

MU ADBMS Lab Reference Material For R - 2020

Meenakshi Garg
July 27, 2020

Experiment 1

Basic R commands

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

## [1] "Hello, World!"

#setwd() - sets working directory.
setwd("E:/R Orientation")
##getwd() - gets current working directory.
getwd()

## [1] "E:/R Orientation"

## dir() - lists the contents of current working directory.
dir()

## [1] "CreditDefaulters.csv" "data.csv"
## [3] "Descriptive_DataSet.csv" "e2.xls"
## [5] "e2.xlsx" "employeeinfo.csv"
## [7] "employeeinfo.xlsx" "employeeinfoMissingData (1).csv" ## [9]
"employeeinfonew.xls" "employment.csv"
## [11] "Experiment" "first.Rmd"
## [13] "first.tex" "groceries.csv"
## [15] "insuranceCharges.csv" "MU-data.html"
## [17] "MU-data.log" "MU-data.tex"
## [19] "MU-data2.html" "MU-data2new.aux"
## [21] "MU-data2new.html" "MU-data2new.log"
## [23] "MU-data2new.out" "MU-data2new.pdf"
## [25] "MU-data2new.Rmd" "MU-data2new.tex"
## [27] "MU data.Rmd" "MU data2.Rmd"
## [29] "MU data2new.Rmd" "mydata.csv"
## [31] "sample.txt" "secod.html"
## [33] "secod.Rmd" "secod.tex"
## [35] "trial.html" "trial.Rmd"

##ls() - lists names of objects in R environment
ls()

## [1] "myString"

##help.start() - provides general help.
##help.start()
##data() - lists all example datasets in currently loaded packages. ##data()
## library() - lists all available packages.
##library()
```

Exp -1

```
#Creating and assigning to a variable:
#alternative: x=1
x<-1
## Checking the type of variable:
class(x)

## [1] "numeric"
#Printing a variable:
#auto-printing
x

## [1] 1
#explicit printing
print(x)

## [1] 1
## is., as. functions: R has is.* and as.* family of functions that can be used to check whether a variable is of a certain class
#check if character
is.character(x)

## [1] TRUE
#check if integer
is.integer(x)

## [1] FALSE
y<-2.14
as.integer(y)

## [1] 2
###Creating Vector: contains objects of same class.
#using c() function
x<-c(11.3,27.5,33.8)
#using vector() function
y<-vector("logical", length=10)
#length of vector x
length(x)

## [1] 3
y

## [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
#Vector operations: Various arithmetic operations can be performed member-wise. Like: #-
#Multiplication by a scalar.
#- Addition of two vectors.
#- Multiplication of two vectors
y<-c(4,5,6)
#multiplication by a scalar
5*x

## [1] 56.5 137.5 169.0
```

#addition of two vectors x+y

#multiplication of two vectors

`x*y`

[1] 45.2 137.5 202.8

#x to the power y

`x^y`

[1] 1.630474e+04 1.572764e+07 1.491077e+09

###Creating Matrix: Two-dimensional array having elements of same class.

#using matrix() function

`m<-matrix(c(11,12,13,55,60,65,66,72,78),nrow=3,ncol=3)`

`m`

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

[1,] 11 55 66

[2,] 12 60 72

[3,] 13 65 78

#dimensions of matrix m

`dim(m)`

[1] 3 3

#attributes of matrix m

`attributes(m)`

\$dim

[1] 3 3

#By default, elements in matrix are filled by column. "byrow" attribute of matrix() can be used to

fill`m<-matrix(c(11,12,13,55,60,65,66,72,78),nrow=3,ncol=3,byrow = TRUE)`

`m`

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

[1,] 11 12 13

[2,] 55 60 65

[3,] 66 72 78

#cbind-ing and rbind-ing:

#By using cbind() and rbind() functions

`x<-c(1,2,3)`

`y<-c(11,12,13)`

`cbind(x,y)`

x y

[1,] 1 11

[2,] 2 12

[3,] 3 13

`rbind(x,y)`

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

x 1 2 3

y 11 12 13

##Matrix operations/functions:

#Multiplication by a scalar.

#Addition, subtraction and multiplication of two matrices.

#Transpose, determinant of a matrix. etc.

#multiplication by a scalar

p<-3*m

p

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

[1,] 33 36 39

[2,] 165 180 195

[3,] 198 216 234

n<-matrix(c(4,5,6,14,15,16,24,25,26),nrow=3,ncol=3)

#addition of two matrices

q<-m+n

q

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

[1,] 15 26 37

[2,] 60 75 90

[3,] 72 88 104

o<-matrix(c(4,5,6,14,15,16),nrow=3,ncol=2)

o

[,1] [,2]

[1,] 4 14

[2,] 5 15

[3,] 6 16

#matrix multiplication by using %%*

r<-m %*% o

r

[,1] [,2]

[1,] 182 542

[2,] 910 2710

[3,] 1092 3252

#transpose of matrix

mdash<-t(m)

mdash

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

[1,] 11 55 66

[2,] 12 60 72

[3,] 13 65 78

s<-matrix(c(4,5,6,14,15,16,24,25,26), nrow=3,ncol=3,byrow=TRUE)

#determinant of s

s_det<-det(s)

s_det

[1] 1.110223e-14

####List: A special type of vector containing elements of different classes. #####Elements of list can be accessed by giving element index or name in [[]].

#using list() function

x<-list(1,"p",TRUE,2+4i)

x

```
## [[1]]
## [1] 1
##
## [[2]]
## [1] "p"
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] 2+4i
```

####Factor: Represents categorical data. Can be ordered or unordered.

```
status<-c("low","high","medium","high","low")
```

#using factor() function

```
x<-factor(status, ordered=TRUE,levels=c("low","medium","high")) x
```

```
## [1] low high medium high low
```

```
## Levels: low < medium < high
```

##levels' argument is used to set the order of levels. #First level forms the baseline level.

Without any order, levels are called nominal. Ex. - Type1, Type2, . # With order, levels are called ordinal. Ex. - low, medium, .

####Data frame: Used to store tabular data. Can contain different classes.

```
student_id<-c(1,2,3)
```

```
student_names<-c("Ram","Shyam","Laxman")
```

```
position<-c("First","Second","Third")
```

#using data.frame() function

```
data<-data.frame(student_id,student_names,position)
```

```
data
```

```
## student_id student_names position
```

```
## 1 1 Ram First
```

```
## 2 2 Shyam Second
```

```
## 3 3 Laxman Third
```

#accessing a particular column

```
data$student_id
```

```
## [1] 1 2 3
```

#no. of rows in data

```
nrow(data)
```

```
## [1] 3
```

#no. of columns in data

```
ncol(data)
```

```
## [1] 3
```

#column names of data. for a dataframe, colnames() can also be used.

```
names(data)
```

```
## [1] "student_id" "student_names" "position" ###Table
```

command is used to create a 2dimensional table in R

```
smoke <- matrix(c(51,43,22,92,28,21,68,22,9),ncol=3,byrow=TRUE)
colnames(smoke) <- c("High","Low","Middle")
rownames(smoke) <- c("current","former","never")
smoke <- as.table(smoke)
smoke
```

```
## High Low Middle
```

```
## current 51 43 22
```

```
## former 92 28 21
```

```
## never 68 22 9
```

```
###installing Package
```

```
#install.packages("XLConnect")
```

```
#library(XLConnect)
```

```
#install.packages("readxl")
```

```
#library(readxl)
```

```
#install.packages("writexl")
```

```
#library(writexl)
```

```
#uncomment this to read data from Excel
```

Reading and writing data from csv

```
dataT <- read.table("mydata.csv", sep = ",", header = T) dataT
```

```
## Name Rollno Class Percent.Marks
```

```
## 1 Sita 23 IV 67%
```

```
## 2 Rita 28 V 98%
```

```
## 3 Naresh 69 VII
```

```
## 4 Mohit 88 V 95%
```

```
# dimension
```

```
dim(dataT)
```

```
## [1] 4 4
```

```
# Load just few lines at the top or bottom
```

```
head(dataT, 2)
```

```
## Name Rollno Class Percent.Marks
```

```
## 1 Sita 23 IV 67%
```

```
## 2 Rita 28 V 98%
```

```
tail(dataT, 2)
```

```
## Name Rollno Class Percent.Marks
```

```
## 3 Naresh 69 VII
```

```
## 4 Mohit 88 V 95%
```

```
z <- data.frame(a = 5, b = 10, c = pi)
```

```
write.csv(z,file="data.csv")
```

Reading and writing data from Excel using XLConnect

```
dataX <- XLConnect::readWorksheetFromFile("employeeinfo.xlsx",sheet=1)
dataX
```

```
## id name salary start_date_of_the_employee_as_per_records dept ## 1 1 Rick 623.30
2012-01-01 IT ## 2 2 Dan 515.20 2013-09-23 Operations ## 3 3 Michelle 611.00 2014-11-15
IT ## 4 4 Ryan 729.00 2014-05-11 HR ## 5 NA Gary 843.25 2015-03-27 Finance ## 6 6 Nina
578.00 2013-05-21 IT ## 7 7 Simon 632.80 2013-07-30 Operations ## 8 8 Guru 722.50
2014-06-17 Finance
## 9 9 John 478.00 2012-05-21 <NA> ## 10 10 Rock 600.80 2013-07-30 HR ## 11 11 Brad
1032.80 2013-07-30 Operations ## 12 12 Ryan 729.00 2014-05-11 HR # Following is called
Subsetting - It will print rows from 1 to 2 and all columns
dataY<- dataX[1:2,]
dataY
```

```
## id name salary start_date_of_the_employee_as_per_records dept ## 1 1 Rick
623.3 2012-01-01 IT ## 2 2 Dan 515.2 2013-09-23 Operations
```

Reading and writing data from Excel using readXL and writeXL

```
#uncomment Package install
#data2 <- read_excel("employeeinfonew.xls", sheet = "employeeinfo") #data2
#data3<- data2[1:2,]
#write_xlsx(data3, "e2.xlsx")

# create an empty data frame
data <- data.frame(Name=character(), Age=numeric())
# load interface and assign edited values to data back - uncomment following #data <-
edit(data)
# print those values
#data
```

MU ADBMS Lab Reference Material -2020
Exp -1