

Application Development Laboratory (CS 33002)

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

School of Computer Engineering



Strictly for internal circulation (within KIIT) and reference only. Not for outside circulation without permission

2 Credit

Analytics Application Development using R

Protocol



2

- ☐ Students should be regular and come prepared for the lab practice.
- ☐ In case a student misses a class, it is his/her responsibility to complete the missed experiment(s).
- ☐ Students should bring the observation book, prescribed textbook and class note for the reference.
- ☐ Once the experiment(s) get executed, they should show the results to the instructors and copy the same in their observation book.
- ☐ The experiments have to be implemented using R-Studio.
- ☐ Weekly evaluation of lab record
- ☐ Written quiz and written viva should be conducted at any point of time and without prior notice.

Evaluation



3

Evaluation Procedure

Type	Mark
Continuous Evaluation	60
End Semester Lab Examination	40

Scheme for Continuous Evaluation

Type	Mark
Record (10 X 1)	10
Written Quiz (2 X 5)	10
In-class program execution (10 X 2)	20
Written Viva (1 X 10)	10
Class participation (10 X 1)	10

Sessional Evaluation Scheme

Type	Mark
Solution Approach	20
Program Execution	20

Lab Objectives



4

- ☐ Explore and understand how to use the R for analytics application development.
- ☐ Expand R by installing R packages for analytics application development.
- ☐ Read structured Data into R from various sources for analytics application development.
- ☐ Understand the different data types and structures in R for analytics application development.
- ☐ Using R for mathematical operations for analytics application development.
- ☐ Write user-defined and looping constructs R functions for analytics application development.
- ☐ Reshape data to support different analyses for analytics application development.

Lab Outcome



5

- ☐ Understand and implement the basics of R for analytics application development.
- ☐ Understand and implement the basics data structure like vector, list, matrices , arrays, data frames in R for analytics application development.
- ☐ Understand and implement different visualisations in R for analytics application development.
- ☐ Understand and implement reading, exporting, manipulating and processing data in R for analytics application development.
- ☐ Understand and implement statistics in R for analytics application development.
- ☐ Demonstrate the knowledge of data analytics and implement different task in R for analytics application development.

Lab Contents



6

Sr #	Major and Detailed Coverage Area	Lab#
1	<ul style="list-style-type: none"><input type="checkbox"/> R-Overview<input type="checkbox"/> What is R used for<input type="checkbox"/> Companies using R<input type="checkbox"/> Environment Setup<input type="checkbox"/> Data Types<input type="checkbox"/> Variables<input type="checkbox"/> Operators<input type="checkbox"/> Basic Syntax	1

R Overview



7

- ❑ R is a programming language and software environment for data analysis, statistical analysis, graphics representation and reporting.
- ❑ R was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand in the early 1990, and is currently developed by the R Development Core Team.
- ❑ R is a high-level interpreted computer language (sometimes called scripting language). Everything is pitched toward helping you analyze data.
- ❑ It is a GNU project, which means that it is free, open source software.
- ❑ R supports a mixture of programming paradigms. At its core, it is an imperative language (you write a script that does one calculation after another), but it also supports object oriented programming (data and functions are combined inside classes) and functional programming (functions are first-class objects; you treat them like any other variable, and you can call them recursively).

Congratulation !



8

You've just begun your quest to become an R programmer. So you don't pull any mental muscles, this starts you off gently with a nice warm-up.



What is R used for?



9

- ☐ Statistical inference
- ☐ Data analysis
- ☐ Machine learning algorithm

Companies Using R



10

Company	Application/Contribution
Twitter	Monitor user experience
Ford	Analyse social media to support design decisions for their cars
New York Times	data journalism
Human Rights Data Analysis Group	Measure the impact of war
Google	Created the R style guide for the R user community inside Google
Microsoft	Released Microsoft R Open, an enhanced R distribution and Microsoft R server after acquiring Revolution Analytics in 2015

Installation



11

<https://www.rstudio.com/products/rstudio/download/>

- ☐ It is an R-specific IDE. That means that you lose the ability to code (easily) in multiple languages, but you do get some features especially for R.
- ☐ Use it if you mainly write R code, don't need advanced editor features, and want a shallow learning curve or the ability to run remotely.

Your first few Programs



12

```
# Print Hello World.  
print("Hello World")
```

```
# Add two numbers.  
print(23.9+11.6)
```

```
# Mean of 1 to 5  
print(mean(1:5))
```

Well done! You've calculated
a statistic using R.

Get Help in R



13

Before you get started writing R code, the most important thing to know is how to get help. There are lots of ways to do this. Firstly, if you want help on a function or a dataset that you know the name of, type ? followed by the name of the function. To find functions, type two question marks (??) followed by a keyword related to the problem to search. Special characters, reserved words, and multiword search terms need enclosing in double or single quotes.

For example:

?mean #opens the help page for the mean function

?"+" #opens the help page for addition

? "if" #opens the help page for if, used for branching code

??plotting #searches for topics containing words like "plotting"

?? "regression model" #searches for topics containing phrases like this

Get Help in R cont'd



14

The functions `help` and `help.search` do the same things as `?` and `??`, respectively, but with these you always need to enclose your arguments in quotes. The following commands are equivalent to the previous lot:

```
help("mean")
```

```
help("+")
```

```
help("if")
```

```
help.search("plotting")
```

```
help.search("regression model")
```

R Nuts and Bolts



15

Entering Input: At the R prompt we type expressions. The `<-` symbol is the assignment operator.

```
x <- 1
```

```
print(x)
```

Evaluation: When a complete expression is entered, it is evaluated and the result of the evaluated expression is returned. The result may be auto-printed.

```
x <- 5 ## nothing printed
```

```
x ## auto-printing occurs
```

```
print(x) ## explicit printing
```

[1] 5 => The `[1]` shown in the output indicates that `x` is a vector and 5 is its first element.

R- Data Types



16

Generally, while doing programming in any programming language, you need to use various variables to store various information. Variables are nothing but reserved memory locations to store values. This means that, when you create a variable you reserve some space in memory.

You may like to store information of various data types like character, integer, floating point, double floating point, boolean etc. Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory. 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

Vector Object



17

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.

Data Type	Example	Verify	Output
Logical	TRUE, FALSE	<pre>v <- TRUE print(class(v))</pre>	[1] "logical"
Numeric	1 2.3, 5, 999	<pre>v <- 23.5 print(class(v))</pre>	[1] "numeric"
Integer	2L, 34L, 0L	<pre>v <- 2L print(class(v))</pre>	[1] "integer"
Complex	3 + 2i	<pre>v <- 5+10i print(class(v))</pre>	[1] "complex"
Character	'a' , "good", "TRUE", '23.4'	<pre>v <- "TRUE" print(class(v))</pre>	[1] "character"
Raw	"Hello" is stored as 48 65 6c 6c 6f	<pre>v <- charToRaw ("TRUE") print(class(v))</pre>	[1] "raw"

Vector Object cont'd



18

When the vector to be created with more than one element, `c()` function should be used which means to combine the elements into a vector.

Create a vector.

```
apple <-c('red','green',"yellow")
```

```
print(apple)
```

Get the class of the vector.

```
print(class(apple))
```

Subsetting a Vector



19

Vectors can be subsetting using the [] operator.

```
x <- c("a", "b", "c", "c", "d", "a")
```

```
x[1] ## Extract the first element and the output [1] "a"
```

```
x[2] ## Extract the second element and the output [1] "b"
```

The [] operator can be used to extract multiple elements of a vector by passing the operator an integer sequence.

```
x[1:4] and the output [1] "a" "b" "c" "c"
```

The sequence does not have to be in order; you can specify any arbitrary integer vector.

```
x[c(1, 3, 4)] and the output is [1] "a" "c" "c"
```

List Object



20

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.
```

```
# Print the list.
```

```
list1 <-list(c(2,5,3),21.3,sin)
```

```
print(list1)
```

Output:- When we execute the above code, it produces the following result:

```
[[1]]
```

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

```
[[2]]
```

```
[1] 21.3
```

```
[[3]]
```

```
function (x) .Primitive("sin")
```

Matrices Object



21

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:- When we execute the above code, it produces the following result:

```
 [,1] [,2] [,3]
```

```
[1,] "a"  "a"  "b"
```

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

Subsetting a Matrix



22

Matrices can be subsetting in the usual way with [i,j] type indices.

```
x <- matrix(1:6, 2, 3)
```

```
print(x)
```

Output:

```
 [,1] [,2] [,3]
```

```
[1,] 1   3   5
```

```
[2,] 2   4   6
```

x[1, 2] and the output is **[1] 3**

x[2, 1] and the output is **[1] 2**

Indices can also be missing. This behavior is used to access entire rows or columns of a matrix.

x[1,] ## Extract the first row and the output is [1] 1 3 5

x[, 2] ## Extract the second column and the output is [1] 3 4

Arrays Object



23

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 dimension. 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)
```

Output:- When we execute the above code, it produces the following result:

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

Factors Object



24

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 modeling.

```
# 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 Object



25

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)
```

R-Variables



26

A variable provides named storage that the programs can manipulate. A variable in R can store an atomic vector, group of atomic vectors or a combination of many R - objects. A valid variable name consists of letters, numbers and the dot or underline characters. The variable name starts with a letter or the dot not followed by a number.

Variable Assignment

The variables can be assigned values using leftward, rightward and equal to operator.

Assignment using equal operator.

```
var.1 = c(0,1,2,3)
```

Assignment using leftward operator.

```
var.2<-c("learn","R")
```

Assignment using rightward operator.

```
c(TRUE,1)->var.3
```

The values of the variables can be printed using print() or cat() function. The cat() function combines multiple items into a continuous print output.

```
print(var.1)
```

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

R-Variables cont'd



27

Data Type of a Variable

In R, a variable itself is not declared of any data type, rather it gets the data type of the R - object assigned to it. So R is called a **dynamically typed language**, which means that we can change a variable's data type of the same variable again and again when using it in a program.

```
var_x <- "Hello"
```

```
cat("The class of var_x is ", class(var_x), "\n")
```

```
var_x <- 34.5
```

```
cat("Now the class of var_x is ", class(var_x), "\n")
```

```
var_x <- 27L
```

```
cat(" Next the class of var_x becomes ", class(var_x), "\n")
```

Finding Variables

To know all the variables currently available in the workspace we use the `ls()` function. E.g. `print(ls())`

Deleting Variables

Variables can be deleted by using the `rm()` function. E.g. `rm(var.3)`

R-Operators



28

An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. R language is rich in built-in operators and provides following types of operators.

Types of Operators

- ☐ Arithmetic Operators
- ☐ Relational Operators
- ☐ Logical Operators
- ☐ Assignment Operators
- ☐ Miscellaneous Operators

R-Operators : Arithmetic Operators



29

```
v <-c(2,5.5,6)
```

```
t <-c(8,3,4)
```

```
print(v+t)
```

```
print(v-t)
```

```
print(v*t)
```

```
print(v/t)
```

```
print(v%%t) # Give the remainder of the first vector with the second
```

```
print(v%/%t) # Give the result of division of first vector with second (quotient)
```

```
print(v^t) # The first vector raised to the exponent of second vector
```

R-Operators cont'd



30

Relational Operators

```
v <-c(2,5.5,6)
```

```
t <-c(8,3,4)
```

```
print(v>t)
```

```
print(v<t)
```

```
print(v==t)
```

```
print(v<=t)
```

```
print(v>=t)
```

```
print(v!=t)
```

Logical Operators

```
v <-c(2,TRUE,6)
```

```
t <-c(8,FALSE,4)
```

```
print(v&t) # Element - wise Logical AND operator
```

```
print(v|t) # Element - wise Logical OR operator
```

```
print(!v)
```

```
print(v&& t) # Logical AND operator
```

```
print(v||t) # Logical OR operator
```

R-Operators cont'd



31

Left Assignment Operator

```
v <-c(2,5.5,6)
```

```
t <<-c(8,3,4)
```

```
u = c(18,13,14)
```

```
print(v)
```

```
print(t)
```

```
print(u)
```

Right Assignment Operator

```
c(2,5.5,6) -> v
```

```
c(8,3,4) ->> t
```

```
print(v)
```

```
print(t)
```

Miscellaneous Operator

```
v <-2:8
```

```
print(v)
```

Colon operator: It creates the series of numbers in sequence for a vector

```
v1 <-8
```

```
v2 <-12
```

```
t <-1:10
```

```
print(v1 %in% t)
```

```
print(v2 %in% t)
```

%in% - This operator is used to identify if an element belongs to a vector.

Special Numbers



32

To help with arithmetic, R supports four special numeric values: Inf, -Inf, NaN, and NA. The first two are, of course, positive and negative infinity. NaN is short for “not-a-number,” and means that calculation either didn’t make mathematical sense or could not be performed properly. NA is short for “not available” and represents a missing value—a problem all too common in data analysis.

Lab Exercise

```
print(c(Inf + 1, Inf - 1, Inf - Inf))  
print(c(1 / Inf, Inf / 1, Inf / Inf))  
print(c(sqrt(Inf), sin(Inf)))  
print(c(log(Inf), log(Inf, base = Inf)))  
print(c(NA + 1, NA * 5, NA + Inf))  
print(c(NA + NA, NaN + NaN, NaN + NA, NA + NaN))
```


Thank You

End of Lab 1

Lab Experiments



34

1. Initialize some variables in the R workspace. Now analyze and display what are variables are created, then delete the single variable as well as all the created variables.
2. Initialize some variables with different types of value. Now analyze what is the type of those variables.
3. Write an R-script to initialize your roll no., name and branch then display all the details.
4. Write an R-script to initialize two variables, then find out the sum, multiplication, subtraction and division of them.
5. Write an R-script to enter 3 numbers from the keyboard, then find out sum of all the 3 numbers.
6. Write an R-script to enter the radius of a circle, then calculate the area and circumference of the circle.
7. Write an R-script to calculate the compound interest of the given P, T, R.
8. Write an R-script to enter two numbers from the keyboard, then swap them without using 3rd variable.

Lab Experiments cont...



35

9. Write an R-script to enter two numbers and implement all the relational operators on that two numbers.
10. Write an R-script to convert given paisa into its equivalent rupee and paisa as per the following format. 550 paisa = 5 Rupee and 50 paisa.
11. Write an R-script to convert given second into its equivalent hour, minute and second as per the following format. Example. 7560 second = 2 Hour, 27 Minute and 40 Second.
12. Write an R-script to convert a quantity in meter entered through keyboard into its equivalent kilometer & meter as per the following format. Example - 2430 meter = 2 Km and 430 meter.
13. A cashier has currency notes of denominations 10, 50 and 100. If the amount to be withdrawn is input through the keyboard in hundreds, write an R-script to find the total number of currency notes of each denomination the cashier will have to give to the withdrawer.
14. Ramesh's basic salary is input through the keyboard. His dearness allowance is 40% of basic salary, and house rent allowance is 20% of basic salary. Write an R-script to calculate his gross salary.