18-PST-044

17/7/18

IMPORTING DATA INTO R PLATFORM

READING TEXT FILES IN R

EXERCISE 9:

AIM: To learn how to read a text file in R.

R CODE AND OUTPUT:

```
> getwd()
[1] "U:/A Demo/Data"
> firsttext=read.table("table1.txt",header=T,sep="\t")
> firsttext
     Name Gender Age Income Qualification
1 Malathi Female  20    100          M.Sc
2 Chellam   Male  20    200          M.Sc
3  Dhusan   Male  21   1000          M.Sc
4   Ashok   Male  25   2000          M.Sc
5  Harini Female  20   3000          M.Sc
```

INTERPRETATION

Used getwd() code for getting the file directory in which the text file is stored and used the code read.table to read the text file. Sep="\t" is used as the input values are separated by a tab and since headings are present we gave header=T .

18-PST-044

17/7/18

READING CSV FILE IN R

EXERCISE 10:

AIM: To learn how to read a csv file in R

R CODE AND OUTPUT

```
> mycsv=read.table("beauty.csv",header=T,sep=",")
> mycsv
   Participants Judge1 Judge2 Judge3
1             1     10      8      5
2             2      2      1      9
3             3      8      9      3
4             4      6      5      1
5             5      5      6      8
6             6      4      4      6
7             7      7      7      2
8             8      3      2      4
9             9      1      3     10
10           10      9     10      7
> 
```

INTERPRETATION

CSV file stands for comma separated file. We used read.table command to read csv file named beauty in csv format. Since headings are present we used header=T . As this a csv data we used sep=",".

18-PST-044

17/7/18

READING EXCEL FILE IN R

EXERCISE 11:

AIM: To learn how to read an excel file in R.

R CODE AND OUTPUT:

```
> install.packages("readxl")
Installing package into 'C:/Users/pgstf03/Documents/R/win-library/3.5'
(as 'lib' is unspecified)
--- Please select a CRAN mirror for use in this session ---
also installing the dependencies 'assertthat', 'fansi', 'utf8', 'rematch', 'cli

trying URL 'https://mirrors.eliteu.cn/CRAN/bin/windows/contrib/3.5/assertthat_0
Content type 'application/zip' length 53821 bytes (52 KB)
downloaded 52 KB

trying URL 'https://mirrors.eliteu.cn/CRAN/bin/windows/contrib/3.5/fansi_0.2.3.
Content type 'application/zip' length 168098 bytes (164 KB)
downloaded 164 KB

trying URL 'https://mirrors.eliteu.cn/CRAN/bin/windows/contrib/3.5/utf8_1.1.4.z
```

```
> library(readxl)
Warning message:
package 'readxl' was built under R version 3.5.1
```

```
> file_excel <- read_excel("simple_regression.xlsx", sheet = 1)
> file_excel
# A tibble: 11 x 2
   `Hours spent` marks
           <dbl> <dbl>
 1            45    40
 2            30    35
 3            90    75
 4            60    65
 5           105    90
 6            65    50
 7            90    90
 8            80    80
 9            55    45
10            75    65
11            10    98
```

```
> file_excel=data.frame(file_excel)
> file_excel
   Hours.spent marks
1           45    40
2           30    35
3           90    75
4           60    65
5          105    90
6           65    50
7           90    90
8           80    80
9           55    45
10          75    65
11          10    98
```

INTERPRETATION

Here we installed a package named readxl using the command install.packages for reading excel file in R. After the installation of that package we used read_excel command(provided by the package) to read the excel file named simple_regression . The code data.frame is used to input the values of this excel file into R.

18-PST-044

21/7/18

RBIND AND CBIND IN R

EXERCISE 12:

AIM: To learn the functions of the codes rbind() and cbind() in R.

R CODE AND OUTPUT:

```
> a=c(1,4,7,2,4)
> b=c(6,7,8,3,9)
> c=cbind(a,b)
> c
     a b
[1,] 1 6
[2,] 4 7
[3,] 7 8
[4,] 2 3
[5,] 4 9

> e=rbind(c,c(-1,-4))
> e
      a  b
[1,]  1  6
[2,]  4  7
[3,]  7  8
[4,]  2  3
[5,]  4  9
[6,] -1 -4
>
```

INTERPRETATION

First we assigned five numerical values to variable a and b. The code cbind() gives the values of a and b in the form of columns. Variable e is assigned with two numerical values -1 and -4. The code rbind() is used with variable e and it had given the value of e in the form of a row.

18-PST-044

21/7/18

SORTING DATASETS IN ASCENDING AND DESCENDING

EXERCISE 14:

AIM: To sort data in ascending and descending order.

R CODE AND OUTPUT:

```
> data=read.csv(file.choose(),header=T,sep=",")
> df=data.frame(data)
> df
   S.NO Sex Age Smo Dia Cho  BMI Hyp EF Sur
1    51   1  56   0   0   1 27.0   0 35   1
2    52   1  51   0   0   1 26.0   0 30   1
3    53   1  55   0   1   1 26.0   1 32   1
4    54   1  46   0   1   0 28.6   0 36   0
5    55   1  55   1   1   0 26.0   0 35   0
6    56   0  54   0   1   1 30.0   1 32   1
7    57   1  48   1   1   1 24.0   1 35   0
8    58   1  51   1   1   1 26.0   1 22   1
9    59   1  51   1   1   1 26.2   1 27   1
10   60   1  56   1   1   1 28.0   1 25   1
11   61   1  59   1   1   1 28.2   1 20   0
12   62   0  46   0   0   1 26.0   0 68   0
13   63   1  46   0   0   0 22.0   0 70   0
14   64   1  55   0   1   0 23.0   1 66   0
15   65   1  44   1   1   1 27.0   1 70   0
16   66   1  60   0   1   0 25.0   1 68   0
17   67   1  46   0   1   0 26.0   1 68   1
18   68   1  42   0   0   0 28.0   0 76   0
19   69   1  78   1   1   1 34.0   1 22   1
20   70   1  56   0   0   0 29.0   0 68   1
21   71   1  60   1   1   0 25.0   1 68   0
22   72   1  56   0   0   1 26.0   0 70   0
23   73   1  54   0   0   0 25.0   0 68   0
```

```
> df_ascending = df[order(df$Smo),]
> df_ascending
   S.NO Sex Age Smo Dia Cho  BMI Hyp EF Sur
1    51   1  56   0   0   1 27.0   0 35   1
2    52   1  51   0   0   1 26.0   0 30   1
3    53   1  55   0   1   1 26.0   1 32   1
4    54   1  46   0   1   0 28.6   0 36   0
6    56   0  54   0   1   1 30.0   1 32   1
12   62   0  46   0   0   1 26.0   0 68   0
13   63   1  46   0   0   0 22.0   0 70   0
14   64   1  55   0   1   0 23.0   1 66   0
16   66   1  60   0   1   0 25.0   1 68   0
17   67   1  46   0   1   0 26.0   1 68   1
18   68   1  42   0   0   0 28.0   0 76   0
20   70   1  56   0   0   0 29.0   0 68   1
22   72   1  56   0   0   1 26.0   0 70   0
23   73   1  54   0   0   0 25.0   0 68   0
25   75   1  52   0   1   0 22.0   1 72   0
26   76   1  40   0   0   1 26.0   0 72   0
27   77   0  60   0   1   0 30.0   1 74   0
28   78   1  55   0   0   0 24.0   1 72   0
29   79   0  59   0   1   0 28.0   1 72   0
31   81   1  54   0   0   0 29.0   0 70   0
38   88   0  54   0   1   1 36.0   1 32   1
40   90   1  40   0   0   1 27.4   0 70   0
41   91   0  60   0   1   0 31.6   1 68   0
42   92   1  55   0   0   1 26.8   1 72   0
43   93   0  56   0   1   0 26.8   1 72   0
45   95   0  56   0   0   0 28.8   0 68   0
5    55   1  55   1   1   0 26.0   0 35   0
7    57   1  48   1   1   1 24.0   1 35   0
8    58   1  51   1   1   1 26.0   1 22   1
9    59   1  51   1   1   1 26.2   1 27   1
```

```
> data=read.csv(file.choose(),header=T,sep=",")
> df=data.frame(data)
> df
   S.NO Sex Age Smo Dia Cho  BMI Hyp EF Sur
1    51   1  56   0   0   1 27.0   0 35   1
2    52   1  51   0   0   1 26.0   0 30   1
3    53   1  55   0   1   1 26.0   1 32   1
4    54   1  46   0   1   0 28.6   0 36   0
5    55   1  55   1   1   0 26.0   0 35   0
6    56   0  54   0   1   1 30.0   1 32   1
7    57   1  48   1   1   1 24.0   1 35   0
8    58   1  51   1   1   1 26.0   1 22   1
9    59   1  51   1   1   1 26.2   1 27   1
10   60   1  56   1   1   1 28.0   1 25   1
11   61   1  59   1   1   1 28.2   1 20   0
12   62   0  46   0   0   1 26.0   0 68   0
13   63   1  46   0   0   0 22.0   0 70   0
14   64   1  55   0   1   0 23.0   1 66   0
15   65   1  44   1   1   1 27.0   1 70   0
16   66   1  60   0   1   0 25.0   1 68   0
17   67   1  46   0   1   0 26.0   1 68   1
18   68   1  42   0   0   0 28.0   0 76   0
19   69   1  78   1   1   1 34.0   1 22   1
20   70   1  56   0   0   0 29.0   0 68   1
21   71   1  60   1   1   0 25.0   1 68   0
22   72   1  56   0   0   1 26.0   0 70   0
23   73   1  54   0   0   0 25.0   0 68   0
24   74   1  55   1   0   0 25.0   0 70   0
25   75   1  52   0   1   0 22.0   1 72   0
26   76   1  40   0   0   1 26.0   0 72   0
27   77   0  60   0   1   0 30.0   1 74   0
28   78   1  55   0   0   0 24.0   1 72   0
```

```
> df_descending = df[order(-df$EF),]
> df_descending
   S.NO Sex Age Smo Dia Cho  BMI Hyp EF Sur
18   68   1  42   0   0   0 28.0   0 76   0
46   96   1  62   1   1   0 28.6   1 76   0
27   77   0  60   0   1   0 30.0   1 74   0
25   75   1  52   0   1   0 22.0   1 72   0
26   76   1  40   0   0   1 26.0   0 72   0
28   78   1  55   0   0   0 24.0   1 72   0
29   79   0  59   0   1   0 28.0   1 72   0
42   92   1  55   0   0   1 26.8   1 72   0
43   93   0  56   0   1   0 26.8   1 72   0
47   97   1  56   1   1   0 28.8   0 72   0
50  100   1  54   1   0   1 25.8   0 72   0
13   63   1  46   0   0   0 22.0   0 70   0
15   65   1  44   1   1   1 27.0   1 70   0
22   72   1  56   0   0   1 26.0   0 70   0
24   74   1  55   1   0   0 25.0   0 70   0
30   80   1  57   1   1   1 30.0   1 70   0
31   81   1  54   0   0   0 29.0   0 70   0
32   82   1  61   1   1   0 26.0   1 70   0
40   90   1  40   0   0   1 27.4   0 70   0
48   98   1  53   1   0   1 26.2   0 69   0
12   62   0  46   0   0   1 26.0   0 68   0
16   66   1  60   0   1   0 25.0   1 68   0
17   67   1  46   0   1   0 26.0   1 68   1
20   70   1  56   0   0   0 29.0   0 68   1
21   71   1  60   1   1   0 25.0   1 68   0
23   73   1  54   0   0   0 25.0   0 68   0
41   91   0  60   0   1   0 31.6   1 68   0
```

INTERPRETATION

At first a csv file is imported and the values under the category smo has been sorted into ascending and values under category EF into descending order. df$smo is used to sort values in ascending order and –df$EF is used to sort values in descending order. “$” is used with the variables to indicate that the variables are new or the values are assigned in different order.

18-PST-044

23/7/18

TRANSFORMING VARIABLES

EXERCISE 15:

AIM: To learn to transform a given variable using the code recode.

R CODE AND OUTPUT:

```
> setwd("U:\\A Demo\\Data")
> data_select=read.table("table2.CSV",header=T,sep=",")
> head(data_select,10)
   S.NO Sex Age Smo Dia Cho  BMI Hyp EF Sur
1    51   1  56   0   0   1 27.0   0 35   1
2    52   1  51   0   0   1 26.0   0 30   1
3    53   1  55   0   1   1 26.0   1 32   1
4    54   1  46   0   1   0 28.6   0 36   0
5    55   1  55   1   1   0 26.0   0 50   0
6    56   0  54   0   1   1 30.0   1 32   1
7    57   1  48   1   1   1 24.0   1 53   0
8    58   1  51   1   1   1 26.0   1 22   1
9    59   1  51   1   1   1 26.2   1 45   1
10   60   1  56   1   1   1 28.0   1 25   1
> tail(data_select,10)
   S.NO Sex Age Smo Dia Cho BMI Hyp EF Sur
15   65   1  44   1   1   1  27   1 70   0
16   66   1  60   0   1   0  25   1 68   0
17   67   1  46   0   1   0  26   1 68   1
18   68   1  42   0   0   0  28   0 76   0
19   69   1  78   1   1   1  34   1 22   1
20   70   1  56   0   0   0  29   0 68   1
21   71   1  60   1   1   0  25   1 68   0
22   72   1  56   0   0   1  26   0 70   0
23   73   1  54   0   0   0  25   0 68   0
24   74   1  55   1   0   0  25   0 70   0
```



```
> d_recode$Sex_val=ifelse(d_recode$Sex == 1,"Male","Female")

> d_recode$Smo_val=ifelse(d_recode$Smo == 0,"No","Yes")

> d_recode$Dia_val=ifelse(d_recode$Dia == 1,"Diabetic","Non-Diabetic")

> d_recode$Cho_N=ifelse(d_recode$Cho == 1,"Fit","Notfit")

> d_recode$Hyp_N=ifelse(d_recode$Hyp == 1,"Tensed","Fine")

> d_recode$Sur_N=ifelse(d_recode$Sur == 1,"Alive","Dead")

> d_recode$EF_N=ifelse(d_recode$EF < 40,"Risk",ifelse(d_recode$EF >= 40 & d_recode$EF < 55,"Moderate","Normal"))
> d_recode
```

	S.NO	Sex	Age	Smo	Dia	Cho	BMI	Hyp	EF	Sur	Sex_val	Smo_val	Dia_val	Cho_N	Hyp_N	Sur_N	EF_N
1	51	1	56	0	0	1	27.0	0	35	1	Male	No	Non-Diabetic	Fit	Fine	Alive	Risk
2	52	1	51	0	0	1	26.0	0	30	1	Male	No	Non-Diabetic	Fit	Fine	Alive	Risk
3	53	1	55	0	1	1	26.0	1	32	1	Male	No	Diabetic	Fit	Tensed	Alive	Risk
4	54	1	46	0	1	0	28.6	0	36	0	Male	No	Diabetic	Notfit	Fine	Dead	Risk
5	55	1	55	1	1	0	26.0	0	50	0	Male	Yes	Diabetic	Notfit	Fine	Dead	Moderate
6	56	0	54	0	1	1	30.0	1	32	1	Female	No	Diabetic	Fit	Tensed	Alive	Risk
7	57	1	48	1	1	1	24.0	1	53	0	Male	Yes	Diabetic	Fit	Tensed	Dead	Moderate
8	58	1	51	1	1	1	26.0	1	22	1	Male	Yes	Diabetic	Fit	Tensed	Alive	Risk
9	59	1	51	1	1	1	26.2	1	45	1	Male	Yes	Diabetic	Fit	Tensed	Alive	Moderate
10	60	1	56	1	1	1	28.0	1	25	1	Male	Yes	Diabetic	Fit	Tensed	Alive	Risk
11	61	1	59	1	1	1	28.2	1	20	0	Male	Yes	Diabetic	Fit	Tensed	Dead	Risk
12	62	0	46	0	0	1	26.0	0	68	0	Female	No	Non-Diabetic	Fit	Fine	Dead	Normal
13	63	1	46	0	0	0	22.0	0	49	0	Male	No	Non-Diabetic	Notfit	Fine	Dead	Moderate
14	64	1	55	0	1	0	23.0	1	66	0	Male	No	Diabetic	Notfit	Tensed	Dead	Normal
15	65	1	44	1	1	1	27.0	1	70	0	Male	Yes	Diabetic	Fit	Tensed	Dead	Normal
16	66	1	60	0	1	0	25.0	1	68	0	Male	No	Diabetic	Notfit	Tensed	Dead	Normal
17	67	1	46	0	1	0	26.0	1	68	1	Male	No	Diabetic	Notfit	Tensed	Alive	Normal
18	68	1	42	0	0	0	28.0	0	76	0	Male	No	Non-Diabetic	Notfit	Fine	Dead	Normal
19	69	1	78	1	1	1	34.0	1	22	1	Male	Yes	Diabetic	Fit	Tensed	Alive	Risk
20	70	1	56	0	0	0	29.0	0	68	1	Male	No	Non-Diabetic	Notfit	Fine	Alive	Normal
21	71	1	60	1	1	0	25.0	1	68	0	Male	Yes	Diabetic	Notfit	Tensed	Dead	Normal
22	72	1	56	0	0	1	26.0	0	70	0	Male	No	Non-Diabetic	Fit	Fine	Dead	Normal
23	73	1	54	0	0	0	25.0	0	68	0	Male	No	Non-Diabetic	Notfit	Fine	Dead	Normal
24	74	1	55	1	0	0	25.0	0	70	0	Male	Yes	Non-Diabetic	Notfit	Fine	Dead	Normal

INTERPRETATION

A csv file is read using read.table. The variables Sex, Smo, Dia, Cho, Hyp, sur, and EF are transformed into Sex_val, Smo_val, Dia_val, Cho_N, Hyp_N, Sur_N and EF_N respectively using the code recode$.

18-PST-044

23/7/18

CREATING NEW VARIABLE USING MATHEMATICAL OPERATOR

EXERCISE 16:

AIM: To create new variables using mathematical operators.

R CODE AND OUTPUT:

```
> data=read.table(file.choose(),header=T,sep=",")
> data
     Name Age Gender Weight Height_cm
1    Emma  21   Male     49       160
2    Tona  30   Male     70       158
3     Saj  34   Male     50       156
4  Reshma  33 Female     57       162
5    Ammu  21 Female     51       154
6     Anu  20 Female     55       160
7   Rufus  12   Male     40       130
8    Akku  16   Male     46       140
9   Ponnu  28 Female     60       145
10  Susan  42 Female     65       162
```

```
> data$height_m=data$Height_cm/100
> data
     Name Age Gender Weight Height_cm height_m
1    Emma  21   Male     49       160     1.60
2    Tona  30   Male     70       158     1.58
3     Saj  34   Male     50       156     1.56
4  Reshma  33 Female     57       162     1.62
5    Ammu  21 Female     51       154     1.54
6     Anu  20 Female     55       160     1.60
7   Rufus  12   Male     40       130     1.30
8    Akku  16   Male     46       140     1.40
9   Ponnu  28 Female     60       145     1.45
10  Susan  42 Female     65       162     1.62
>
```

```
> data$bmi=(data$Weight)/(data$height_m^2)
> data
     Name Age Gender Weight Height_cm height_m      bmi
1    Emma  21   Male     49       160     1.60 19.14062
2    Tona  30   Male     70       158     1.58 28.04038
3     Saj  34   Male     50       156     1.56 20.54569
4  Reshma  33 Female     57       162     1.62 21.71925
5    Ammu  21 Female     51       154     1.54 21.50447
6     Anu  20 Female     55       160     1.60 21.48437
7   Rufus  12   Male     40       130     1.30 23.66864
8    Akku  16   Male     46       140     1.40 23.46939
9   Ponnu  28 Female     60       145     1.45 28.53746
10  Susan  42 Female     65       162     1.62 24.76757
> data$bmi_cat=ifelse(data$bmi<18.5,"under weight",
+ ifelse(data$bmi>=18.5 & data$bmi<25,"normal weight",
+ ifelse(data$bmi>=25 & data$bmi<30,"over weight","obese")))
> data
     Name Age Gender Weight Height_cm height_m      bmi       bmi_cat
1    Emma  21   Male     49       160     1.60 19.14062 normal weight
2    Tona  30   Male     70       158     1.58 28.04038   over weight
3     Saj  34   Male     50       156     1.56 20.54569 normal weight
4  Reshma  33 Female     57       162     1.62 21.71925 normal weight
5    Ammu  21 Female     51       154     1.54 21.50447 normal weight
6     Anu  20 Female     55       160     1.60 21.48437 normal weight
7   Rufus  12   Male     40       130     1.30 23.66864 normal weight
8    Akku  16   Male     46       140     1.40 23.46939 normal weight
9   Ponnu  28 Female     60       145     1.45 28.53746   over weight
10  Susan  42 Female     65       162     1.62 24.76757 normal weight
```

INTERPRETATION

The data is read using file.choose() . New variables height_cm,height_m and bmi_cat are created using mathematical operations on the variables height and bmi. If else command to used to categorise bmi into normal weight,over weight and obese under the variable bmi_cat.

18-PST-044

23/7/18

HANDLING MISSING DATA

EXERCISE 17:

AIM: To learn how to handle datasets with missing values.

R CODE AND OUTPUT:

```
> B <- c(33, 77, NA, 15, 13, 7, NA, NA, 5, 2, 9)
> B
 [1] 33 77 NA 15 13  7 NA NA  5  2  9
> B[ is.na(B)] <- 99
> B
 [1] 33 77 99 15 13  7 99 99  5  2  9
>
```

```
data_missing=read.table(file.choose(),header=T,sep=",")
data_missing
 NAME GENDER AGE INCOME
  GLO FEMALE  25  25000
 JOHN          48     NA
        MALE  55  32000
  LIZ FEMALE  NA  50000
HARRY          35     NA
 LIMA FEMALE  NA   6000
data=na.omit(data_missing)
data
NAME GENDER AGE INCOME
 GLO FEMALE  25  25000
       MALE  55  32000
data=edit(data_missing)
```

Data Editor

	NAME	GENDER	AGE	INCOME
1	GLO	FEMALE	25	25000
2	JOHN		48	NA
3		MALE	55	32000
4	LIZ	FEMALE	NA	50000
5	HARRY		35	NA
6	LIMA	FEMALE	NA	6000
7				

```
data_missing
 NAME GENDER AGE INCOME
  GLO FEMALE  25  25000
 JOHN          48     NA
        MALE  55  32000
  LIZ FEMALE  NA  50000
HARRY          35     NA
 LIMA FEMALE  NA   6000
data
 NAME GENDER AGE INCOME
  GLO FEMALE  25  25000
 JOHN          48  52000
        MALE  55  32000
  LIZ FEMALE  28  50000
HARRY          35   2500
 LIMA FEMALE  45   6000
```

INTERPRETATION

In the first command we created a variable “B” with missing observations by assigning NA in the place of missing observations. Then we substituted those NA’s with 99 using is.na() code. Next we imported a dataset named missing_data with missing observations and after that we omitted the rows with NA using the na.omit(). Data editor window is used to input value 25 in the placed of NA under age.

18-PST-044

23/7/18

MERGING DATASETS

EXERCISE 18:

AIM: To learn different methods of merging datasets in R.

R CODE AND OUTPUT:

```
> tbl=read.csv(file.choose(),header=T,sep=",")
> tbl
  Dept.No    Name Gender
1  NGS 02   Jithu   Male
2  NGS 04   Alwin   Male
3  NGS 07    Tojo   Male
4  NGS 09  Melvin   Male
5  NGS 15   Jancy Female
6  NGS 23   Nimmy Female
7  NGS 25    Riya Female
8  NGS 27    Anju Female
> tb2=read.csv(file.choose(),header=T,sep=",")
> tb2
  Dept.No Age Height
1  NGS 03  16    180
2  NGS 06  25    165
3  NGS 07  24    157
4  NGS 09  36    164
5  NGS 12  32    169
6  NGS 23  26    153
7  NGS 25  28    174
8  NGS 29  35    169
```

*OUTER JOIN

```
> OJ=merge(x=tbl,y=tb2,by="Dept.No",all=TRUE)
> OJ
   Dept.No    Name Gender Age Height
1   NGS 02   Jithu   Male  NA     NA
2   NGS 04   Alwin   Male  NA     NA
3   NGS 07    Tojo   Male  24    157
4   NGS 09  Melvin   Male  36    164
5   NGS 15   Jancy Female  NA     NA
6   NGS 23   Nimmy Female  26    153
7   NGS 25    Riya Female  28    174
8   NGS 27    Anju Female  NA     NA
9   NGS 03    <NA>   <NA>  16    180
10  NGS 06    <NA>   <NA>  25    165
11  NGS 12    <NA>   <NA>  32    169
12  NGS 29    <NA>   <NA>  35    169
```

*INNER JOIN

```
> IJ=merge(tbl,tb2)
> IJ
  Dept.No    Name Gender Age Height
1  NGS 07    Tojo   Male  24    157
2  NGS 09  Melvin   Male  36    164
3  NGS 23   Nimmy Female  26    153
4  NGS 25    Riya Female  28    174
```

*LEFT JOIN

```
> LJ=merge(x=tb1,y=tb2,by="Dept.No",all.x=T)
> LJ
  Dept.No    Name Gender Age Height
1  NGS 02   Jithu   Male  NA     NA
2  NGS 04   Alwin   Male  NA     NA
3  NGS 07    Tojo   Male  24    157
4  NGS 09  Melvin   Male  36    164
5  NGS 15   Jancy Female  NA     NA
6  NGS 23   Nimmy Female  26    153
7  NGS 25    Riya Female  28    174
8  NGS 27    Anju Female  NA     NA
```

*RIGHT JOIN

```
> RJ=merge(x=tb1,y=tb2,by="Dept.No",all.y=T)
> RJ
  Dept.No    Name Gender Age Height
1  NGS 07    Tojo   Male  24    157
2  NGS 09  Melvin   Male  36    164
3  NGS 23   Nimmy Female  26    153
4  NGS 25    Riya Female  28    174
5  NGS 03    <NA>   <NA>  16    180
6  NGS 06    <NA>   <NA>  25    165
7  NGS 12    <NA>   <NA>  32    169
8  NGS 29    <NA>   <NA>  35    169
```

*CROSS JOIN

```
> CJ=merge(x=tb1,y=tb2,by=NULL)
> CJ
   Dept.No.x    Name Gender Dept.No.y Age Height
1      NGS 02   Jithu   Male     NGS 03  16    180
2      NGS 04   Alwin   Male     NGS 03  16    180
3      NGS 07    Tojo   Male     NGS 03  16    180
4      NGS 09  Melvin   Male     NGS 03  16    180
5      NGS 15   Jancy Female     NGS 03  16    180
6      NGS 23   Nimmy Female     NGS 03  16    180
7      NGS 25    Riya Female     NGS 03  16    180
8      NGS 27    Anju Female     NGS 03  16    180
9      NGS 02   Jithu   Male     NGS 06  25    165
10     NGS 04   Alwin   Male     NGS 06  25    165
11     NGS 07    Tojo   Male     NGS 06  25    165
12     NGS 09  Melvin   Male     NGS 06  25    165
13     NGS 15   Jancy Female     NGS 06  25    165
14     NGS 23   Nimmy Female     NGS 06  25    165
15     NGS 25    Riya Female     NGS 06  25    165
16     NGS 27    Anju Female     NGS 06  25    165
17     NGS 02   Jithu   Male     NGS 07  24    157
18     NGS 04   Alwin   Male     NGS 07  24    157
```

INTERPRETATION

CSV files tb1 with variables deptno, name and gender and tb2 with variables deptno, age and height are imported. In outer join we merged data according to deptno and put all=TRUE. We got a table with observations from both the tables tb1 and tb2 with NA in the place of missing observations as the output. In inner join only the code merge is used and variables common in both datasets are produced as the output.

In left join we put all.x=TRUE and by=”deptno”, here variables and observations of tb2 are joined with tb1. In right join we put all.y=TRUE and by=”deptno”, here variables and observations of tb1 are joined with tb2. In cross join we put by=”NULL” here multiplication of both tables occurs.

18-PST-O44

28/7/18

CONDITIONAL STATEMENTS

EXERCISE 19:

AIM: To learn conditional statement for and its uses.

R CODE AND OUTPUT:

```
> for(i in 1:10)
+ {
+ print("hi")
+ }
[1] "hi"
[1] "hi"
[1] "hi"
[1] "hi"
[1] "hi"
[1] "hi"
[1] "hi"
[1] "hi"
[1] "hi"
[1] "hi"
```

```
> X=1:10
> for(i in 1:10)
+ {
+ X[i]=i*i
+ }
> X
 [1]   1   4   9  16  25  36  49  64  81 100
>
```

```
> x=c(11,12,13,14,15)
> y=c(1,3,5,7,9)
> z=1:10
> for(i in 1:10)
+ {
+ z[i]=x[i]*y[i]
+ }
> z
 [1]  11  36  65  98 135  NA  NA  NA  NA  NA
```

INTERPRETATION

In the beginning we used for loop to print the word Hi 10 times by assigning sequence 1 to 10 to variable i. Then we used for loop to print squares of numbers from 1 to 10 by putting x[i]=i*i . x is assigned with sequence 1 to 10. In the last case variables x and y are given different values and output z is the product of values of x and y. Since there are 5 values each for x and y and for loop is executed 10 times last 6 output observations are NA.

18-PST-044

28/7/18

DEFINING FUNCTIONS IN R

EXERCISE 20:

AIM: To learn how to define different functions in R.

R CODE AND OUTPUT:

```
> ncx<-function(n,x)
+ {
+ factorial(n)/(factorial(x)*factorial(n-x))
+ }
> ncx(5,2)
[1] 10
```

*Statistical functions

```
> basic.stats<-function(x)
+ {
+ stats<-list()
+ stats$n<-length(x)
+ stats$mn<-mean(x)
+ stats$sd<-sd(x)
+ stats$var<-var(x)
+ stats$mi<-min(x)
+ stats$mx<-max(x)
+ unlist(stats)
+ }
> basic.stats(x)
        n        mn        sd       var        mi        mx
  4.00000  25.00000  12.90994 166.66667  10.00000  40.00000
>
```

INTERPRETATION

In the first case function ncx is defined with x and n as parameters and the mathematical equation for the function is then defined. In the last case function basic.stats is introduced with parameter x and length,mean,standard deviation,variance,minimum and maximum of x are assigned to the function.

OPERATIONS ON MATRIX

EXERCISE 21:

AIM: To learn different matrix operations in R.

R CODE AND OUTPUT:

```
> a=round(runif(25,1,10))
> a
 [1]  1  1  7  9  3  8  8 10  7  7  8  9  7  3  9  5  4  5  3  2  3  4  1  3  3
> b=round(runif(25,1,15))
> b
 [1] 12  5 13  7  9  6 10  1  7  4 13 15  6 11  6 15  7  6  9  7  4  7  2  3  7
> A=matrix(a,5,5)
> A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    8    8    5    3
[2,]    1    8    9    4    4
[3,]    7   10    7    5    1
[4,]    9    7    3    3    3
[5,]    3    7    9    2    3
> B=matrix(b,5,5)
> B
     [,1] [,2] [,3] [,4] [,5]
[1,]   12    6   13   15    4
[2,]    5   10   15    7    7
[3,]   13    1    6    6    2
[4,]    7    7   11    9    3
[5,]    9    4    6    7    7
> A+B
     [,1] [,2] [,3] [,4] [,5]
[1,]   13   14   21   20    7
[2,]    6   18   24   11   11
[3,]   20   11   13   11    3
[4,]   16   14   14   12    6
[5,]   12   11   15    9   10
> A-B
     [,1] [,2] [,3] [,4] [,5]
[1,]  -11    2   -5  -10   -1
[2,]   -4   -2   -6   -3   -3
[3,]   -6    9    1   -1   -1
[4,]    2    0   -8   -6    0
[5,]   -6    3    3   -5   -4

> 4*A
     [,1] [,2] [,3] [,4] [,5]
[1,]    4   32   32   20   12
[2,]    4   32   36   16   16
[3,]   28   40   28   20    4
[4,]   36   28   12   12   12
[5,]   12   28   36    8   12
> 5*B
     [,1] [,2] [,3] [,4] [,5]
[1,]   60   30   65   75   20
[2,]   25   50   75   35   35
[3,]   65    5   30   30   10
[4,]   35   35   55   45   15
[5,]   45   20   30   35   35
```

```
> A%*%B
     [,1] [,2] [,3] [,4] [,5]
[1,]  218  141  254  185  112
[2,]  233  139  255  189  118
[3,]  269  188  344  269  134
[4,]  230  160  291  250  121
[5,]  229  123  238  187  106
> B%*%A
     [,1] [,2] [,3] [,4] [,5]
[1,]  256  407  322  202  130
[2,]  204  368  319  175  112
[3,]  116  228  191  121   73
[4,]  181  306  250  151   96
[5,]  139  262  234  126   91
> t(A)
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    1    7    9    3
[2,]    8    8   10    7    7
[3,]    8    9    7    3    9
[4,]    5    4    5    3    2
[5,]    3    4    1    3    3

> t(B)
     [,1] [,2] [,3] [,4] [,5]
[1,]   12    5   13    7    9
[2,]    6   10    1    7    4
[3,]   13   15    6   11    6
[4,]   15    7    6    9    7
[5,]    4    7    2    3    7
> det(A)
[1] -271
> det(B)
[1] -7609
> solve(A)
           [,1]       [,2]       [,3]        [,4]       [,5]
[1,]  0.86715867 -1.1107011 -0.3062731  0.20664207  0.5092251
[2,] -2.54612546  2.8782288  0.9630996 -0.37269373 -1.2398524
[3,]  1.22878229 -1.4760148 -0.4169742  0.08856089  0.7896679
[4,]  2.16974170 -2.1918819 -0.6642066  0.29151292  0.6826568
[5,] -0.05904059  0.2841328 -0.2472325  0.20295203 -0.1070111
> solve(B)
           [,1]        [,2]        [,3]         [,4]        [,5]
[1,] -0.13799448 -0.07504271  0.10211592  0.184912603  0.04547247
[2,] -0.38454462 -0.26245236 -0.07543698  0.755289788  0.18004994
[3,]  0.19503220  0.25272703  0.09567617 -0.421343146 -0.21093442
[4,]  0.13707452 -0.08450519 -0.12143514 -0.003679853  0.04244973
[5,]  0.09291628  0.11433828 -0.04875805 -0.304507820  0.11985806
```

```
> round(A%*%solve(A))
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    1    0    0    0
[3,]    0    0    1    0    0
[4,]    0    0    0    1    0
[5,]    0    0    0    0    1
> round(B%*%solve(B))
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    1    0    0    0
[3,]    0    0    1    0    0
[4,]    0    0    0    1    0
[5,]    0    0    0    0    1
```

INTERPRETATION

First of all we created two vectors a and b as random uniforms variates. And using matrix() we created two five ordered square matrices A and B with the values of a and b. After that we perform matrix addition , subtraction and scalar multiplication on matrices using A+B, A-B, X*A AND X*B where X represents the scalar value. The code A%*%B is used to multiply matrices A and B. The codes t() and det() are used to obtain transpose and determinant of A and B respectively. The code solve() is used to obtain inverse of the matrix and we can obtain identity matrix A using the code A%*solve(A).

18-PST-044

14/8/18

MATRIX ALGEBRA

EXERCISE 22:

AIM: To find rank of a matrix and to check dependency of given vectors.

R CODE AND OUTPUT:

1. Test linear dependency of given vectors v1=[1,4,2,-3] , v2=[2,8,3,-1], v3=[-3,12,2,1] andv4=[2,8,4,-6].

```
> v1=c(1,4,2,-3)
> v2=c(2,8,3,-1)
> v3=c(-3,12,2,1)
> v4=c(2,8,4,-6)
> a=cbind(v1,v2,v3,v4)
> A=matrix(a,4,4)
> A
     [,1] [,2] [,3] [,4]
[1,]    1    2   -3    2
[2,]    4    8   12    8
[3,]    2    3    2    4
[4,]   -3   -1    1   -6
> det(A)
[1] 0
```

2.Determine rank of matrix $\begin{pmatrix} 5 & 15 & 0 & 1 \\ 6 & 18 & 1 & 0 \\ 2 & 6 & 0 & 0 \\ 4 & 12 & 0 & 0 \end{pmatrix}$

```
> a=c(5,15,0,1,6,18,1,0,2,6,0,0,4,12,0,0)
> M=matrix(a,4,4,byrow=T)
> M
     [,1] [,2] [,3] [,4]
[1,]    5   15    0    1
[2,]    6   18    1    0
[3,]    2    6    0    0
[4,]    4   12    0    0
> det(M)
[1] 0
> S=M[1:3,1:3]
> S
     [,1] [,2] [,3]
[1,]    5   15    0
[2,]    6   18    1
[3,]    2    6    0
> det(S)
[1] 3.552714e-15
>
```

3.Test whether the given vectors are linearly dependent, v1=[1,5,6,4] , V2=[8,2,8,7] V3=[9,7,5,5], V4=[7,2,5,5]

```
> v1=c(1,5,6,4)
> v2=c(8,2,8,7)
> v3=c(9,7,5,5)
> v4=c(7,2,5,5)
> v=cbind(v1,v2,v3,v4)
> v
     v1 v2 v3 v4
[1,]  1  8  9  7
[2,]  5  2  7  2
[3,]  6  8  5  5
[4,]  4  7  5  5
> V=matrix(v,4,4)
> V
     [,1] [,2] [,3] [,4]
[1,]    1    8    9    7
[2,]    5    2    7    2
[3,]    6    8    5    5
[4,]    4    7    5    5
> det(V)
[1] 75
```

4.Determine rank of given matrix A= $\begin{bmatrix} 1 & 2 & 5 & 5 & 4 \\ 5 & 10 & 45 & 2 & 9 \\ 6 & 12 & 7 & 4 & 6 \\ 4 & 8 & 84 & 6 & 8 \\ 8 & 16 & 5 & 8 & 10 \end{bmatrix}$

```
> a=c(1,2,5,5,4,5,10,45,2,9,6,12,7,4,6,4,8,84,6,8,8,16,5,8,10)
> A=matrix(a,5,5,byrow=T)
> A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    5    5    4
[2,]    5   10   45    2    9
[3,]    6   12    7    4    6
[4,]    4    8   84    6    8
[5,]    8   16    5    8   10
> det(A)
[1] 0
> B=A[1:4,2:5]
> B
     [,1] [,2] [,3] [,4]
[1,]    2    5    5    4
[2,]   10   45    2    9
[3,]   12    7    4    6
[4,]    8   84    6    8
> det(B)
[1] 16004
```

INTERPRETATION

In the first part we are given four vectors and to test the independency among them we convert them into matrix A format after that we can see that the determinant of that matrix is zero that is, the given vectors are dependent.

In the next part we are going to find out the rank of the given matrix. So we find the determinant of the given matrix and it is zero and hence we can say that the rank is not same as order of matrix. So we move to make matrix in next lower form i.e. 3*3 matrix. Here we got a nonzero value as our determinant. So we can say that the rank of matrix 3 .

In the third question given vectors are made into matrix V and determinant of V is calculated. Determinant is non-zero so vectors v1,v2,v3 and v4 are independent.

In the last question given matrix A is a square matrix of order 5. Determinant of A is zero so i.e. 5 is not the rank of A. Then we find determinant of square sub-matrix B of order 4. Determinant of B is non-zero so rank of matrix A is 4.

EIGEN VALUES AND VECTORS

EXERCISE 23:

AIM: To learn how to get eigen values and eigen vectors of a given matrix.

R CODE AND OUTPUT:

1.Find the eigen values and eigen vectors for the matrix $\begin{bmatrix} 1 & 2 & -1 & 1 \\ -1 & 1 & 1 & 2 \\ 0 & -2 & -1 & 1 \\ 1 & 1 & 0 & 2 \end{bmatrix}$

```
> a=c(1,2,-1,1,-1,1,1,2,0,-2,-1,1,1,1,0,2)
> A=matrix(a,4,4,byrow=T)
> A
     [,1] [,2] [,3] [,4]
[1,]    1    2   -1    1
[2,]   -1    1    1    2
[3,]    0   -2   -1    1
[4,]    1    1    0    2
> eig=eigen(A)
> eig$values
[1]  3.341080+0.000000i  0.389148+1.964521i  0.389148-1.964521i -1.119375+0.000000i
> eig$vectors
                [,1]                    [,2]                    [,3]           [,4]
[1,] 0.605083279+0i -0.6061058+0.0000000i -0.6061058+0.0000000i  0.49774516+0i
[2,] 0.352365298+0i  0.3179039-0.4453812i  0.3179039+0.4453812i -0.02484975+0i
[3,] 0.002121148+0i  0.2019322+0.4988270i  0.2019322-0.4988270i  0.85360990+0i
[4,] 0.713938669+0i -0.0636347+0.1988820i -0.0636347-0.1988820i -0.15159940+0i
```

2.Find the eigen values and eigen vectors for the matrix $\begin{bmatrix} 9 & -2 & -2 & 4 & 0 \\ 1 & 8 & 5 & -1 & 3 \\ 7 & 3 & 5 & -1 & 8 \\ -3 & -4 & -4 & 6 & 6 \\ 10 & 2 & 5 & -5 & 0 \end{bmatrix}$

```
> b=c(9,-2,-2,4,0,1,8,5,-1,3,7,3,5,-1,8,-3,-4,-4,6,6,10,2,5,-5,0)
> B=matrix(b,5,5,byrow=T)
> B
     [,1] [,2] [,3] [,4] [,5]
[1,]    9   -2   -2    4    0
[2,]    1    8    5   -1    3
[3,]    7    3    5   -1    8
[4,]   -3   -4   -4    6    6
[5,]   10    2    5   -5    0
```

```
> eiB$values
[1] 12.5168303+5.039280i 12.5168303-5.039280i  1.4055920+4.389126i  1.4055920-4.389126i  0.1551554+0.000000i
> eiB$vectors
                      [,1]                  [,2]                  [,3]                  [,4]          [,5]
[1,]  0.3152055-0.3363954i  0.3152055+0.3363954i -0.1320501-0.0938533i -0.1320501+0.0938533i -0.1037267+0i
[2,] -0.5681577+0.0000000i -0.5681577+0.0000000i  0.0335016-0.3229608i  0.0335016+0.3229608i -0.3577889+0i
[3,] -0.4103173-0.3425146i -0.4103173+0.3425146i  0.5010180+0.2563729i  0.5010180-0.2563729i  0.7898021+0i
[4,]  0.2116911-0.0699165i  0.2116911+0.0699165i  0.6209539+0.0000000i  0.6209539+0.0000000i  0.4453682+0i
[5,] -0.2060666-0.2946846i -0.2060666+0.2946846i -0.1851646+0.3629222i -0.1851646-0.3629222i -0.1977059+0i
```

3.Find the eigen values and eigenvectors for the following matrix $\begin{bmatrix} 1 & 1 & -2 \\ -1 & 2 & 1 \\ 0 & 1 & -1 \end{bmatrix}$

```
> c=c(1,-1,0,1,2,1,-2,1,-1)
> C=matrix(c,3,3,)
> C
     [,1] [,2] [,3]
[1,]    1    1   -2
[2,]   -1    2    1
[3,]    0    1   -1
> eiC=eigen(C)
> eiC$values
[1]  2  1 -1

> eiC$vectors
          [,1]       [,2]          [,3]
[1,] 0.3015113 -0.8017837  7.071068e-01
[2,] 0.9045340 -0.5345225 -1.922963e-16
[3,] 0.3015113 -0.2672612  7.071068e-01
```

4. Find the eigen values and eigen vectors for a 2x2 matrix

```
> d=round(runif(4,10,20))
> D=matrix(d,2,2)
> D
     [,1] [,2]
[1,]   11   17
[2,]   16   17
> eiD=eigen(D)
> eiD$values
[1] 30.763055 -2.763055
> eiD$vectors
           [,1]       [,2]
[1,] -0.6521225 -0.7772192
[2,] -0.7581136  0.6292300
```

INTERPRETATION

In the first part we find eigen values and eigen vectors for the 4x4 matrix using the code eigen() and using the codes 'eig$values' and 'eig$vectors' we can print the eigen values and vectors of the given matrix. Same steps are repeated in sections 2,3,4 for finding eigen values and eigen vectors for 5x5,3x3,2x2 matrices respectively.