R Overview

R Overview

- 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.
- 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.
- R is freely available under the GNU.
- R is free software distributed under a GNU-style copy left, and an official part of the GNU project called **GNU S**.

Features of R

- R is a well-developed, simple and effective programming language .
- 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.
- As a conclusion, R is world’s most widely used statistics programming language.

Installation

- https://cran.r-project.org/bin/windows/base/
- https://rstudio.com/products/rstudio/download/#download

R Objects and Attributes

- R has five basic or atomic classes of objects:
  - character
  - numeric(real number)
  - integer
  - complex
  - logical(True/False)

- The most basic object is a vector
  - A vector can only contain an object of the same class.
  - Empty vector can be created with vector() function.

Numbers

- Numbers in R are generally treated as numeric objects, so all numbers are treated as double number precision real numbers.
- If you explicitly need an integer, you have to mention L
- Example, if you just enter the number 1 in R, that gives you a numeric object, entering 1L gives you an integer.
- There's also a special number called inf, which stands for
- Infinity.
- Example, 1/0, you'll get infinity and if you take 1 and divide it by infinity you'll get zero.
- There's another special value called NAN or Nan. And this represents an undefined value.

Attributes

- R objects can have attributes
  - Names,dimnames
  - Dimensions(matrices,arrays)
  - Class
  - Length

  * Attributes of an object can be accessed using attributes() function

Creating Vectors

- The c() function can be used to create vectors of objects
  - x<-c(0.5,0.6)
  - x<-c(TRUE,FALSE)
  - x<-c(T,F)
  - x<-c(“a”,”b”,”c”)
  - x<-9:29
  - x<-c(1+0i,2+4i)

- Using the vector function
  - x<-vector(“numeric”,length=10)

Mixing Objects

- y<-c(1.7,’a’)
- y<-c(TRUE,2)
- y<-c(“a”,TRUE)

  - When different objects are mixed in a vector, coercion occurs so that every element in the vector is of the same class. → Implicit coercion

Explicit Coercion

- Objects can be explicitly coerced from one class to another using the as. Function.
  - x<-0:6
  - class(x)
  - as.numeric(x)
  - as.logical(x)
  - as.character(x)

Lists

- Lists are a special type of vector that can contain elements of different classes. Lists are a very important data type in R.
- x<-list(1,”a”,TRUE,1+4i)
- x

Matrices

- Matrices are the vectors with a dimension attribute. The dimension attribute is itself an integer vector of length 2 (nrow,ncol)
  - m<-matrix(nrow=2,ncol=3)
  - dim(m)
  - m<-matrix(1:6, nrow=2,ncol=3)
  - m<-1:10
    dim(m)<-c(2,5)
  - x<-1:10
    y<-2:11
    cbind(x,y)
    rbind(x,y)

Decision Making

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

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

Loops

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

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

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

Functions

- The different parts of a function are −
  - **Function Name**
  - **Arguments**
  - **Function Body**
  - **Return Value**

```
function_name <- function(arg_1, arg_2, ...)
{ Function body }
```