**ITA 0443 - STATISTICS WITH R PROGRAMMING FOR VECTORIZED EXPRESSIONS**

**DAY 2 – LAB EXERCISES**

**Reg No : 191911147**

**Name : A. Kailash nath Reddy**

**IMPLEMENTATION OF VECTOR RECYCLING, APPLY FAMILY & RECURSION**

**1. Demonstrate Vector Recycling in R.**

> vec1=1:6

> vec2=1:2

> print(vec1+vec2)

[1] 2 4 4 6 6 8

**2. Demonstrate the usage of apply function in R**

> sample\_matrix <- matrix(C<-(1:10),nrow=3, ncol=10)

>

> print( "sample matrix:")

[1] "sample matrix:"

> sample\_matrix

[,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]

[1,] 1 4 7 10 3 6 9 2 5 8

[2,] 2 5 8 1 4 7 10 3 6 9

[3,] 3 6 9 2 5 8 1 4 7 10

> print("sum across rows:")

[1] "sum across rows:"

> apply( sample\_matrix, 1, sum)

[1] 55 55 55

> print("mean across columns:")

[1] "mean across columns:"

> apply( sample\_matrix, 2, mean)

[1] 2.000000 5.000000 8.000000 4.333333 4.000000 7.000000 6.666667 3.000000

[9] 6.000000 9.000000

**3. Demonstrate the usage of lapply function in R**

> names <- c("priya", "raj","pawan","sudha","devraj")

> print( "original data:")

[1] "original data:"

> names

[1] "priya" "raj" "pawan" "sudha" "devraj"

> print("data after lapply():")

[1] "data after lapply():"

> lapply(names, toupper)

[[1]]

[1] "PRIYA"

[[2]]

[1] "RAJ"

[[3]]

[1] "PAWAN

[[4]]

[1] "SUDHA"

[[5]]

[1] "DEVRAJ"

**4. Demonstrate the usage of sapply function in R**

> sample\_data<- data.frame( x=c(1,2,3,4,5,6),y=c(3,2,4,2,34,5))

> print( "original data:")

[1] "original data:"

> sample\_data

x y

1 1 3

2 2 2

3 3 4

4 4 2

5 5 34

6 6 5

> print("data after sapply():")

[1] "data after sapply():"

> sapply(sample\_data, max)

x y

6 34

**5. Demonstrate the usage of tapply function in R**

library(tidyverse)

print(" Head of data:")

head(diamonds)

print("Average price for each cut of diamond:")

tapply(diamonds$price, diamonds$cut, mean)

**6. Demonstrate the usage of mapply function in R**

> A = list(c(1, 2, 3, 4))

> B = list(c(2, 5, 1, 6))

> result = mapply(sum, A, B)

> print(result)

[1] 24

**7. Sum of Natural Numbers using Recursion**

> sum<-function(n){

> if (n<=1){

> return(n)

> }else{

> return(n+sum(n-1))

> }

> }

> sum(7)

[1] 28

**8. Write a program to generate Fibonacci sequence using Recursion in R**

> Fibonacci <- numeric(10)

> Fibonacci[1] <- Fibonacci[2] <- 1

> for (i in 3:10) Fibonacci[i] <- Fibonacci[i - 2] + Fibonacci[i - 1]

> print("First 10 Fibonacci numbers:")

[1] "First 10 Fibonacci numbers:"

> print(Fibonacci)

[1] 1 1 2 3 5 8 13 21 34 55

**9. Write a program to find factorial of a number in R using recursion.**

rec\_fac <- function(x){

if(x==0 || x==1)

{

return(1)

}

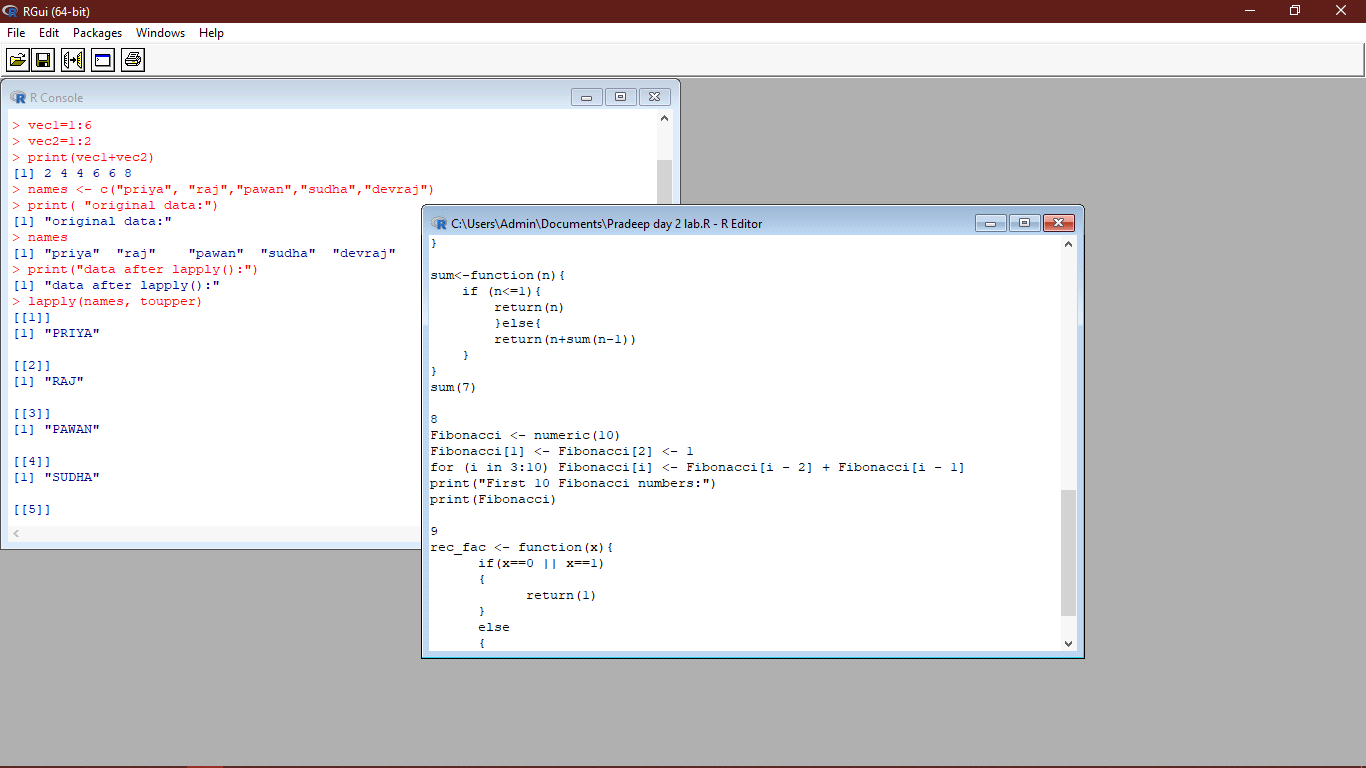
else

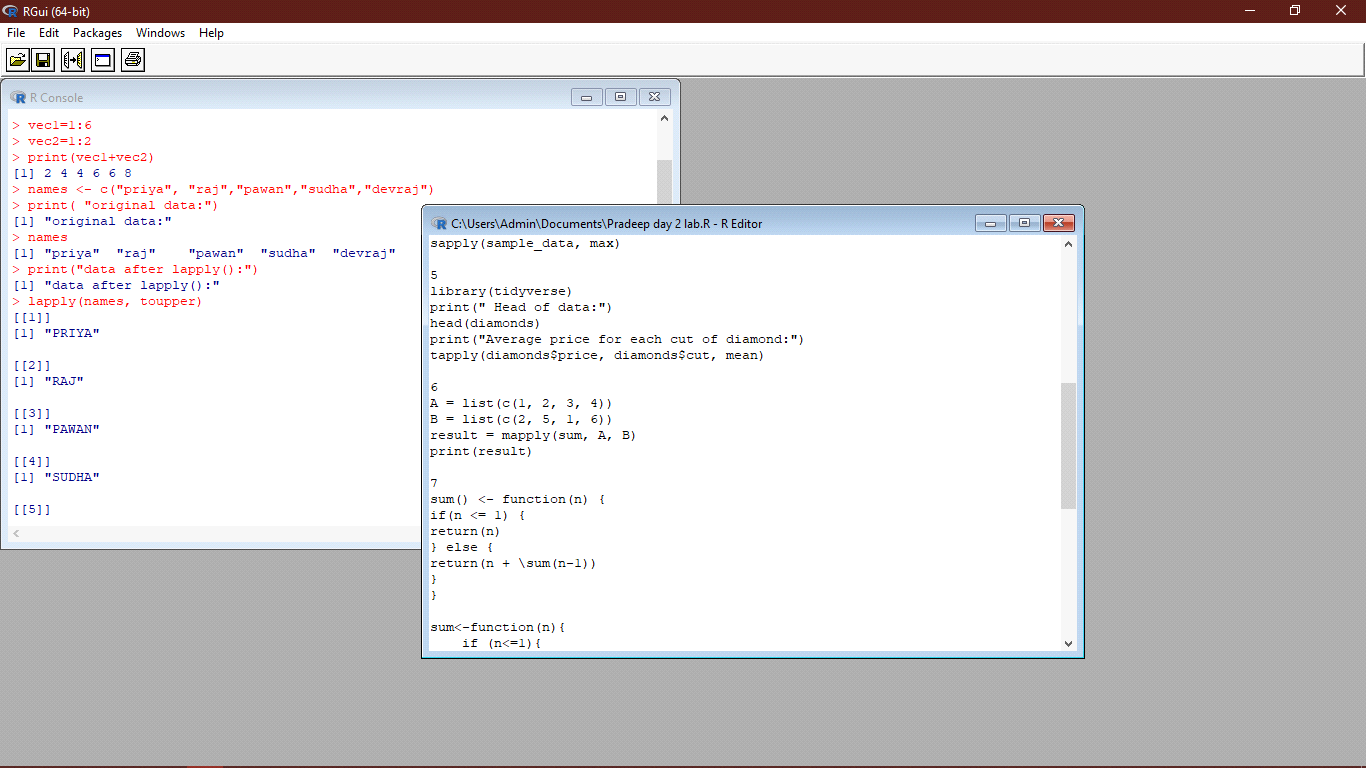
{

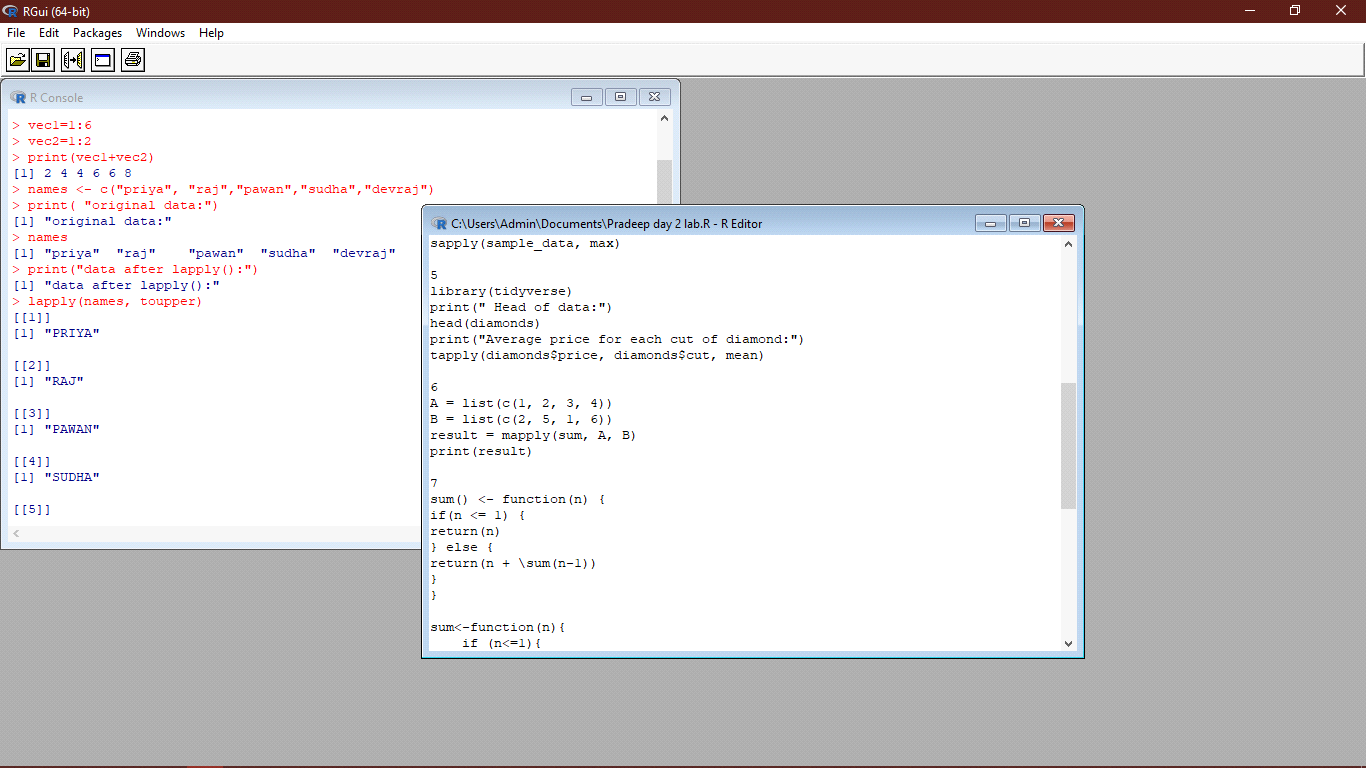
return(x\*rec\_fac(x-1))

}

}







**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

**Program:**

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(letters[1:3], 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

**Exercise 2**

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

**Program:**

library(reshape2)

# Reshape df from long to wide format

df\_wide <- dcast(df, Id ~ paste0(Letter, ".x"), value.var = c("x", "y"))

# Rename the columns

names(df\_wide)[-1] <- gsub("\\.", "\_", names(df\_wide)[-1])

# Print the resulting data frame

df\_wide

**Output:**

Id x\_a\_y x\_a\_x x\_b\_y x\_b\_x x\_c\_y x\_c\_x

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

**Exercise 3**

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

**Program:**

df1 <- data.frame(Id = 1:4, Age = c(14, 12, 15, 10))

df2 <- data.frame(Id = 1:4, Sex = c("F", "M", "M", "F"), Code = c("a", "b", "c", "d"))

M <- merge(df1, df2, by = "Id")

**Output:**

Id Age Sex Code

1 1 14 F a

2 2 12 M b

3 3 15 M c

4 4 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

**Program:**

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

N <- merge(M, df3, by.x = "Id", by.y = "id2")

**Output:**

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

**Exercise 5**

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

**Program:**

N\_new <- subset(N, select = c("Id", "Age", "score"))

N\_stack <- stack(N\_new)

names(N\_stack) <- c("values", "ind")

**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

**Program:**

A <- data.frame(

Girth = c(mean(trees$Girth), min(trees$Girth), max(trees$Girth), sum(trees$Girth)),

Height = c(mean(trees$Height), min(trees$Height), max(trees$Height), sum(trees$Height)),

Volume = c(mean(trees$Volume), min(trees$Volume), max(trees$Volume), sum(trees$Volume))

)

rownames(A) <- c("mean\_tree", "min\_tree", "max\_tree", "sum\_tree")

**Output:**

Girth Height Volume

mean\_tree 13.24839 76.00000 30.17097

min\_tree 8.30000 63.00000 10.20000

max\_tree 20.60000 87.00000 77.00000

sum\_tree 410.70000 2356.00000 935.30000

**Exercise 7**

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

**Program:**

data(trees)

trees\_df <- as.data.frame(trees)

A <- data.frame(

Girth = c(mean(trees\_df$Girth), min(trees\_df$Girth), max(trees\_df$Girth), sum(trees\_df$Girth)),

Height = c(mean(trees\_df$Height), min(trees\_df$Height), max(trees\_df$Height), sum(trees\_df$Height)),

Volume = c(mean(trees\_df$Volume), min(trees\_df$Volume), max(trees\_df$Volume), sum(trees\_df$Volume)),

row.names = c("mean\_tree", "min\_tree", "max\_tree", "sum\_tree")

)

A <- A[order(A$Girth),]

rownames(A) <- c("mean", "min", "max", "tree")

A

**Output:**

Girth Height Volume

mean 13.24839 76.00000 30.170973

min 8.30000 63.00000 10.200000

max 20.60000 87.00000 77.000000

tree 410.70000 2356.00000 935.300000

**Exercise 8**

Create an empty data frame with column types:

&gt;df

IntsLogicals Doubles Characters

(or 0-length row.names)

**Program:**

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

Logicals = logical(),

Doubles = numeric(),

Characters = character(),

row.names = c())

**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

**Program:**

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

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

XY = data.frame(X = X, Y = Y)

**Output:**

X=1,2,3,1,4,5,2

Y=0,3,2,0,5,9,3

**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

**Program:**

is.data.frame(Titanic)

Titanic <- as.data.frame(Titanic)

new\_df <- Titanic[Titanic$Class == "1st" & Titanic$Survived == "No", c("Sex", "Age", "Freq")]

new\_df

**Output:**

Sex Age Freq

1 Male Child 0

5 Female Child 0

9 Male Adult 118

13 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”.

**Program:**

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")

**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.

**Program:**

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")

**DIFFERENT TYPES OF MERGE IN R**

**Exercise 12**

a)InnerJoin:

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.

**Program:**

To perform the merge in Exercise 11a without specifying the key variable, we can use the by argument in the merge() function to specify the common variable name to join on. Since both data frames have a variable called "location", we can use by="location" as the argument:

buildingStats <- merge(buildings, data, by="location")

**13MergingDataframe rows Exercise:**

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”.

**Program:**

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

buildings2 <- data.frame(location=c(5, 4, 6), name=c("building5", "building4", "building6"))

allBuildings <- rbind(buildings, buildings2)

**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.

**Program:**

buildings3 <- data.frame(location=c(7, 8, 9), name=c("building7", "building8", "building9"),

startEfficiency=c(75,87,91))

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

**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.

**Program:**

allBuildings <- rbind(buildings, buildings2, buildings3)

buildings[, "startEfficiency"] <- NA

buildings2[, "startEfficiency"] <- NA