of the variable. There are many types of R-objects. The frequently used ones are:

• Vectors

• Lists

• Matrices

• Arrays

• Factors

• Data Frames

The simplest of these objects is the vector object and there are six data types of these atomic vectors, also termed as six classes of vectors. The other R-Objects are built upon the atomic vectors.

Logical ----TRUE , FALSE

Numeric ------12,3.76,10000

Integer ---------4L,5L,89L

Complex-----------2+3i

Character--------- ‘a’,”hello”,”6.89”,”False”

Raw----------"Hello" is stored as 48 65 6c 6c 6f

v <- TRUE

print(class(v))

v <- charToRaw("Hello")

print(class(v))

**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)

print(class(apple))

**((-4:0))**

**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 dimensions. 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 modelling.

Factors are created using the factor() function. The levels functions give 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(162, 171.5, 165),

weight = c(84,73, 78),

Age =c(45,39,36) )

print(BMI)

Finding Variables To know all the variables currently available in the workspace we use the ls() function. Also the ls() function can use patterns to match the variable names.

print(ls())

**The ls() function can use patterns to match the variable names.**

print(ls(pattern="var"))

print(ls(all.name=TRUE))

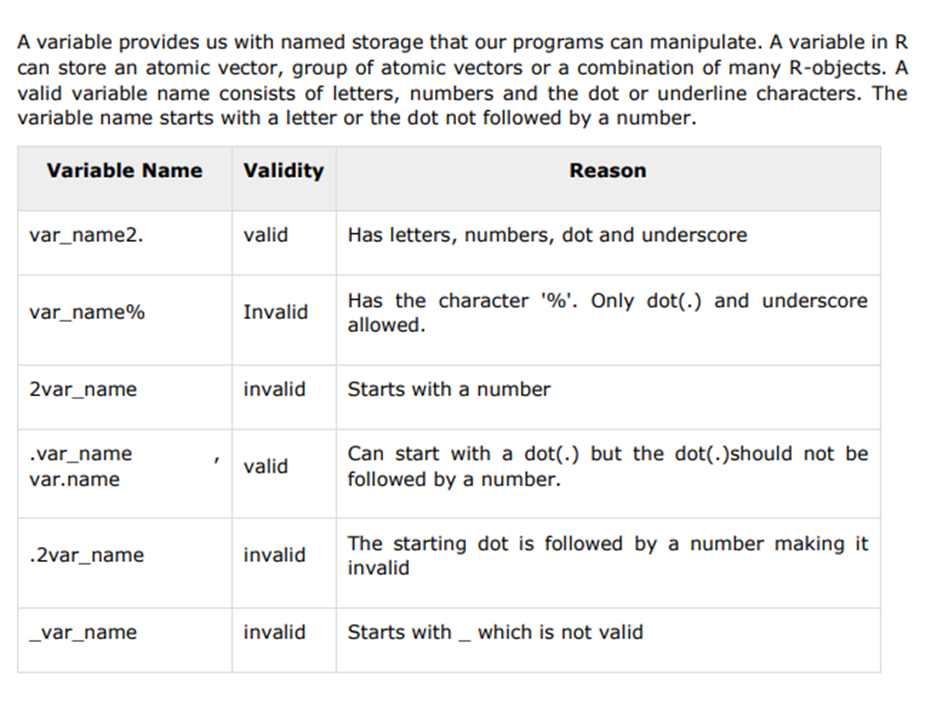
**Deleting Variables**

Variables can be deleted by using the rm() function.

All the variables can be deleted by using the rm() and ls() function together.

rm(list=ls()) print(ls())

R variables

****

**Variable Assignment**

The variables can be assigned values using leftward, rightward, and equal to operator. The values of the variables can be printed using print() or cat()function. The cat() function combines multiple items into a continuous print output.

#Assignment using equal operator.

var.1 = c(1,2,3)

# Assignment using leftward operator.

var.2 <- c(‘learn’,"R")

# Assignment using rightward operator.

c(TRUE,12) -> var.3

print(var.1)

cat ("var.1 is ", var.1 ,"\n")

cat ("var.2 is ", var.2 ,"\n")

cat ("var.3 is ", var.3 ,"\n")

**Finding Variables**

To know all the variables currently available in the workspace we use the ls() function. Also the ls() function can use patterns to match the variable names.

print(ls())

The variables starting with dot(.) are hidden, they can be listed using "all.names=TRUE" argument to ls() function.

print(ls(all.name=TRUE))

**Deleting Variables**

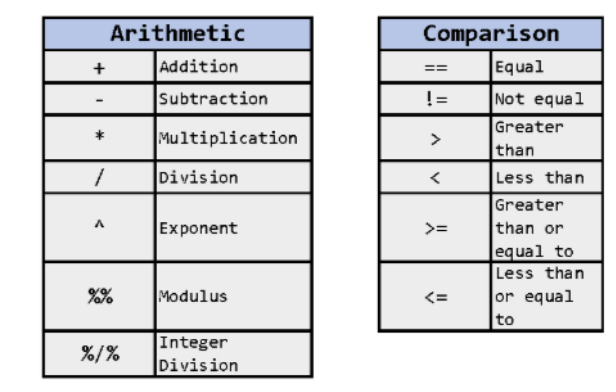
Variables can be deleted by using the rm() function. Below we delete the variable var.3. On printing the value of the variable error is thrown.

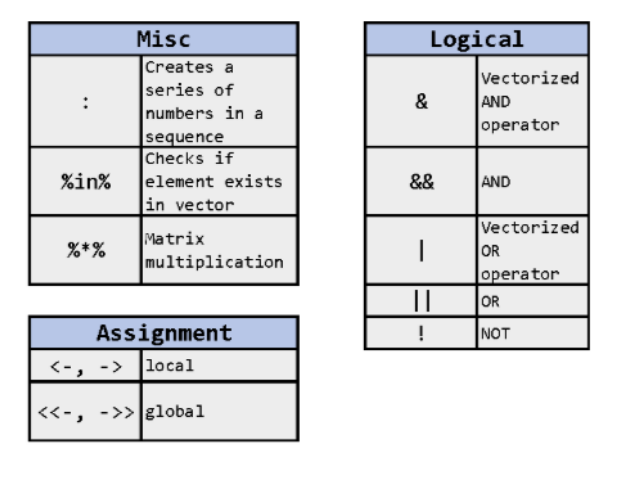
**R – Operators**

An operator is a symbol that allows you to perform an action or define some sort of logic. The following image demonstrates the operators that are available to you in R.

**Types of Operators**

We have the following types of operators in R programming: • Arithmetic Operators • Relational Operators • Logical Operators • Assignment Operators • Miscellaneous Operators

****

****