DT:09-02-23 **LAB-EXCERCISES**

ITA0443-STATISTICS WITH R PROGRAMING

NAME:S.SHASHIKANTH

REG.NO:192125026

GITHUB LINK:https://github.com/SHASHIKANTHSURABHI/ITA0443-STATISTICS-WITH-R-PROGRAMMIN-G

5.CREATION AND MANIPULATION OF DATAFRAMES IN R

Exercise 1

Consider two vectors: x=seq(1,43,along.with=Id)

y=seq(-20,0,along.with=Id)

Create a data frame 'df' as shown below.

```
&gt;df
Id Letter x y
1 1 a 1.000000 -20.000000
2 1 b 4.818182 -18.181818
3 1 c 8.636364 -16.363636
4 2 a 12.454545 -14.545455
5 2 b 16.272727 -12.727273
6 2 c 20.090909 -10.909091
7 3 a 23.909091 -9.090909
8 3 b 27.727273 -7.272727
9 3 c 31.545455 -5.454545
10 4 a 35.363636 -3.636364
```

```
11 4 b 39.181818 -1.818182
12 4 c 43.000000 0.000000
```

CODE:

```
 x <- seq(1, 43, along.with=Id)
y <- seq(-20, 0, along.with=Id)

df <- data.frame(Id = rep(1:4, each=3),
                 Letter = rep(c("a", "b", "c"), times=4),
                 x = x,
                 y = y)
df
```

OUTPUT:

```
   Id Letter         x          y
1   1      a  1.000000 -20.000000
2   1      b  4.818182 -18.181818
3   1      c  8.636364 -16.363636
4   2      a 12.454545 -14.545455
5   2      b 16.272727 -12.727273
6   2      c 20.090909 -10.909091
7   3      a 23.909091  -9.090909
8   3      b 27.727273  -7.272727
9   3      c 31.545455  -5.454545
10  4      a 35.363636  -3.636364
11  4      b 39.181818  -1.818182
12  4      c 43.000000   0.000000
```

<u>Exercise 2</u>

Using the data frame ‘df’ in Exercise1, Construct the following data frame. Id

x.ay.ax.by.bx.cy.c 1 1 1.00000 -20.000000 4.818182 -18.181818

8.636364 -16.363636 4 2 12.45455 -14.545455 16.272727 -12.727273

20.090909 -10.909091 7 3 23.90909 -9.090909 27.727273 -7.272727

31.545455 -5.454545 10 4 35.36364 -3.636364 39.181818 -1.818182

43.000000 0.000000

<u>CODE:</u>

```
df2 <- data.frame(Id = unique(df$Id),
                  x.a = df[df$Letter == "a", "x"],
                  y.a = df[df$Letter == "a", "y"],
                  x.b = df[df$Letter == "b", "x"],
                  y.b = df[df$Letter == "b", "y"],
                  x.c = df[df$Letter == "c", "x"],
                  y.c = df[df$Letter == "c", "y"])
df2
```

<u>OUTPUT:</u>

```
  Id       x.a        y.a       x.b        y.b       x.c        y.c
1  1  1.000000 -20.000000  4.818182 -18.181818  8.636364 -16.363636
2  2 12.454545 -14.545455 16.272727 -12.727273 20.090909 -10.909091
3  3 23.909091  -9.090909 27.727273  -7.272727 31.545455  -5.454545
4  4 35.363636  -3.636364 39.181818  -1.818182 43.000000   0.000000
```

<u>Exercise 3</u>

Create two data frame df1 and df2:

&gt; df1

Id Age

1 1 14

2 2 12

3 3 15

4 4 10

&gt; df2

Id Sex Code

1 1 F a

2 2 M b

3 3 M c

4 4 F d

From df1 and df2 create M:

&gt;M

Id Age Sex Code

1 1 14 F a

2 2 12 M b

3 3 15 M c 4 4 10 F d

CODE

```
> id<-c("11","22","33","44")

> age<-c("14","12","15","10")

> df1<-data.frame(id,age)

> id<-c("11","22","33","44")

> sex<-c("F","M","M","F")
> code<-c("a","b","c","d")

> df2<-data.frame(id,sex,code)

> m<-merge(df1,df2,by="id")

> print(m)
```

OUTPUT:

```
 id age sex code

1 11  14   F    a

2 22  12   M    b

3 33  15   M    c

4 44  10   F    d
```

Exercise 4

Create a data frame df3:

&gt; df3 id2

score 1 4

100

2 3 98

3 2 94

4 1 99

From M (used in Exercise-3) and df3 create N:

Id Age Sex Code score

1 1 14 F a 99

2 2 12 M b 94

3 3 15 M c 98 4 4 10 F d 100

<u>CODE:</u>

```
df3<-data.frame(id2=c(4,3,2,1),score=c(100,98,94,99))

> df3<-data.frame(id2=c(4,3,2,1),score=c(100,98,94,99))

> n<-merge(m,df3,by.x="id",by.y="id2")

> print(n)
```

<u>OUTPUT:</u>

```
Id Age Sex Code score

1 1 14 F a 99

2 2 12 M b 94

3 3 15 M c 98

 4 4 10 F d 100
```

<u>Exercise 5</u>

Consider the previous one data frame N:

1) Remove the variables Sex and Code

2) From N, create a data frame:

```
values ind

1 1 Id

2 2 Id

3 3 Id

4 4 Id

5 14 Age

6 12 Age

7 15 Age

8 10 Age
```

```
9 99 score

10 94 score

11 98 score

12 100 score
```

CODE:

```
N_without_sex_code <- N[,c("Id", "Age", "score")

values <- c(N_without_sex_code$Id, N_without_sex_code$Age, N_without_sex_code$score)

ind <- c(rep("Id", 4), rep("Age", 4), rep("score", 4)) df_values_ind <- data.frame(values, ind)
```

OUTPUT:

```
values ind

1 1 Id

2 2 Id

3 3 Id

4 4 Id

5 14 Age

6 12 Age

7 15 Age

8 10 Age

9 99 score

10 94 score

11 98 score

12 100 score
```

Exercise 6

For this exercise, we'll use the (built-in) dataset trees.

a) Make sure the object is a data frame, if not change it to a data frame.

b) Create a new data frame A:

&gt;A

Girth Height Volume

mean_tree 13.24839 76 30.17097

min_tree 8.30000 63 10.20000

max_tree 20.60000 87 77.00000

sum_tree 410.70000 2356 935.30000

<u>CODE:</u>

```
data("trees")

if (!is.data.frame(trees)) {

  trees <- as.data.frame(trees)

}

mean_tree <- mean(trees$Girth, na.rm = TRUE)

min_tree <- min(trees$Girth, na.rm = TRUE)

max_tree <- max(trees$Girth, na.rm = TRUE)

sum_tree <- sum(trees$Girth, na.rm = TRUE)

A <- data.frame(

  Girth = c(mean_tree, min_tree, max_tree, sum_tree),

  Height = c(76, 63, 87, 2356),

  Volume = c(30.17097, 10.20000, 77.00000, 935.30000),

  row.names = c("mean_tree", "min_tree", "max_tree", "sum_tree")

)

A
```

<u>OUTPUT:</u>

```
          Girth Height    Volume
mean_tree 13.24839     76 30.17097
min_tree  8.30000     63 10.20000
max_tree 20.60000     87 77.00000
sum_tree 410.70000   2356 935.30000
```

<u>Exercise 7</u>

Consider the data frame A:

1)Order the entire data frame by the first column.

2)Rename the row names as follows: mean, min, max, tree

<u>CODE:</u>

```
A <- A[order(A[, 1]), ]
row.names(A) <- c("min", "mean", "max", "tree")
A
```

<u>OUTPUT:</u>

```
     Girth Height    Volume
min  8.30000     63 10.20000
mean 13.24839     76 30.17097
max  20.60000     87 77.00000
tree 410.70000   2356 935.30000
```

<u>Exercise 8</u>

Create an empty data frame with column types:

```
&gt;df

IntsLogicals Doubles Characters

(or 0-length row.names)
```

CODE:

```
df <- data.frame(

  IntsLogicals = numeric(),

  Doubles = numeric(),

  Characters = character(),

  stringsAsFactors = FALSE

)

df
```

OUTPUT:

```
[1] IntsLogicals Doubles Characters

<0 rows> (or 0-length row.names)
```

Exercise 9

Create a data frame XY

X=c(1,2,3,1,4,5,2)

Y=c(0,3,2,0,5,9,3)

```
&gt; XY

X Y

1 1 0

2 2 3

3 3 2

4 1 0
```

5 4 5

6 5 9

7 2 3

1) look at duplicated elements using a provided R function.

2) keep only the unique lines on XY using a provided R function.

CODE:

```
X = c(1, 2, 3, 1, 4, 5, 2)

Y = c(0, 3, 2, 0, 5, 9, 3)

XY = data.frame(X, Y)

duplicated(XY)

XY_unique = unique(XY)
```

OUTPUT:

```
  X Y

1 1 0

2 2 3

3 3 2

4 4 5

5 5 9
```

Exercise 10

Use the (built-in) dataset Titanic.

a) Make sure the object is a data frame, if not change it to a data frame.

b) Define a data frame with value 1st in Class variable, and value NO in Survived variable and variables Sex, Age and Freq.

```
Sex Age Freq

1 Male Child 0

5 Female Child 0

9 Male Adult 118

13 Female Adult 4
```

CODE:

```
if (!is.data.frame(Titanic)) {

  Titanic = as.data.frame(Titanic)

}

df = subset(Titanic, Class == "1st" & Survived == "No")

df = table(df$Sex, df$Age)

df = as.data.frame(df)

df = cbind(Sex = row.names(df), Age = rep(c("Child", "Adult"), each = 2), Freq = df[,1])
```

OUTPUT:

```
   Sex   Age Freq

1 Male Child    0

2 Female Child    0

3   Male Adult 118

4 Female Adult   4
```

MERGING DATAFRAMES

Exercise 11 a)

Create the following dataframes to merge:

buildings&lt;- data.frame(location=c(1, 2, 3), name=c(&quot;building1&quot;, &quot;building2&quot;,&quot;building3&quot;))

data &lt;-

data.frame(survey=c(1,1,1,2,2,2),location=c(1,2,3,2,3,1),efficiency=c(51,64,70,7,80,58))

The dataframes, buildingsand datahave a common key variable called, “location”.

Use the merge() function to merge the two dataframes by “location”, into a new

dataframe,“buildingStats”.

CODE:

```
buildings <- data.frame(location = c(1, 2, 3),
                    name = c("building1", "building2", "building3"))
data <- data.frame(survey = c(1,1,1,2,2,2),
                location = c(1,2,3,2,3,1),
                efficiency = c(51,64,70,7,80,58))
buildingStats <- merge(buildings, data, by = "location")\
```

OUTPUT:

```
  location     name survey efficiency
1        1 building1     1         51
2        1 building1     2         58
3        2 building2     1         64
4        2 building2     2          7
5        3 building3     1         70
6        3 building3     2         80
```

Exercise 11 b)

Give the dataframes different key variable names:

buildings&lt;- data.frame(location=c(1, 2, 3), name=c(&quot;building1&quot;,&quot;building2&quot;,

&quot;building3&quot;))

data &lt;- data.frame(survey=c(1,1,1,2,2,2), LocationID=c(1,2,3,2,3,1),

efficiency=c(51,64,70,71,80,58))

The dataframes, buildings and data have corresponding variables called, location, and LocationID. Use the merge() function to merge the columns of the two dataframes by the corresponding variables.

<u>CODE:</u>

```
buildings <- data.frame(location = c(1, 2, 3),

                    name = c("building1", "building2", "building3"))

data <- data.frame(survey = c(1,1,1,2,2,2),

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

                efficiency = c(51,64,70,71,80,58))

buildingStats <- merge(buildings, data, by.x = "location", by.y = "LocationID")
```

<u>OUTPUT:</u>

```
  location    name survey efficiency

1      1 building1    1      51

2      1 building1    2      58

3      2 building2    1      64

4      2 building2    2      71

5      3 building3    1      70

6      3 building3    2      80
```

<u>DIFFERENT TYPES OF MERGE IN R</u>

<u>Exercise 12a)InnerJoin:</u>

The R merge() function automatically joins the frames by common variable names. In that case, demonstrate how you would perform the merge in Exercise 11a without specifying the key variable.

CODE:

```
buildings <- data.frame(location = c(1, 2, 3),
                   name = c("building1", "building2", "building3"))
data <- data.frame(survey = c(1,1,1,2,2,2),
               location = c(1,2,3,2,3,1),
               efficiency = c(51,64,70,71,80,58))
buildingStats <- merge(buildings, data)
```

OUTPUT:

```
  location    name survey efficiency
1      1 building1    1      51
2      1 building1    2      58
3      2 building2    1      64
4      2 building2    2      71
5      3 building3    1      70
6      3 building3    2      80
```

Exercise 12b)OuterJoin:

Merge the two dataframes from Exercise 11a. Use the “all=” parameter in the merge() function to return all records from both tables. Also, merge with the key variable, “location”.

CODE:

```
buildings <- data.frame(location = c(1, 2, 3),
                   name = c("building1", "building2", "building3"))
```

```
data <- data.frame(survey = c(1,1,1,2,2,2),
               location = c(1,2,3,2,3,1),
               efficiency = c(51,64,70,71,80,58))
buildingStats <- merge(buildings, data, by = "location", all = TRUE)
```

OUTPUT:

```
  location    name survey efficiency
1      1 building1     1      51
2      1 building1     2      58
3      2 building2     1      64
4      2 building2     2      71
5      3 building3     1      70
6      3 building3     2      80
```

Exercise 12c)Left Join:

Merge the two dataframes from Exercise 11a, and return all rows from the left table. Specify the matching key from Exercise 11a.

CODE:

```
buildings <- data.frame(location = c(1, 2, 3),
                   name = c("building1", "building2", "building3"))
data <- data.frame(survey = c(1,1,1,2,2,2),
               location = c(1,2,3,2,3,1),
               efficiency = c(51,64,70,71,80,58))
buildingStats <- merge(buildings, data, by = "location", all.x = TRUE)
```

OUTPUT:

```
  location    name survey efficiency
```

```
1      1 building1     1       51

2      1 building1     2       58

3      2 building2     1       64

4      2 building2     2       71

5      3 building3     1       70

6      3 building3     2       80
```

Exercise 12d)Right Join:

Merge the two dataframes from Exercise 11a, and return all rows from the right table. Use the matching key from Exercise 11a to return matching rows from the left table.

CODE:

```
buildings <- data.frame(location = c(1, 2, 3),  name = c("building1", "building2", "building3"))

data <- data.frame(survey = c(1,1,1,2,2,2),

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

                efficiency = c(51,64,70,71,80,58))

buildingStats <- merge(buildings, data, by = "location", all.y = TRUE)
```

OUTPUT:

```
  location     name survey efficiency

1      1 building1     1       51

2      2 building2     1       64

3      2 building2     2       71

4      3 building3     1       70

5      3 building3     2       80

6     NA      NA     2      58

7     NA      NA     2      71
```

Exercise 12e)Cross Join:

Merge the two dataframes from Exercise 11a, into a "Cross Join" with each row of "buildings" matched to each row of "data". What new column names are created in "buildingStats"?

CODE:

```
buildingStats <- merge(buildings, data, by = NULL, all = TRUE)
```

OUTPUT:

The new column names created in buildingStats will be location, name, survey, and efficiency.

```
  location   name survey efficiency
1       1 building1     1      51
2       1 building1     1      64
3       1 building1     1      70
4       1 building1     2      71
5       1 building1     2      80
6       1 building1     2      58
7       2 building2     1      51
8       2 building2     1      64
9       2 building2     1      70
10      2 building2     2      71
11      2 building2     2      80
12      2 building2     2      58
13      3 building3     1      51
14      3 building3     1      64
15      3 building3     1      70
```

```
16      3 building3     2      71

17      3 building3     2      80

18      3 building3     2      58
```

Exercise 13MergingDataframe rows:

To join two data frames (datasets) vertically, use the rbind function. The two data frames must have the same variables, but they do not have to be in the same order.

Merge the rows of the following two dataframes:

buildings&lt;- data.frame(location=c(1, 2, 3), name=c(&quot;building1&quot;, &quot;building2&quot;, &quot;building3&quot;))

buildings2 &lt;- data.frame(location=c(5, 4, 6), name=c(&quot;building5&quot;, &quot;building4&quot;, &quot;building6&quot;))

Also, specify the new dataframe as, "allBuidings".

CODE:

```
allBuildings <- rbind(buildings, buildings2)
```

OUTPUT:

```
allBuildings

  location    name

1        1 building1

2        2 building2

3        3 building3

4        5 building5

5        4 building4

6        6 building6
```

Exercise 14

Create a new dataframe, buildings3, that has variables not found in the previous dataframes.

buildings3 &lt;- data.frame(location=c(7, 8, 9), name=c(&quot;building7&quot;, &quot;building8&quot;, &quot;building9&quot;),

startEfficiency=c(75,87,91))

Create a new buildings3 without the extra variables.

CODE:

```
buildings3_new <- subset(buildings3, select = c("location", "name"))
```

OUTPUT:

```
> buildings3_new <- buildings3[, c("location", "name")]

> buildings3_new

  location     name

1        7 building7

2        8 building8

3        9 building9
```

Exercise 15

Instead of deleting the extra variables from buildings3 . append the buildings, and buildings2 with the new variable in buildings3, (from Exercise 14). Set the new data in buildings and buildings2 , (from Exercise 13), to NA.

CODE:

```
buildings$startEfficiency <- NA

buildings2$startEfficiency <- NA

allBuildings <- rbind(buildings, buildings2, buildings3)

allBuildings
```

OUTPUT:

```
  location    name startEfficiency

1        1 building1             NA

2        2 building2             NA

3        3 building3             NA

4        5 building5             NA

5        4 building4             NA

6        6 building6             NA

7        7 building7             75

8        8 building8             87

9        9 building9             91
```