|  |
| --- |
|  |
| **SUBJECT CODE &NAME: CS012 R Programmning** |
| **Course Coordinator: Dr. S. Palaniappan**  **Course Faculties: Dr. Golda, Dr. Malathi, Dr. Jose and Mrs.M. Rajasuguna** |

**Tutorial Book / Solutions**

**Course Faculty HOD**

**Unit 1**

1. i. Mention some software environment used for Statistical Analysis.

## [MaxStat](http://www.capterra.com/statistical-analysis-software/spotlight/143824/MaxStat/MaxStat%20Software) by MaxStat Software

Complete statistics package with intuitive user interface and easily understandable results. Designed for researchers and students

[**SPSS**](http://www.capterra.com/statistical-analysis-software/spotlight/13990/SPSS/IBM)**by IBM**

Predictive Analytics can uncover unexpected patterns and associations and develop models to guide front-line interaction

[**Minitab 17**](http://www.capterra.com/statistical-analysis-software/spotlight/109731/Minitab%2017/Minitab)**by Minitab**

Analyze your data and improve your products and services with the leading statistical software used for quality improvement worldwide.

[**DataMelt ("Dmelt")**](http://jwork.org/dmelt)**by jWork.ORG**

Data analysis, math and data visuzalization program which combines the power of Python and Java (free)

[**Analytica**](http://www.capterra.com/statistical-analysis-software/spotlight/129545/Analytica/Lumina%20Decision%20Systems)**by Lumina Decision Systems**

Analytica is a powerful, stand-alone application for visual quantitative modeling with a full array of statistical analysis functions.

1. Name some industry using R

The following is a list of top brands or large organizations using R.

1. Facebook – For behavior analysis related to status updates and profile pictures.
2. Google – For advertising effectiveness and economic forecasting.
3. Twitter – For data visualization and semantic clustering
4. Microsoft – Acquired Revolution R company and use it for a variety of purposes.
5. Uber – For statistical analysis
6. Airbnb – Scale data science.
7. IBM – Joined R Consortium Group
8. ANZ – For credit risk modeling
9. HP
10. Ford
11. If you are a data scientist, Which Application Area you will select? Mention the reason for selecting that application area?

Healthcare Application

Reason : wide scope

It useful for society to create awareness

Learn more

1. What are all the modification you will do to update the application for better improvement?
   * Rural Area healthcare system
   * Improvement of Medical Facility
2. a. Calculate the square roots of 729

sqrt(729)

27

b. Create a numeric variable b that display Republic year of India.

b->1950

c. In each case, what is the value of x? (Try to think it through before you try it in R)

* + 1. x<-2-1\*2

Ans: 0

* + 1. x<-6/3-2+1\*0+3/3-3

Ans:-2

1. a. Give the R code required to produce the list 10 20 30 40

list1<-list(c(10,20,30,40)

print(list1)

b. Create in R the Matrices x = 3 2 4 5 and y = 6 7 5 4; Find the following and check your answers in R

> x<- matrix (3 2 4 5, 2:2)

> x

|  |  |  |
| --- | --- | --- |
|  | [,1 ] | [,2 ] |
| [1, ] | 3 | 4 |
| [2, ] | 2 | 5 |

> y<- matrix (6 7 5 4 , 2:2)

> y

|  |  |  |
| --- | --- | --- |
|  | [,1 ] | [,2 ] |
| [1, ] | 6 | 5 |
| [2, ] | 7 | 4 |

i) 2 \* x

> c <- 2 \* x

|  |  |  |
| --- | --- | --- |
|  | [,1 ] | [,2 ] |
| [1, ] | 6 | 8 |
| [2, ] | 4 | 10 |

ii) y \* y

<- y \* y

|  |  |  |
| --- | --- | --- |
|  | [,1 ] | [,2 ] |
| [1, ] | 71 | 50 |
| [2, ] | 70 | 51 |

iii) t (y)

> t(y)

|  |  |  |
| --- | --- | --- |
|  | [,1 ] | [,2 ] |
| [1, ] | 6 | 7 |
| [2, ] | 5 | 4 |

iv) x % \* % y

<- x % \* % y

|  |  |  |
| --- | --- | --- |
|  | [,1 ] | [,2 ] |
| [1, ] | 46 | 31 |
| [2, ] | 47 | 30 |

c. Create an array which shows your favourite food names

Solution:

ar<array(data=1:27,dim=c(3,3,3),dimnames=list(c(“Pizza”,”sandwitch”,”Iddly”),c(“vada”,”dosa”,”poori”),c(“bread”,”biriyani”,”chicken65”)))

dimnames(ar)[[3]]<-c(“bread”,”biriyani”,”chicken65”)

1. Create a data frame of 5 sonar company Employee details.

Solution:

Id<-c(1:5)

Name<-c(“mala”,”raj”,kala”,Mani”,”Hasid”)

Occupation<-c(“Doctor”,”Software Engineer”,”Pilot”,”Driver”,”Teacher”)

Employee<-data.frame(id,Name,Occupation)

4. a. Decide what the following sequences are and use R to check your answers

i) 7:11

7 8 9 10 11

ii) seq (2,9)

2 3 4 5 6 7 8 9

iii) seq (6,-4,by=-2)

6 4 2 0 -2 -4

iv) rep (2,4)

2 2 2 2

v) rep (c(1,2),4)

1 2 1 2 1 2 1 2

b. Create and Define matrices x(3 rows,3 columns), y(3 rows,3 columns) decide What the result will be of the following

> x<- matrix (1:9, 3:3)

> x

|  |  |  |  |
| --- | --- | --- | --- |
|  | [,1 ] | [,2 ] | [,3 ] |
| 1, ] | 1 | 4 | 7 |
| [2, ] | 2 | 5 | 8 |
| [3, ] | 3 | 6 | 9 |

> y <- matrix (9:1, 3:3)

> y

|  |  |  |  |
| --- | --- | --- | --- |
|  | [,1 ] | [,2 ] | [,3 ] |
| [1, ] | 9 | 6 | 3 |
| [2, ] | 8 | 5 | 2 |
| [3, ] | 7 | 4 | 1 |

i) 2\*x

>c<- 2

>c\*x

|  |  |  |  |
| --- | --- | --- | --- |
|  | [,1 ] | [,2 ] | [,3 ] |
| 1, ] | 2 | 8 | 14 |
| [2, ] | 4 | 10 | 16 |
| [3, ] | 6 | 12 | 18 |

ii) y\*y

> y\*y

|  |  |  |  |
| --- | --- | --- | --- |
|  | [,1 ] | [,2 ] | [,3 ] |
| [1, ] | 81 | 36 | 9 |
| [2, ] | 64 | 25 | 4 |
| [3, ] | 49 | 16 | 1 |

iii) t(y)

> t(y)

|  |  |  |  |
| --- | --- | --- | --- |
|  | [,1 ] | [,2 ] | [,3 ] |
| [1, ] | 9 | 8 | 7 |
| [2, ] | 6 | 5 | 4 |
| [3, ] | 3 | 2 | 1 |

iv) x%\*%y

<- x%\*%y

|  |  |  |  |
| --- | --- | --- | --- |
|  | [,1 ] | [,2 ] | [,3 ] |
| [1, ] | 90 | 54 | 18 |
| [2, ] | 114 | 69 | 24 |
| [3, ] | 138 | 84 | 30 |

c. Create 2 vectors of different length

1. Take these vectors as input to the array.

Vector1<-c(8,9,1)

Vector2<-c(20,21,22,23,24,25)

1. You can provide names to the rows, columns along with matrices within the array by using the 'dimnames' parameter.

Result<array(c(vector1,vector2),dim=(3,3,2),dimnames=list(row.names,columnnames,matrixnames))

5. Create an s3 object of class fruit can be created

i) Create a list with following required components name, quantity, cost

> # create a list with required components name=character, quantity=numeric,cost=numeric

> f <- list(name = "mango", Quantity = 11, cost = 36.5)

> # name the class appropriately

> class(f) <- "fruit"

> #object of class "fruit"

> f

1. Define and create s4 objects.

setClass(“fruit”, slots=list(name=”character”, quantity=”numeric”cost=”numeric”))

f <- new (“fruit”, name = "mango", Quantity = 11, cost = 36.5)

1. Define a reference class of fruit

>setRefClass(“fruit”)

>setRefClass(“fruit”, fields = list(name =”character”, quantity=”numeric” cost=”numeric”))

>fruit<- setRefClass(“fruit”, fields = list(name =”character”, quantity=”numeric” cost=”numeric”))

>s

**Unit 2**

**FOR LOOP**

1. Write a for loop that iterates over the numbers 1 to 7 and prints the cube of each number using print().

Solution:

for (i in 1:7) {

print(i^3)

}

## [1] 1

## [1] 8

## [1] 27

## [1] 64

## [1] 125

## [1] 216

## [1] 343

1. Write a for loop that iterates over the column names of the inbuilt iris dataset and print each together with the number of characters in the column name in parenthesis. Example output: Sepal.Length (12). Use the following functions print(), paste0() and nchar().

Solution:

for (n in names(iris)) {

print(paste0(n, " (", nchar(n), ")"))

}

## [1] "Sepal.Length (12)"

## [1] "Sepal.Width (11)"

## [1] "Petal.Length (12)"

## [1] "Petal.Width (11)"

## [1] "Species (7)"

**WHILE LOOP**

Write a while loop starting with x = 0. The loop prints all numbers up to 35 but it skips number 7.

#expected result

[1] 1

[1] 2

[1] 3

[1] 4

[1] 5

[1] 6

[1] 8

[1] 9

[1] 10

[1] 11

[...]

Solution :

x = 0  
  
while(x < 35) {  
    x = x+1   
    if (x == 7) next   
    print(x)}

**REPEAT LOOP**

Write a repeat loop containing three random numbers. The loop repeats itself exactly ten times before it stops.

|  |
| --- |
| #expected result    [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671  [1] 0.1932123 -0.4346821 0.9132671 |

**Solution:**

set.seed(23)  
randomnr <- rnorm(3)  
reps <- 1  
  
repeat {  
    print(randomnr)  
    reps <- reps + 1  
  
    if(reps > 10) {  
       break}}

**BREAK**

Write a while loop that prints out standard random normal numbers (use rnorm()) but stops (breaks) if you get a number bigger than 1.

Solution:

set.seed(3)

while (TRUE) {

x <- rnorm(1)

print(x)

if (x > 1) {

break

}

}

## [1] -0.9619334

## [1] -0.2925257

## [1] 0.2587882

## [1] -1.152132

## [1] 0.1957828

## [1] 0.03012394

## [1] 0.08541773

## [1] 1.11661

**NEXT**

Write a while loop that prints out standard random normal numbers (use rnorm()) but stops (breaks) if you get a number bigger than 1. Using **next**adapt the loop so that doesn’t print negative numbers.

**Solution:**

set.seed(3)

while (TRUE) {

x <- rnorm(1)

if (x < 0) {

next

}

print(x)

if (x > 1) {

break

}

}

## [1] 0.2587882

## [1] 0.1957828

## [1] 0.03012394

## [1] 0.08541773

## [1] 1.11661

**FOR LOOP combined with IF statement**

Write a for loop that prints the Displacement (‘disp’) of the ‘mtcars’ dataset.  
a. This loop will only print observations of 160 or higher in ‘disp’.  
b. This loop will stop as soon as an observation is smaller than 160 in ‘disp’.

|  |  |
| --- | --- |
|  | #expected result  #a  [1] 160  [1] 160  [1] 258  [1] 360  [1] 225  [1] 360  [1] 167.6  [1] 167.6  [...]    #b  [1] 160  [1] 160 |

Solution

a)

for (i in mtcars$disp){  
   if(i<160)  
     next  
   print (i)}

b)

for (i in mtcars$disp){  
   if(i<160)  
     break  
   print (i)}

**IF … ELSE**

#What is the output n of:

z='i'

if (z %in% letters)

if (z=='a') n=1 else

if (z=='e') n=2 else

if (z=='i') n=3 else

if (z=='o') n=4 else n=5

n

**Solution :**

## [1] 3

#What is the output n of:

z='u'

if (z %in% letters)

if (z=='a') n=1 else

if (z=='e') n=2 else

if (z=='i') n=3 else

if (z=='o') n=4 else n=5

n

**Solution :**

## [1] 5

**APPLY FAMILY FUNCTIONS**

**APPLY**

Titanic Casualties **– Use the standard ‘Titanic’ dataset which is part of R Base to answer the following questions.**

**a.** Use an appropriate apply function to get the sum of males vs females aboard.

|  |  |
| --- | --- |
|  | #expected result    Male Female   1731 470 |

**b.** Get a table with the sum of survivors vs sex.

|  |  |
| --- | --- |
|  | #expected result           Survived   Sex     No Yes   Male  1364 367   Female 126 344 |

**c.** Get a table with the sum of passengers by sex vs age.

|  |  |
| --- | --- |
|  | #expected result             Sex     Age  Male Female   Child    64     45   Adult  1667    425 |

**Solution**

**a.**  
apply(Titanic, 2, sum)  
  
**b.**  
apply(Titanic, c(2,4), sum)  
  
**c.**  
apply(Titanic, c(3,2), sum)

**LAPPLY , SAPPLY , MAPPLY**

Use ‘mtcars’ dataset for answering the following questions.

1. Use three ‘apply’ family functions to get the minimum values of each column of the ‘mtcars’ dataset (hint: ‘lapply’, ‘sapply’, ‘mapply’). Store each output in a separate object (‘l’, ‘s’, ‘m’) and get the following outputs.

#expected result

>l

$mpg

[1] 10.4

$cyl

[1] 4

$disp

[1] 71.1

$hp

[1] 52

$drat

[1] 2.76

$wt

[1] 1.513

$qsec

[1] 14.5

$vs

[1] 0

$am

[1] 0

$gear

[1] 3

$carb

[1] 1

>s

   mpg   cyl   disp     hp  drat    wt   qsec    vs    am  gear  carb

10.400 4.000 71.100 52.000 2.760 1.513 14.500 0.000 0.000 3.000 1.000

>m

   mpg   cyl   disp     hp  drat    wt   qsec    vs    am  gear  carb

10.400 4.000 71.100 52.000 2.760 1.513 14.500 0.000 0.000 3.000 1.000

**Solution :**

lapply(mtcars, FUN = min) -> l  
sapply(mtcars, FUN = min) -> s  
mapply(mtcars, FUN = min) -> m  
l; s; m

**b.** Put the three outputs ‘l’, ‘s’, ‘m’ in the list ‘listobjects’

Solution :

listobjects = list(l, s, m)

**c.** Use a suitable ‘apply’ function to get the class of each of the three list elements in ‘listobjects’

Solution :

sapply(FUN = class, X = listobjects)

**d.** Name the output classes for each of the three functions used in the exercise.

Solution :

'lapply' gives a list,  
'sapply' and 'mapply' give vectors per default

**tapply**

Use **iris** data set. Check the structure of the data set using **str(iris)**. Using tapply perform the following operations.

1. Calculate the mean of the Sepal Length for each Species.
2. Calculate the mean of the Petal Length for each Species.

**Solution:**

tapply(iris$Sepal.Length, iris$Species, mean)

**Output:**

setosa versicolor virginica   
5.006    5.936         6.588

**b)**

tapply(iris$Petal.Length, iris$Species, mean)

**Output:**

setosa versicolor virginica   
1.462      4.260        5.552

**Vector Recycling**

Consider two vectors:

p <- c (3, 5, 6, 8)

and

q <- c (3, 3, 3)

What is the value of:

p+q

Solution :

6, 8, 9, 11

Predict the output of the following

> c(1,2,3,4,5,6) + c(1,3)

**Solution :**

[1] 2 4 3 7 6 9

The c(1,3) vector repeated itself to form c(1,3,1,3,1,3) so that it could successfully match the previous term.

Problem

> c(1,2,3,4,5) + c(1,3)

**Solution :**

[1] 2 5 4 7 6

Warning message:

In c(1, 2, 3, 4, 5) + c(1, 3) :

longer object length is not a multiple of shorter object length

**Problem**

> c(1,2,3) \* c(0,3,6)

**Solution :**

[1] 0 6 18

> c(1,3,5) \* c(2,4)

**Solution :**

[1] 2 12 10

Warning message:

In c(1, 3, 5) \* c(2, 4) :

longer object length is not a multiple of shorter object length

**FUNCTIONS**

Write a function called kelvin\_to\_celsius() that takes a temperature in Kelvin and returns that temperature in Celsius.

Hint: To convert from Kelvin to Celsius you subtract 273.15

**Solution:**

kelvin\_to\_celsius <- function(temp) {

celsius <- temp - 273.15

return(celsius)

}

Define two functions that will convert temperature from Fahrenheit to Kelvin, and Kelvin to Celsius.

**Solution:**

fahr\_to\_kelvin <- function(temp) {

kelvin <- ((temp - 32) \* (5 / 9)) + 273.15

return(kelvin)

}

kelvin\_to\_celsius <- function(temp) {

celsius <- temp - 273.15

return(celsius)

}

Define the function to convert directly from Fahrenheit to Celsius, by reusing the two functions above (or using your own functions if you prefer).

**Solution:**

fahr\_to\_celsius <- function(temp) {

temp\_k <- fahr\_to\_kelvin(temp)

result <- kelvin\_to\_celsius(temp\_k)

return(result)

}

**RECURSION**

Convert decimal number into binary by creating a recursive function, convert\_to\_binary().

**Solution**

# Program to convert decimal number into binary number using recursive function

convert\_to\_binary <- function(n) {

if(n > 1) {

convert\_to\_binary(as.integer(n/2))

}

cat(n %% 2)

}

**Output**

> convert\_to\_binary(52)

110100

**Unit 3**

1. **Find the difference between Data Frames and other Data Structures with example.**

**Solution:**

**Data Structure:**

There **is** also an array **data structure** that extends this idea to more than two dimensions. A collection **of** vectors that all have the same length. This **is** like a matrix, except that each column can contain a different **data** type.

**Eg:** Array, Linked Lists, Stack, Queues, Trees, Graphs, Sets, Hash Tables.

**Data Frame:**

A **data frame** can be used to represent an entire **data** set. A data frame is a table or a two-dimensional array-like structure in which each column contains values of one variable and each row contains one set of values from each column.

**Eg:** Matrices

1. **How to create the data frame and print it for the employee data set.**

**Solution:**

**Create Dataframe:**

emp.data <- data.frame(

emp\_id = c (1:5),

emp\_name = c("Ricky","Danish","Mini","Ryan","Gary"),

salary = c(643.3,515.2,671.0,729.0,943.25),

start\_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15", "2014-05-11","2015-03-27")),

stringsAsFactors = FALSE

)

**Print the Data Frame:**

print(emp.data)

**Output:**

emp\_id emp\_name salary start\_date

1 Ricky 643.30 2012-01-01

2 Danish 515.20 2013-09-23

3 Mini 671.00 2014-11-15

4 Ryan 729.00 2014-05-11

5 Gary 943.25 2015-03-27

1. **Write the code to get the Structure of the R Data Frame.**

**Solution:**

str(emp.data)

**Output:**

'data.frame': 5 obs. of 4 variables:

$ emp\_id : int 1 2 3 4 5

$ emp\_name : chr "Ricky" "Danish" "Mini" "Ryan" ...

$ salary : num 643 515 671 729 943

$start\_date : Dat, efrmoat: "2012-01-01" "2013-09-23" "214-011-15" "214-00511-" ...

1. **How to extract data from data frame for the employee dataset.**

**Solution:**

**Create Dataframe:**

emp.data <- data.frame(

emp\_id = c (1:5),

emp\_name = c("Ricky","Danish","Mini","Ryan","Gary"),

salary = c(643.3,515.2,671.0,729.0,943.25),

start\_date = as.Date(c("2012-01-01","2013-09-23","2014-11-15","2014-05-11","2015-03-27")),

stringsAsFactors = FALSE

)

**Extract Data Frame:**

result <- data.frame(emp.data$emp\_name,emp.data$salary)

print(result)

**Output:**

emp.data.emp\_name. emp.data.salary

1 Ricky 643.30

2 Danish 515.20

3 Mini 671.00

4 Ryan 729.00

5 Gary 943.25

1. **How to extract the first two rows and then all columns in employee data frame.**

**Solution:**

**Create the Data Frame:**

emp.data <- data.frame(

emp\_id = c (1:5),

emp\_name = c("Ricky","Danish","Mini","Ryan","Gary"),

salary = c(643.3,515.2,671.0,729.0,943.25),

start\_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15", "2014-05-11","2015-03-27")),

stringsAsFactors = FALSE

)

**Extract First two rows:**

result <- emp.data[1:2,]

print(result)

**Output:**

emp\_id emp\_name salary start\_date

1 Ricky 643.3 2012-01-01

2 Danish 515.2 2013-09-23

1. **Write a code to extract 3rd and 5th row with 2nd and 4th column of the employee data.**

**Create the Data Frame:**

emp.data <- data.frame(

emp\_id = c (1:5),

emp\_name = c("Ricky","Danish","Mini","Ryan","Gary"),

salary = c(643.3,515.2,671.0,729.0,943.25),

start\_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15", "2014-05-11",

"2015-03-27")),

StringsAsFactors = FALSE

)

**Extract 3rd and 5th row with 2nd and 4th column.**

result <- emp.data[c(3,5),c(2,4)]

print(result)

**Output:**

emp\_name start\_date

3 Mini 2014-11-15

5 Gary 2015-03-27

**Data Reshaping:**

Data reshaping means changing how data is represented in rows and column. It includes splitting, merging or interchanging the rows and columns.

**Reshaping functions:**

* cbind()
* rbind()
* mergr()

1. **How to expand the data frame by adding rows and columns in data frame for employee data set.**

**Add Column:**

**Create Data Frame:**

emp.data <- data.frame(

emp\_id = c (1:5),

emp\_name = c("Ricky","Danish","Mini","Ryan","Gary"),

salary = c(643.3,515.2,671.0,729.0,943.25),

start\_date = as.Date(c("2012-01-01","2013-09-23","2014-11-15", "2014-05-11","2015-03-27")),

stringsAsFactors = FALSE

)

**Add the “dept” column**

emp.data$dept <- c("IT","Operations","IT","HR","Finance")

v <- emp.data

print(v)

**Output:**

emp\_id emp\_name salary start\_date dept

1 Ricky 643.30 2012-01-01 IT

2 Danish 515.20 2013-09-23 Operations

3 Mini 671.00 2014-11-15

4 Ryan 729.00 2014-05-11 HR

5 Gary 943.25 2015-03-27 Finance

**Add Row:**

**Create the first data frame.**

emp.data <- data.frame(

emp\_id = c (1:5),

emp\_name = c("Ricky","Danish","Mini","Ryan","Gary"),

salary = c(643.3,515.2,671.0,729.0,943.25),

start\_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15", "2014-05-11","2015-03-27")),

dept = c("IT","Operations","IT","HR","Finance"),

stringsAsFactors = FALSE

)

**Create the second R data frame**

emp.newdata <- data.frame(

emp\_id = c (6:8),

emp\_name = c("Rasmi","Pranab","Tusar"),

salary = c(578.0,722.5,632.8),

start\_date = as.Date(c("2013-05-21","2013-07-30","2014-06-17")),

dept = c("IT","Operations","Fianance"),

stringsAsFactors = FALSE

)

**Bind the two data frames.**

emp.finaldata <- rbind(emp.data,emp.newdata)

print(emp.finaldata)

**Output:**

emp\_id emp\_name salary start\_date dept

1 Ricky 643.30 2012-01-01 IT

2 Danish 515.20 2013-09-23 Operations

3 Mini 671.00 2014-11-15 IT

4 Ryan 729.00 2014-05-11 HR

5 Gary 943.25 2015-03-27 Finance

6 Rasmi 578.00 2013-05-21 IT

7 Pranab 722.50 2013-07-30 Operations

8 Tusar 632.80 2014-06-17 Fianance

1. **How to use the cbind() and rbind() in data frame for the fields city and zipcode datas using vector and data frame.**

**Create a vectors:**

City <- c(“delhi”, “bangalore”,”chennai”,”mumbai”)

Zipcode<- c(123456,789654,698748,456986)

**cbind() function:**

oldaddresses <- cbind(city,zipcode)

print(oldaddresses)

**Output:**

**city zipcode**

[1] delhi 123456

[2] bangalore 789654

[3] chennai 698748

[4] mumbai 456986

**rbind() function:**

**Create Data frame:**

newaddress <- data.frame(city<-c(“punjab”,”kerala”), zipcode<-c(“456978,569875))

print(newaddress)

**rbind()**

totaladdress<-rbind(oldaddress,newaddress)

print(totaladdress)

**Output:**

**city zipcode**

[1] delhi 123456

[2] bangalore 789654

[3] chennai 698748

[4] mumbai 456986

[5] punjab 456978

[6] kerala 569875

1. **Create First Dataset with variables**

* **surname**
* **nationality**

**Create Second Dataset with variables**

* **surname**
* **movies**

**The common key variable is surname. How to merge both data and check if the dimensionality is 7x3.**

**# Create origin dataframe(**

producers <- data.frame(

surname = c("Spielberg","Scorsese","Hitchcock","Tarantino","Polanski"),

nationality = c("US","US","UK","US","Poland"),

stringsAsFactors=FALSE)

# **Create destination dataframe**

movies <- data.frame(

surname = c("Spielberg",

"Scorsese",

"Hitchcock",

"Hitchcock",

"Spielberg",

"Tarantino",

"Polanski"),

title = c("Super 8",

"Taxi Driver",

"Psycho",

"North by Northwest",

"Catch Me If You Can",

"Reservoir Dogs","Chinatown"),

stringsAsFactors=FALSE)

**# Merge two datasets**

m1 <- merge(producers, movies, by.x = "surname")

m1

dim(m1)

**Output:**

**surname nationality title**

1 Hitchcock UK Psycho

2 Hitchcock UK North by Northwest

3 Polanski Poland Chinatown

4 Scorsese US Taxi Driver

5 Spielberg US Super 8

6 Spielberg US Catch Me If You Can

* 1. Tarantino US Reservoir Dogs

1. **Write a R program to create an empty data frame.**

df = data.frame(Ints=integer(),

Doubles=double(),

Characters=character(),

Logicals=logical(),

Factors=factor(),

stringsAsFactors=FALSE)

print("Structure of the empty dataframe:")

print(str(df))

**Output:**

[1] "Structure of the empty dataframe:"

'data.frame': 0 obs. of 5 variables:

$ Ints : int

$ Doubles : num

$ Characters: chr

$ Logicals : logi

$ Factors : Factor w/ 0 levels:

NULL

1. **Write a R program to create a data frame from four given vectors**

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas')

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19)

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1)

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

print("Original data frame:")

print(name)

print(score)

print(attempts)

print(qualify)

df = data.frame(name, score, attempts, qualify)

print(df)

**Output:**

[1] "Original data frame:"

[1] "Anastasia" "Dima" "Katherine" "James" "Emily" "Michael"

[7] "Matthew" "Laura" "Kevin" "Jonas"

[1] 12.5 9.0 16.5 12.0 9.0 20.0 14.5 13.5 8.0 19.0

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

[1] "yes" "no" "yes" "no" "no" "yes" "yes" "no" "no" "yes"

**name score attempts qualify**

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

1. **Write a R program to extract specific column from a data frame using column name.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("Extract Specific columns:")

result <- data.frame(exam\_data$name,exam\_data$score)

print(result)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "Extract Specific columns:"

exam\_data.name exam\_data.score

1 Anastasia 12.5

2 Dima 9.0

3 Katherine 16.5

4 James 12.0

5 Emily 9.0

6 Michael 20.0

7 Matthew 14.5

8 Laura 13.5

9 Kevin 8.0

10 Jonas 19.0

1. **Write a R program to extract first two rows from a given data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("Extract first two rows:")

result = exam\_data[1:2,]

print(result)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "Extract first two rows:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

1. **Write a R program to extract 3rd and 5th rows with 1st and 3rd columns from a given data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("Extract 3rd and 5th rows with 1st and 3rd columns :")

result = exam\_data[c(3,5),c(1,3)]

print(result)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "Extract 3rd and 5th rows with 1st and 3rd columns :"

name attempts

3 Katherine 2

5 Emily 2

1. **Write a R program to add a new column in a given data frame**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("New data frame after adding the 'country' column:")

exam\_data$country = c("USA","USA","USA","USA","USA","USA","USA","USA","USA","USA")

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "New data frame after adding the 'country' column:"

name score attempts qualify country

1 Anastasia 12.5 1 yes USA

2 Dima 9.0 3 no USA

3 Katherine 16.5 2 yes USA

4 James 12.0 3 no USA

5 Emily 9.0 2 no USA

6 Michael 20.0 3 yes USA

7 Matthew 14.5 1 yes USA

8 Laura 13.5 1 no USA

9 Kevin 8.0 2 no USA

10 Jonas 19.0 1 yes USA

1. **Write a R program to add new row(s) to an existing data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

new\_exam\_data = data.frame(

name = c('Robert', 'Sophia'),

score = c(10.5, 9),

attempts = c(1, 3),

qualify = c('yes', 'no')

)

exam\_data = rbind(exam\_data, new\_exam\_data)

print("After adding new row(s) to an existing data frame:")

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "After adding new row(s) to an existing data frame:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

11 Robert 10.5 1 yes

12 Sophia 9.0 3 no

1. **Write a R program to drop column(s) by name from a given data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

exam\_data = subset(exam\_data, select = -c(name, qualify))

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

score attempts

1 12.5 1

2 9.0 3

3 16.5 2

4 12.0 3

5 9.0 2

6 20.0 3

7 14.5 1

8 13.5 1

9 8.0 2

10 19.0 1

1. **Write a R program to drop row(s) by number from a given data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

exam\_data <- exam\_data[-c(2, 4, 6), ]

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

name score attempts qualify

1 Anastasia 12.5 1 yes

3 Katherine 16.5 2 yes

5 Emily 9.0 2 no

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

1. **Write a R program to sort a given data frame by multiple column(s).**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("dataframe after sorting 'name' and 'score' columns:")

exam\_data = exam\_data[with(exam\_data, order(name, score)), ]

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "dataframe after sorting 'name' and 'score' columns:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

5 Emily 9.0 2 no

4 James 12.0 3 no

10 Jonas 19.0 1 yes

3 Katherine 16.5 2 yes

9 Kevin 8.0 2 no

8 Laura 13.5 1 no

7 Matthew 14.5 1 yes

6 Michael 20.0 3 yes

1. **Write a R program to create inner, outer, left, right join(merge) from given two data frames.**

df1 = data.frame(numid = c(12, 14, 10, 11))

df2 = data.frame(numid = c(13, 15, 11, 12))

print("Left outer Join:")

result = merge(df1, df2, by = "numid", all.x = TRUE)

print(result)

print("Right outer Join:")

result = merge(df1, df2, by = "numid", all.y = TRUE)

print(result)

print("Outer Join:")

result = merge(df1, df2, by = "numid", all = TRUE)

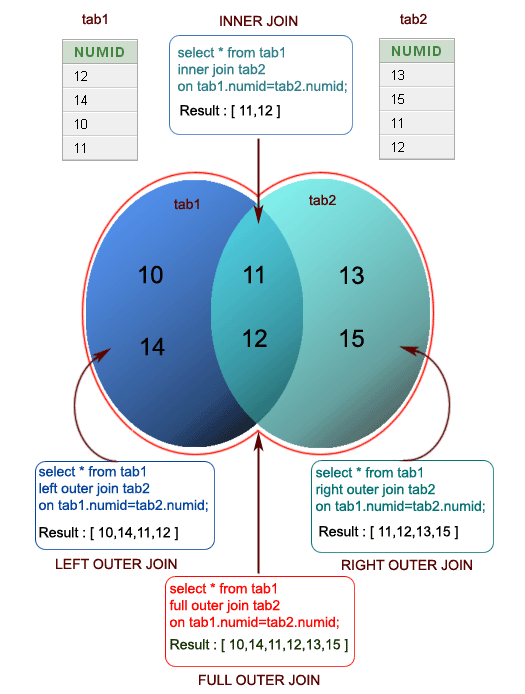
print(result)

print("Cross Join:")

result = merge(df1, df2, by = NULL)

print(result)

**Output:**



[1] "Left outer Join:"

numid

1 10

2 11

3 12

4 14

[1] "Right outer Join:"

numid

1 11

2 12

3 13

4 15

[1] "Outer Join:"

numid

1 10

2 11

3 12

4 13

5 14

6 15

[1] "Cross Join:"

numid.x numid.y

1 12 13

2 14 13

3 10 13

4 11 13

5 12 15

6 14 15

7 10 15

8 11 15

9 12 11

10 14 11

11 10 11

12 11 11

13 12 12

14 14 12

15 10 12

1. 11 12
2. **Write a R program to replace NA values with 3 in a given data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

exam\_data[is.na(exam\_data)] = 3

print("After removing NA with 3, the said dataframe becomes:")

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 NA no

3 Katherine 16.5 2 yes

4 James 12.0 NA no

5 Emily 9.0 2 no

6 Michael 20.0 NA yes

7 Matthew 14.5 1 yes

8 Laura 13.5 NA no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "After removing NA with 3, the said dataframe becomes:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 3 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

1. **Write a R program to change a column name of a given data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("Change column-name 'name' to 'student\_name' of the said dataframe:")

colnames(exam\_data)[which(names(exam\_data) == "name")] = "student\_name"

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 NA no

3 Katherine 16.5 2 yes

4 James 12.0 NA no

5 Emily 9.0 2 no

6 Michael 20.0 NA yes

7 Matthew 14.5 1 yes

8 Laura 13.5 NA no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "Change column-name 'name' to 'student\_name' of the said dataframe:"

student\_name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 NA no

3 Katherine 16.5 2 yes

4 James 12.0 NA no

5 Emily 9.0 2 no

6 Michael 20.0 NA yes

7 Matthew 14.5 1 yes

8 Laura 13.5 NA no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

1. **Write a R program to change more than one column name of a given data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("Change more than one column name of the said dataframe:")

colnames(exam\_data)[which(names(exam\_data) == "name")] = "student\_name"

colnames(exam\_data)[which(names(exam\_data) == "score")] = "avg\_score"

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 NA no

3 Katherine 16.5 2 yes

4 James 12.0 NA no

5 Emily 9.0 2 no

6 Michael 20.0 NA yes

7 Matthew 14.5 1 yes

8 Laura 13.5 NA no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "Change more than one column name of the said dataframe:"

student\_name avg\_score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 NA no

3 Katherine 16.5 2 yes

4 James 12.0 NA no

5 Emily 9.0 2 no

6 Michael 20.0 NA yes

7 Matthew 14.5 1 yes

8 Laura 13.5 NA no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

1. **Write a R program to select some random rows from a given data frame.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("Select three random rows of the said dataframe:")

print(exam\_data[sample(nrow(exam\_data), 3),])

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "Select three random rows of the said dataframe:"

name score attempts qualify

10 Jonas 19.0 1 yes

7 Matthew 14.5 1 yes

4 James 12.0 3 no

1. **Write a R program to reorder an given data frame by column name.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("Reorder by column name:")

exam\_data = exam\_data[c("name", "attempts", "score", "qualify")]

print(exam\_data)

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "Reorder by column name:"

name attempts score qualify

1 Anastasia 1 12.5 yes

2 Dima 3 9.0 no

3 Katherine 2 16.5 yes

4 James 3 12.0 no

5 Emily 2 9.0 no

6 Michael 3 20.0 yes

7 Matthew 1 14.5 yes

8 Laura 1 13.5 no

9 Kevin 2 8.0 no

10 Jonas 1 19.0 yes

1. Write a R program to compare two data frames to find the elements in first data frame that are not present in second data frame.

a = c("a", "b", "c", "d", "e")

b = c("d", "e", "f", "g")

print("Original Dataframes")

print(a)

print(b)

print("Data in first dataframe that are not present in second dataframe:")

result = setdiff(a, b)

print(result)

**Output:**

[1] "Original Dataframes"

[1] "a" "b" "c" "d" "e"

[1] "d" "e" "f" "g"

[1] "Data in first dataframe that are not present in second dataframe:"

[1] "a" "b" "c"

1. **Write a R program to find elements which are present in two given data frames.**

a = c("a", "b", "c", "d", "e")

b = c("d", "e", "f", "g")

print("Original Dataframes")

print(a)

print(b)

print("Elements which are present in both dataframe:")

result = intersect(a, b)

print(result)

**Output:**

[1] "Original Dataframes"

[1] "a" "b" "c" "d" "e"

[1] "d" "e" "f" "g"

[1] "Elements which are present in both dataframe:"

[1] "d" "e"

1. **Write a R program to find elements come only once that are common to both given data frames.**

a = c("a", "b", "c", "d", "e")

b = c("d", "e", "f", "g")

print("Original Dataframes")

print(a)

print(b)

print("Find elements come only once that are common to both given dataframes:")

result = union(a, b)

print(result)

**Output:**

[1] "Original Dataframes"

[1] "a" "b" "c" "d" "e"

[1] "d" "e" "f" "g"

[1] "Find elements come only once that are common to both given dataframes:"

[1] "a" "b" "c" "d" "e" "f" "g"

1. **Write a R program to save the information of a data frame in a file and display the information of the file.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

save(exam\_data,file="data.rda")

load("data.rda")

file.info("data.rda")

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

3 Katherine 16.5 2 yes

4 James 12.0 3 no

5 Emily 9.0 2 no

6 Michael 20.0 3 yes

7 Matthew 14.5 1 yes

8 Laura 13.5 1 no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

size isdir mode mtime ctime

data.rda 344 FALSE 644 2018-10-25 12:06:09 2018-10-25 12:06:09

atime uid gid uname grname

data.rda 2018-10-25 12:06:09 1000 1000 trinket trinket

1. **Write a R program to count the number of NA values in a data frame column.**

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam\_data)

print("The number of NA values in attempts column:")

print(sum(is.na(exam\_data$attempts)))

**Output:**

[1] "Original dataframe:"

name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 NA no

3 Katherine 16.5 2 yes

4 James 12.0 NA no

5 Emily 9.0 2 no

6 Michael 20.0 NA yes

7 Matthew 14.5 1 yes

8 Laura 13.5 NA no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "The number of NA values in attempts column:"

[1] 4

1. **Write a R program to create a data frame using two given vectors and display the duplicated elements and unique rows of the said data frame.**

a = c(10,20,10,10,40,50,20,30)

b = c(10,30,10,20,0,50,30,30)

print("Original data frame:")

ab = data.frame(a,b)

print(ab)

print("Duplicate elements of the said data frame:")

print(duplicated(ab))

print("Unique rows of the said data frame:")

print(unique(ab))

**Output:**

[1] "Original data frame:"

a b

1 10 10

2 20 30

3 10 10

4 10 20

5 40 0

6 50 50

7 20 30

8 30 30

[1] "Duplicate elements of the said data frame:"

[1] FALSE FALSE TRUE FALSE FALSE FALSE TRUE FALSE

[1] "Unique rows of the said data frame:"

a b

1 10 10

2 20 30

4 10 20

5 40 0

6 50 50

8 30 30

1. **Write a R program to call the (built-in) dataset airquality. Check whether it is a data frame or not? Order the entire data frame by the first and second column.**

data = airquality

print("Original data: Daily air quality measurements in New York, May to September 1973.")

print(class(data))

print(head(data,10))

result = data[order(data[,1]),]

print("Order the entire data frame by the first and second column:")

print(result)

**Output:**

[1] "Original data: Daily air quality measurements in New York, May to September 1973."

[1] "data.frame"

Ozone Solar.R Wind Temp Month Day

1 41 190 7.4 67 5 1

2 36 118 8.0 72 5 2

3 12 149 12.6 74 5 3

4 18 313 11.5 62 5 4

5 NA NA 14.3 56 5 5

6 28 NA 14.9 66 5 6

7 23 299 8.6 65 5 7

8 19 99 13.8 59 5 8

9 8 19 20.1 61 5 9

10 NA 194 8.6 69 5 10

[1] "Order the entire data frame by the first and second column:"

Ozone Solar.R Wind Temp Month Day

21 1 8 9.7 59 5 21

23 4 25 9.7 61 5 23

18 6 78 18.4 57 5 18

...........

119 NA 153 5.7 88 8 27

150 NA 145 13.2 77 9 27

1. **Write a R program to call the (built-in) dataset airquality. Remove the variables 'Solar.R' and 'Wind' and display the data frame.**

data = airquality

print("Original data: Daily air quality measurements in New York, May to September 1973.")

print(data)

data[,c("Solar.R")]=NULL

data[,c("Wind")]=NULL

print("data.frame after removing 'Solar.R' and 'Wind' variables:")

print(data)

**Output:**

[1] "Original data: Daily air quality measurements in New York, May to September 1973."

Ozone Solar.R Wind Temp Month Day

1 41 190 7.4 67 5 1

2 36 118 8.0 72 5 2

3 12 149 12.6 74 5 3

4 18 313 11.5 62 5 4

5 NA NA 14.3 56 5 5

.........

152 18 131 8.0 76 9 29

153 20 223 11.5 68 9 30

[1] "data.frame after removing 'Solar.R' and 'Wind' variables:"

Ozone Temp Month Day

1 41 67 5 1

2 36 72 5 2

3 12 74 5 3

4 18 62 5 4

5 NA 56 5 5

.........

152 18 76 9 29

153 20 68 9 30

**Unit 4**

4.1 Measuring the Central Tendency

4.1.1 Mean

A group of customer service surveys were sent out at random.

The scores were 90, 50, 70, 80, 70, 60, 20, 30, 80, 90, and 20.

Find the mean score

> m1 = c(90,50,70,80,70,60,20,30,80,90,20)

> m1

[1] 90 50 70 80 70 60 20 30 80 90 20

> m2 = mean(m1)

> m2

[1] 60

4.1.2 Weighted arithmetic mean

The following table of grouped data represents the weight (in pounds) of 100 computers towers. Calculate the mean weight for a computer.

Weight (pounds) Number of Computers

(3 - 5) 8

(5 - 7) 25

(7 - 9) 45

(9 - 11) 18

(11 - 13) 4

> pounds = c(4,6,8,10,12)

> comp = c(8,25,45,18,4)

> wm = weighted.mean(pounds,comp)

> wm

[1] 7.7

4.1.3 Median

Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70. What is the median?

> age = c(13,15,16,16,19,20,20,21,22,22,25,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)

> me = median(age)

> me

[1] 25

> me = quantile(age,.50)

> me

50%

25

4.1.4 Mode

A student recorded her scores on weekly math quizzes that were marked out of a

possible 10 points. Her scores were as follows:

8, 5, 8, 5, 7, 6, 7, 7, 5, 7, 5, 5, 6, 6, 9, 8, 9, 7, 9, 9, 6, 8, 6, 6, 7

What is the mode of her scores on the weekly math quizzes?

> scores = c(8, 5, 8, 5, 7, 6, 7, 7, 5, 7, 5, 5, 6, 6, 9, 8, 9, 7, 9, 9, 6, 8, 6, 6, 7)

> getmode = function(v){uniqv = unique(v)

+ uniqv[which.max(tabulate(match(v,uniqv)))]}

> result = getmode(scores)

> print(result)

[1] 7

4.1.5 Standard Deviation.

Suppose a hospital tested the age and body fat data for 18 randomly selected adults with the following result

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *age* | 23 | 23 | 27 | 27 | 39 | 41 | 47 | 49 | 50 |
| *%fat* | 9.5 | 26.5 | 7.8 | 17.8 | 31.4 | 25.9 | 27.4 | 27.2 | 31.2 |
| *age* | 52 | 54 | 54 | 56 | 57 | 58 | 58 | 60 | 61 |
| *%fat* | 34.6 | 42.5 | 28.8 | 33.4 | 30.2 | 34.1 | 32.9 | 41.2 | 35.7 |

Calculate the standard deviation of age and %fat.

> stand\_devia <- read.csv("C:/Users/saha/Google Drive/2018 - 2019 Odd Sem/Phase 3/R Programming My trials/stand\_devia.csv")

> View(stand\_devia)

> sd\_age = sd(stand\_devia$age)

> print(sd\_age)

[1] 13.21862

> sd\_fat = sd(stand\_devia$X.fat)

> print(sd\_fat)

[1] 9.254395

4.1.6 Variance

Calculate the Variance of age and %fat for the above dataset.

> var\_age = var(stand\_devia$age)

> print(var\_age)

[1] 174.732

> var\_fat = var(stand\_devia$X.fat)

> print(var\_fat)

[1] 85.64382

4.2 Measuring the Dispersion of Data

4.2.1 Quartiles (Q1 and Q3)

Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.

Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?

> age

[1] 13 15 16 16 19 20 20 21 22 22 25 25 25 25 30 33 33 35 35 35 35 36 40 45 46 52

[27] 70

> Q1 = quantile(age,.25)

> print(Q1)

25%

20.5

> Q3 = quantile(age,.75)

> print(Q3)

75%

35

4.2.2 Inter Quartile Range

For the above dataset calculate the IQR

> Range = IQR(age)

> Range

[1] 14.5

4.2.3 Skewness and kurtosis Problem 1

Gopal travels daily from his house located at santhom to his office located at OMR road by his car and he wants know how much time he spends on travel. He does record the time taken to reach the off from his home for about a week and has the following value

46.45, 34.34, 30, 56,12,44.67,43,36.45,48, 35.67, 37.23, 32.7,39.20,40.01,45.02,34.12,33.19

Help gopal to analyse the time data using skewness and kurtosis and give your interpretation.

#Install Library

> library(e1071)

# Load the dataset

> time = c(46.45, 34.34, 30, 56,12,44.67,43,36.45,48, 35.67, 37.23, 32.7,39.20,40.01,45.02,34.12,33.19)

> skewness(time)

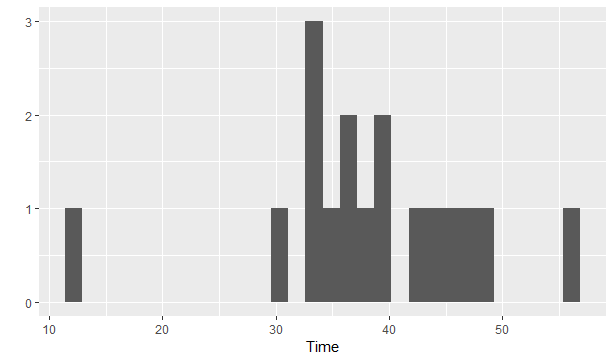
[1] -0.7371395

> kurtosis(time)

[1] 1.262199

The skewness of the simulated data is -0.7371395. This concludes that the data is close to bell shape but slightly skewed to the left.

> qplot(time, geom = 'histogram', bandwidth = 3) +xlab('Time')



4.2.4 Skewness and kurtosis Problem 2

Generate a sample of 5000 random numbers and create a normal distribution with a mean value of 70 and respectively fix the Standard deviation to 7.2. Calculate the skewness of the normal distribution along with kurtosis and interpret your results.

#Install Library

> library(e1071)

# Load the dataset

> rand.sample = rnorm(n = 5000, mean = 70 , sd = 7.2)

> rand.sample

[1] 66.63769 63.25416 65.07659 64.93925 66.31599 61.54850 70.82306 64.03036

[9] 63.02065 65.64421 86.16290 70.83292 78.04587 64.98296 84.27603 68.15348

> skewness(rand.sample)

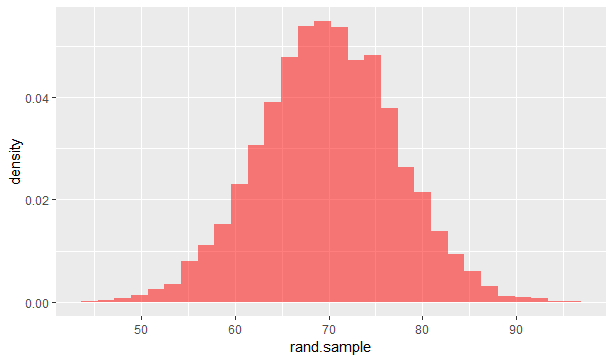
[1] -0.02082295

> kurtosis(rand.sample)

[1] 0.05415064

> datasim <- data.frame(rand.sample)

> ggplot(datasim, aes(x = rand.sample), binwidth = 2)+ geom\_histogram(aes(y = ..density..), fill = 'red', alpha = 0.5)



4.3 Crosstab

Create a data fame for the given contingency table

S.No Age Sex Country Health

1 16-29 Male Scotland Good

2 65+ Female Wales Average

3 0-15 Male Wales Poor

4 16-29 Male N. Ireland Average

5 30-44 Female Wales Good

6 30-44 Female Wales Average

Load the dataset

> s.no <- seq(1:177)

> age = sample(c("16-29","65+","0-15","30-44"),177,replace = TRUE)

> sex = sample(c("Male","Female"),177,replace = TRUE)

> country = sample(c("Scotland","Wales","N. Ireland "),177,replace = TRUE)

> Health = sample(c("Good","Average","Poor"),177,replace = TRUE)

Create a data frame

> Survey <- data.frame(age, sex, country, Health)

> head(Survey)

age sex country Health

1 0-15 Male Wales Poor

2 16-29 Female Scotland Poor

3 30-44 Male Scotland Good

4 65+ Female Wales Good

5 30-44 Male Scotland Good

6 30-44 Female Scotland Average

Load the function for crosstab

> source("http://pcwww.liv.ac.uk/~william/R/crosstab.r")

Perform the following Queries

1. Display the frequency, or count, of the levels of categorical variables (Count the levels in Sex, Country and Health)

> crosstab(Survey,col.vars = "sex")

sex Count

Female 83

Male 94

Sum 177

> crosstab(Survey,col.vars = "country”)

country Count

N. Ireland 46

Scotland 69

Wales 62

Sum 177

> crosstab(Survey,col.vars = "Health”)

Health Count

Average 55

Good 63

Poor 59

Sum 177

2. Discover the relationships within a dataset between country and health

> crosstab(Survey, row.vars = "country", col.vars = "Health")

Health Average Good Poor Sum

country

N. Ireland 11 17 18 46

Scotland 19 23 27 69

Wales 25 23 14 62

Sum 55 63 59 177

3. Create a Multi-Dimensional Table

> crosstab(Survey, row.vars = c("age","sex"), col.vars = c("Health"))

Health Average Good Poor Sum

age sex

0-15 Female 6 6 6 18

Male 12 12 5 29

Sum 18 18 11 47

16-29 Female 8 5 9 22

Male 7 7 7 21

Sum 15 12 16 43

4. Calculate the Row percentage for variable Age by health

> crosstab(Survey, row.vars = "age", col.vars = "Health")

Health Average Good Poor Sum

age

0-15 18 18 11 47

16-29 15 12 16 43

30-44 12 21 7 40

65+ 10 12 25 47

Sum 55 63 59 177

4.4 Covariance and correlation

Children of three ages are asked to indicate their preference for three photographs of adults. Do the data suggest that there is a significant relationship between age and photograph preference? What is wrong with this study?

**Photograph:**

**Age of child** A B C

5-6 years: 18 22 20

7-8 years: 2 28 40

9-10 years: 20 10 40

1. Use cov() to calculate the sample covariance between B and C.
2. Use another call to cov() to calculate the sample covariance matrix for the preferences.
3. Use cor() to calculate the sample correlation between B and C.
4. Use another call to cor() to calculate the sample correlation matrix for the preferences.

Load the dataset

> age1 = c("5-6","7-8","9-10")

> A = c(18,2,20)

> B = c(22,28,10)

> C = c(20,40,40)

> photo = data.frame(age1,A,B,C)

> photo

age1 A B C

1 5-6 18 22 20

2 7-8 2 28 40

3 9-10 20 10 40

1. Use cov() to calculate the sample covariance between B and C.

> cov(photo$B,photo$C)

[1] -20

1. Use another call to cov() to calculate the sample covariance matrix for the preferences.

> photo1 = data.frame(A,B,C)

> cov(photo1)

A B C

A 97.33333 -74 -46.66667

B -74.00000 84 -20.00000

C -46.66667 -20 133.33333

1. Use cor() to calculate the sample correlation between B and C.

> cor(photo$B,photo$C)

[1] -0.1889822

Has a negative correlation

1. Use another call to cor() to calculate the sample correlation matrix for the preferences.

> cor(photo1)

A B C

A 1.0000000 -0.8183918 -0.4096440

B -0.8183918 1.0000000 -0.1889822

C -0.4096440 -0.1889822 1.0000000

**Unit 5**

**Boxplot:**

**5.1.Create a Boxplot graph for the relation between "mpg"(miles per galloon) and "cyl"(number of Cylinders) for the dataset "mtcars" available in R Environment.**

input <- mtcars[,c('mpg','cyl')]

print(head(input))

executing above code, it produces following result −

mpg cyl

Mazda RX4 21.0 6

Mazda RX4 Wag 21.0 6

Datsun 710 22.8 4

Hornet 4 Drive 21.4 6

Hornet Sportabout 18.7 8

Valiant 18.1 6

# Give the chart file a name.

png(file = "boxplot.png")

# Plot the chart.

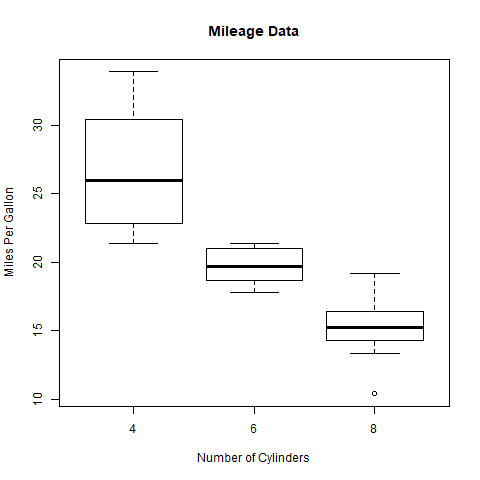
boxplot(mpg ~ cyl, data = mtcars, xlab = "Number of Cylinders",

ylab = "Miles Per Gallon", main = "Mileage Data")

# Save the file.

dev.off()

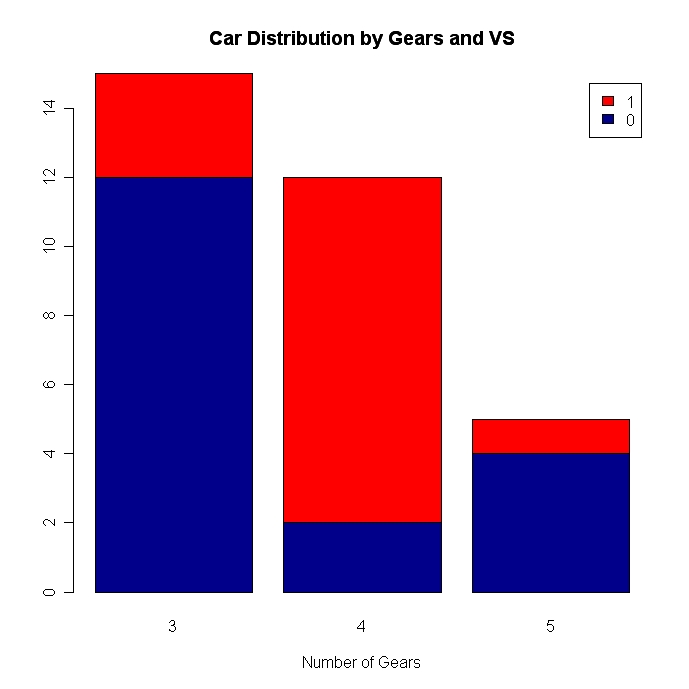
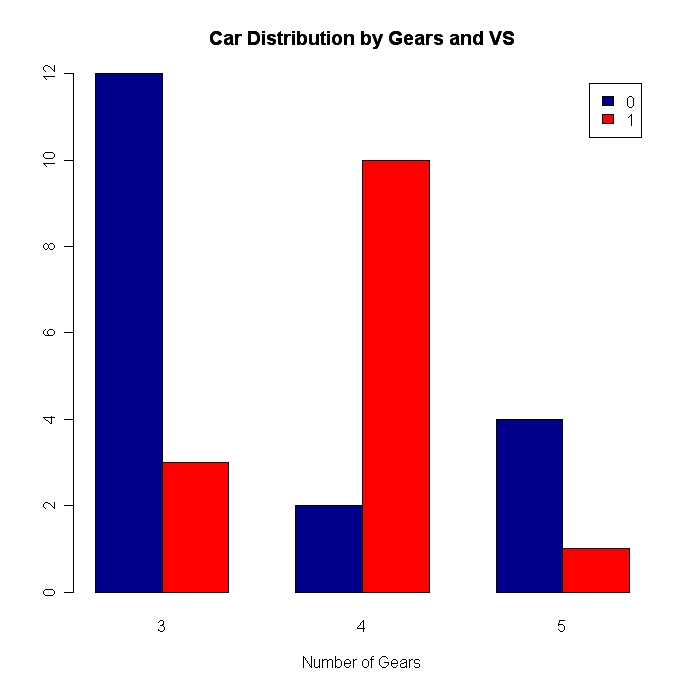
**OUTPUT:**



5.2 .Write a R code for generating Stacked Barplot and Grouped Barplot . Use Dataset “mtcars”and generate barplots shown below for the column “gear” and “vs” with the colour and legends.

***Expected Result:***

**STACKED BARPLOT**  **GROUPED BARPLOT**

Answer:

Stacked BarPlot:

# Stacked Bar Plot with Colors and Legend

counts <- table(mtcars$vs, mtcars$gear)

barplot(counts, main="Car Distribution by Gears and VS",

xlab="Number of Gears", col=c("darkblue","red"),

legend = rownames(counts))

Grouped Barplot:

# Grouped Bar Plot

counts <- table(mtcars$vs, mtcars$gear)

barplot(counts, main="Car Distribution by Gears and VS",

xlab="Number of Gears", col=c("darkblue","red"),

legend = rownames(counts), beside=TRUE

**HISTOGRAM:**

**5.3.** Make a histogram for the “AirPassengers “dataset, start at 100 on the x-axis, and from values 200 to 700, make the bins 150 wide

**Answer:**

png(file="HISTOGRAM1.png")

> hist(AirPassengers$value,

+ main="Histogram for Air Passengers",

+ xlab="value",

+ border="blue",

+ col="green",

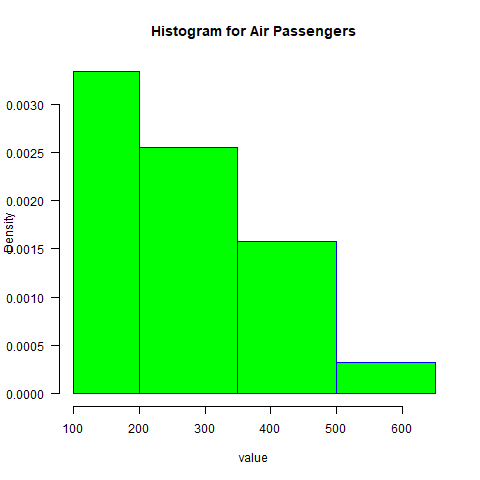
+ las=1,

+ breaks=c(100,seq(200,700,150)))

> dev.off()

RStudioGD

2



**PIECHART:**

5.4 .Create a 3D Pie Chart for the dataset “political Knowledge” with suitable labels and colours.

Answer:

# Get the library.

> library(plotrix)

> # Create data for the graph.

> f=politicalKnowledge$PoliticalKnowledge.hs

>

> # Give the chart file a name.

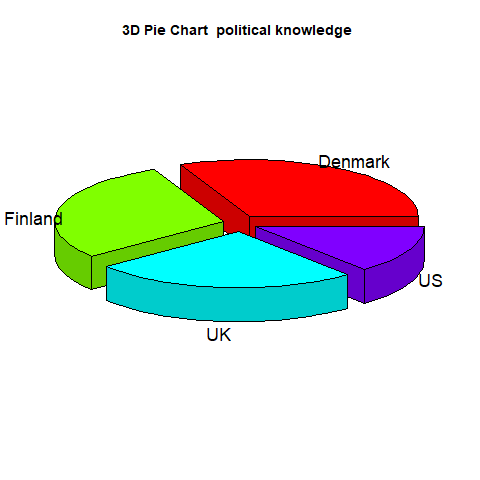
> png(file = "3d\_pie\_chart\_politicalknowledge.jpg")

> # Plot the chart.

> pie3D(f,labels=politicalKnowledge$country,col=rainbow(length(f)),explode = 0.1, main = "3D Pie Chart political knowledge ")

> # Save the file.

> dev.off()



**LINE CHART:**

5.5 .Obtain Multiple Lines in Line Chart using a single Plot Function in R.Use attributes“mpg”and“qsec”of the dataset “mtcars”

**# Create the data for the chart.**

> p=mtcars$mpg

> u=mtcars$qsec

**> # Give the chart file a name.**

> png(file = "line\_chart\_mtcars.png")

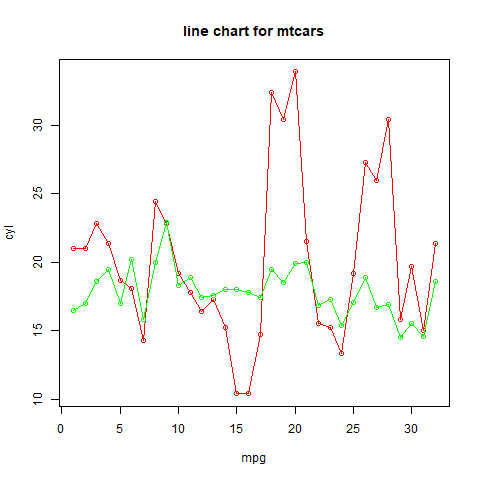
**> # Plot the line chart.**

> plot(p,type = "o",col = "red", xlab = "mpg", ylab = "cyl", main = "line chart for mtcars")

> lines(u, type = "o", col = "green")

**> # Save the file.**

> dev.off()



5.6.Download the Dataset "water" From R dataset Link.Find out whether there is a linear relation between attributes"mortality" and"hardness" by plot function.Fit the Data into the Linear Regression model.Predict the mortality for the hardness=88.

**Answer:**

**Loading the dataset:**

>u=water$mortality

> v=water$hardness

**Establishing Linear relation:**

> relation=lm(u~v)

> print(relation)

Call:

lm(formula = u ~ v)

Coefficients:

(Intercept) v

1676.356 -3.226

**Predict the moratality for hardness=54:**

> w=data.frame(v=54)

> res=predict(relation,w)

> print(res)

1

1502.147

**Linear Regression Plot:**

# Give the chart file a name.

> png(file = "linearregression1.png")

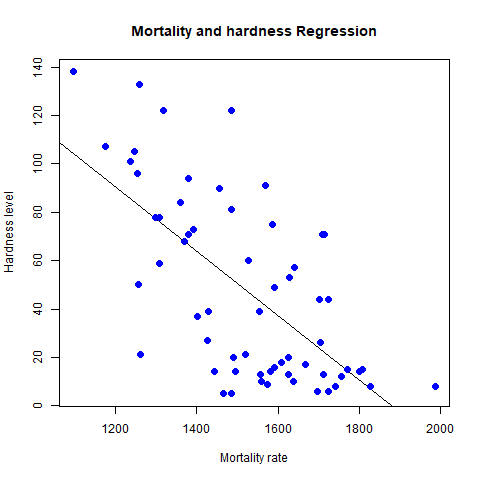
> # Plot the chart.

> plot(u,v,col = "blue",main = "Mortality and hardness Regression",

+ abline(lm(v~u)),cex = 1.3,pch = 16,xlab = "Mortality rate",ylab = "Hardness level")

> # Save the file.

> dev.off()



5.7 Consider the data set "delivery" available in the R environment. It gives a deliverytime (“delTime”)of production materials(number of productions “n.prod”) with the given distance(“distance”) to reach the destination place.

a)Create the model to establish the relationship between "delTime" as a response variable with "n.prod" and "distance" as predictor variables.

b)Predict the delTime for the given number of production(“n.prod”)=9 and distance(“distance”)=450

**Solution:**

**Loadingthe dataset:**

> delivery <- read.csv("C:/Users/CHARTHIK/Downloads/delivery.csv")

> View(delivery)

> input=delivery[, c("n.prod","distance","delTime")]

> print(head(input))

n.prod distance delTime

1 7 560 16.68

2 3 220 11.50

3 3 340 12.03

4 4 80 14.88

5 6 150 13.75

6 7 330 18.11

**Establishing the model:**

> model <- lm(delTime~n.prod+distance, data = input)

>

> # Show the model.

> print(model)

Call:

lm(formula = delTime ~ n.prod + distance, data = input)

Coefficients:

(Intercept) n.prod distance

2.34123 1.61591 0.01438

**Get the Intercept and coefficients as vector elements:**

cat("# # # # The Coefficient Values # # # ","\n")

a <- coef(model)[1]

print(a)

(Intercept)

2.341231

Xn.prod <- coef(model)[2]

Xdistance <- coef(model)[3]

print(Xn.prod)

n.prod

1.615907

print(Xdistance)

distance

0.01438483

**Create Equation for Regression Model**

Based on the above intercept and coefficient values, we create the mathematical equation.

finaldelivery = a+Xn.prod.x1+Xdistance

or

**> finaldelivery= 2.341231+(1.615907)\*x1+(0.01438483)\*x2**

**Apply Equation for predicting New Values**

The regression equation created above to predict the delivertime when a new set of values for n.prod and distance is provided.

For n.prod = 9,distance = 450 the predicted deliverytime is –

> x1=9

> x2=450

> finaldelivery= 2.341231+(1.615907)\*x1+(0.01438483)\*x2

> print(finaldelivery)

[1] 23.35757