**DAY 2**

NAME: Yennam Sreeja

Reg.No:192011283

***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*

**SOURCE CODE:**

Id <- rep(1:4, each = 3)

Letter <- rep(letters[1:3], 4)

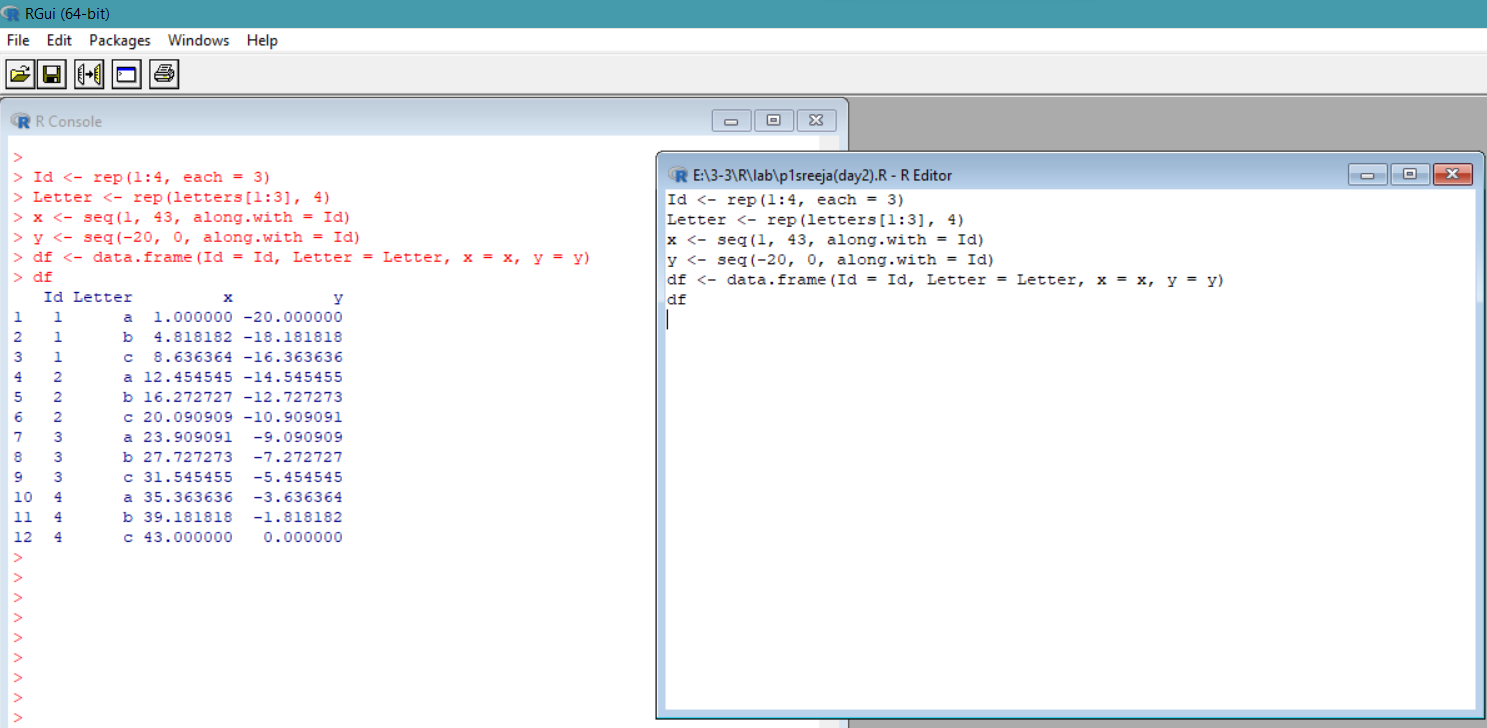
x <- seq(1, 43, along.with = Id)

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

df <- data.frame(Id = Id, Letter = Letter, x = x, y = y)

df

**OUTPUT:**



**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*

**SOURCE CODE:**

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"))

df1

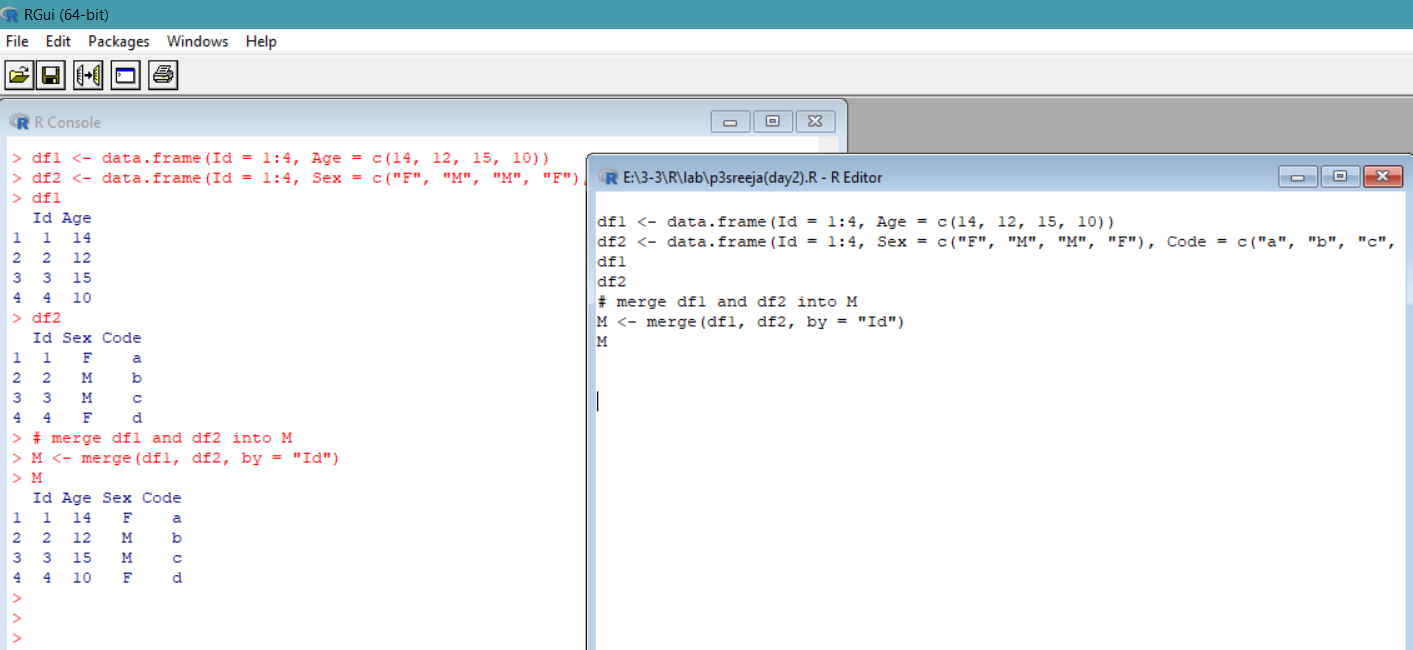
df2

# merge df1 and df2 into M

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

M

**OUTPUT:**

****

**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**

***SOURCE CODE:***

#df1 and df2

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"))

# create M by merging df1 and df2

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

#df3

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

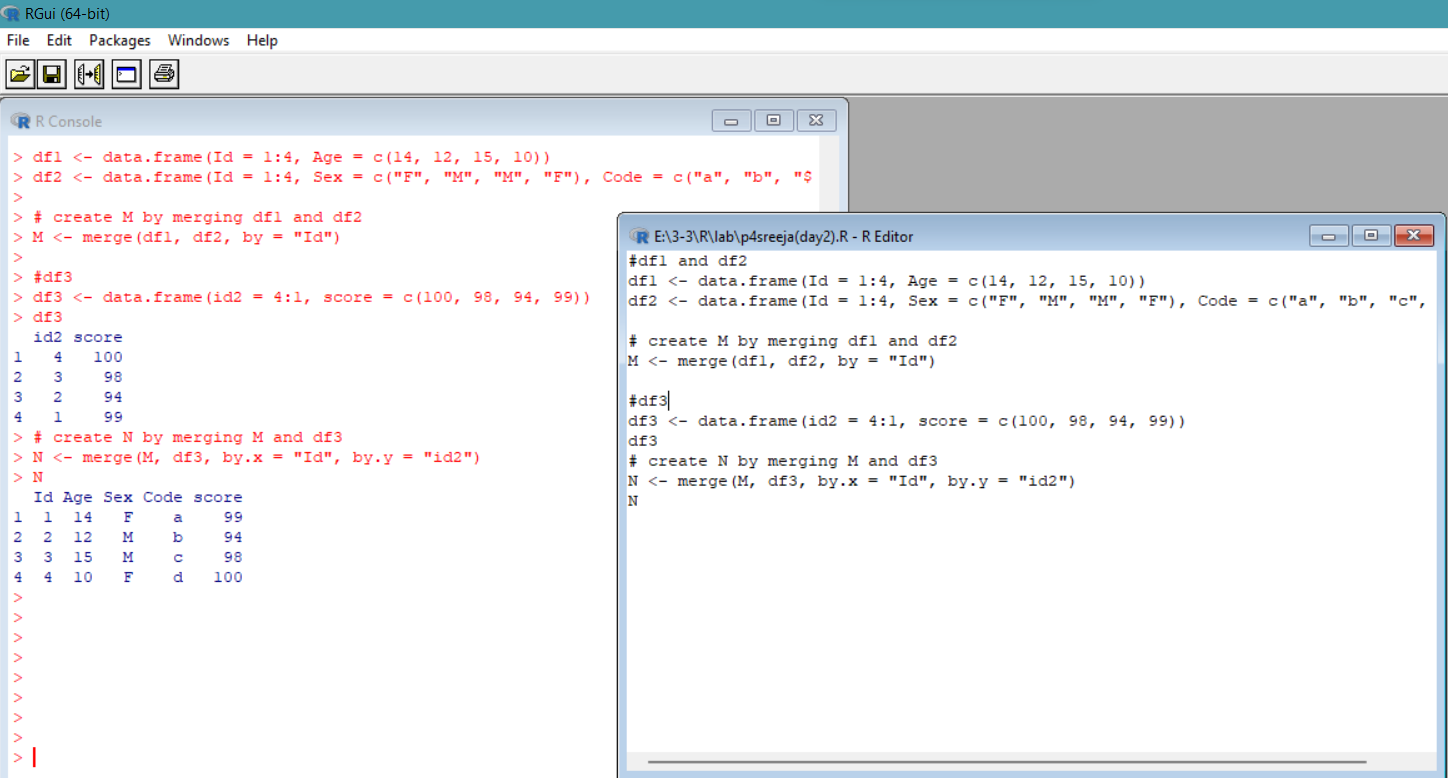
df3

# create N by merging M and df3

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

N

***OUTPUT:***



**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

**SOURCE CODE:**

#df1 and df2

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"))

df1

df2

# Merging df1 and df2 to create M

M <- merge(df1, df2)

#df3

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

df3

# Merging M and df3 to create N

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

# Removing Sex and Code columns

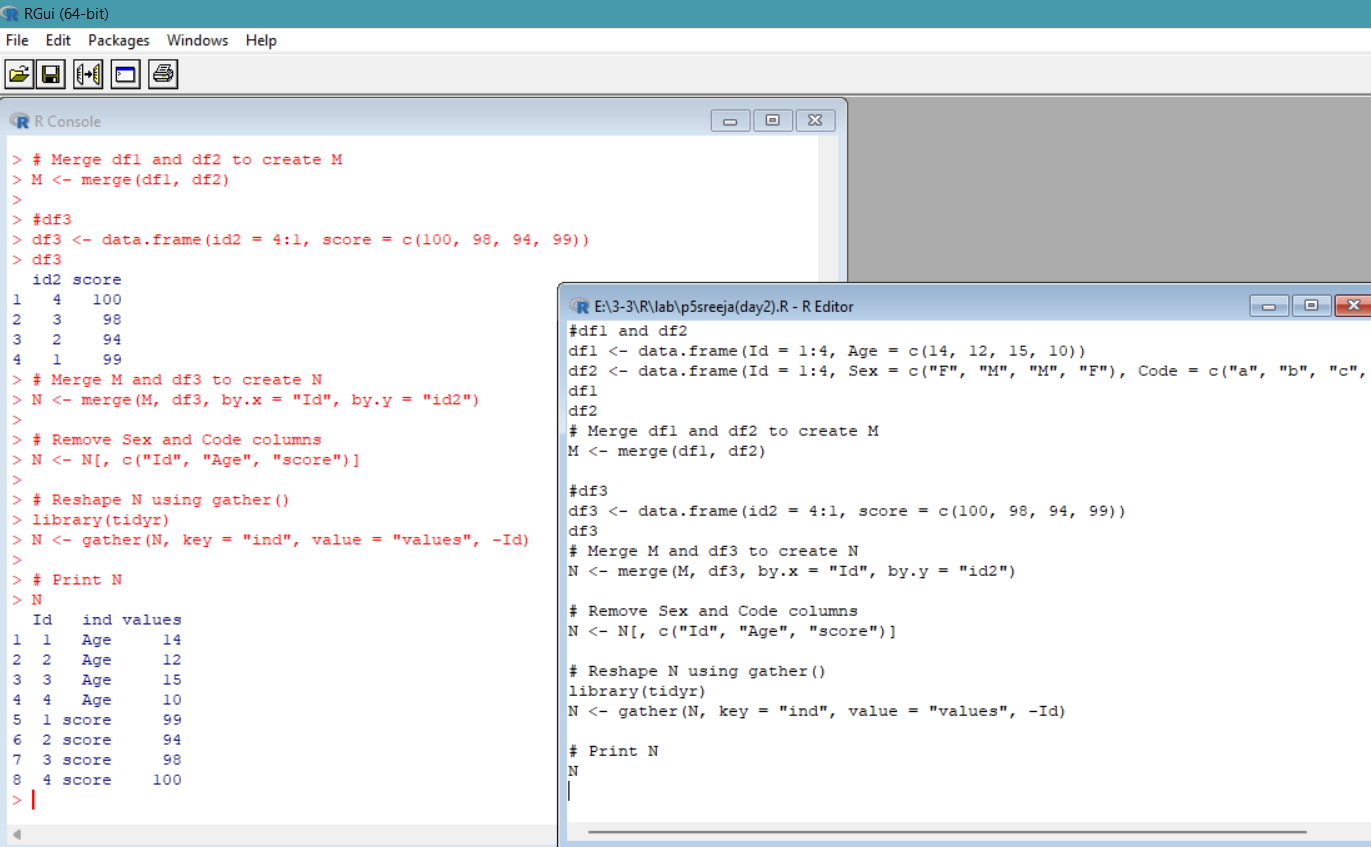
N <- N[, c("Id", "Age", "score")]

# Reshape N using gather()

library(tidyr)

N <- gather(N, key = "ind", value = "values", -Id)

N



**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.

**SOURCE CODE:**

# create data frame XY

X <- c(1,2,3,1,4,5,2)

Y <- c(0,3,2,0,5,9,3)

XY <- data.frame(X,Y)

# check for duplicated elements

duplicated\_rows <- duplicated(XY) | duplicated(XY, fromLast=TRUE)

duplicated\_XY <- XY[duplicated\_rows,]

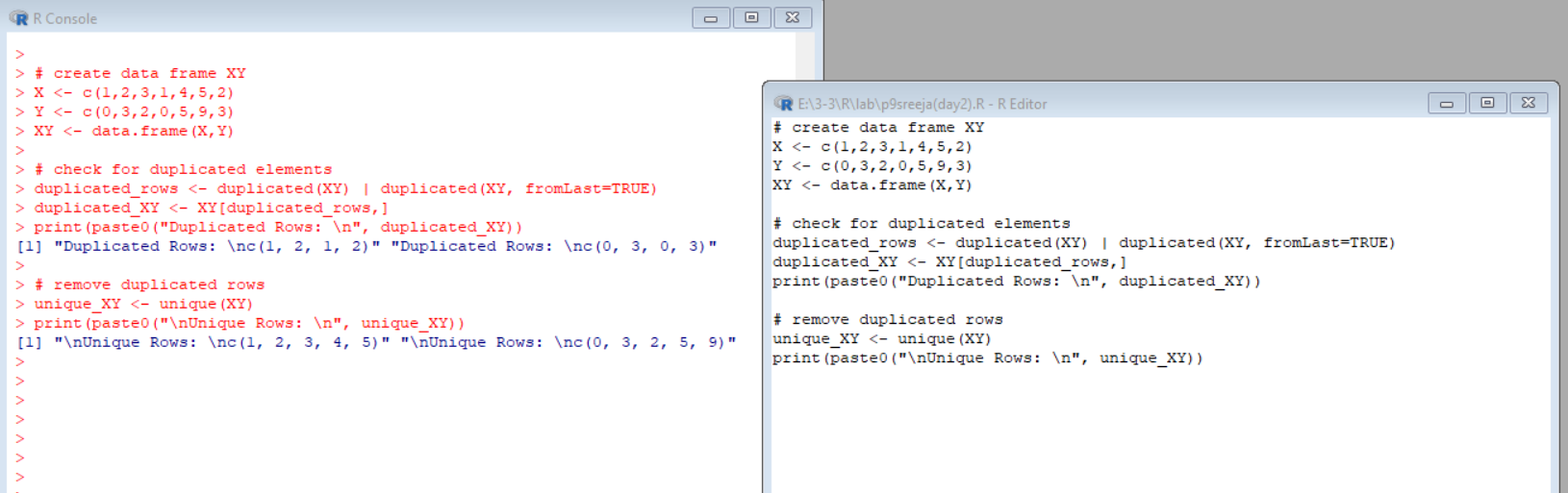
print(paste0("Duplicated Rows: \n", duplicated\_XY))

# remove duplicated rows

unique\_XY <- unique(XY)

print(paste0("\nUnique Rows: \n", unique\_XY))

**OUTPUT:**

****

**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

**SOURCE CODE:**

data(Titanic)

# Making sure the dataset is a data frame

Titanic <- as.data.frame(Titanic)

# Subset the Titanic dataset

subset\_df <- subset(Titanic, Class == "1st" & Survived == "No", select = c(Sex, Age, Freq))

# Display the resulting data frame

subset\_df

# Define another data frame

other\_df <- data.frame(Sex = c("Male", "Female"), Age = c("Child", "Adult"), Value = c(10, 20))

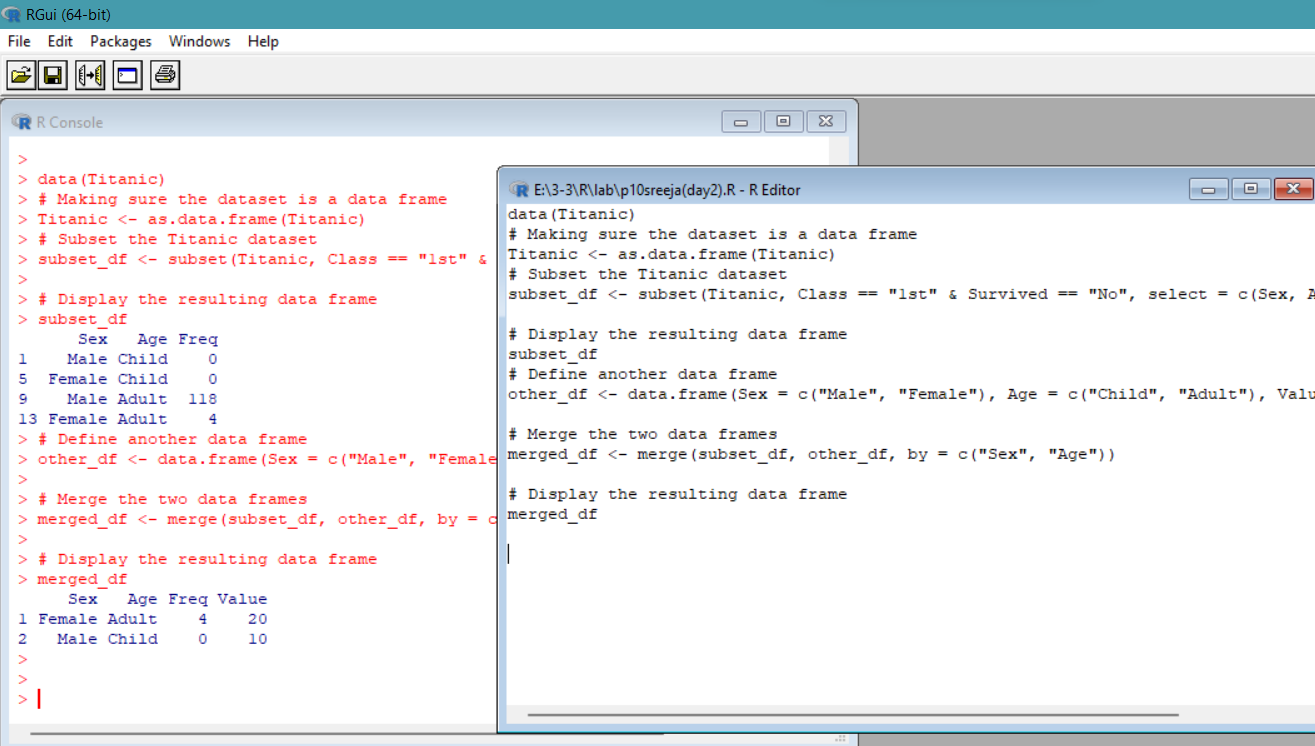
# Merge the two data frames

merged\_df <- merge(subset\_df, other\_df, by = c("Sex", "Age"))

# Display the resulting data frame

merged\_df

**OUTPUT:**

****

**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”.

**SOURCE CODE:**

# Define the buildings data frame

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

# Define the data data frame

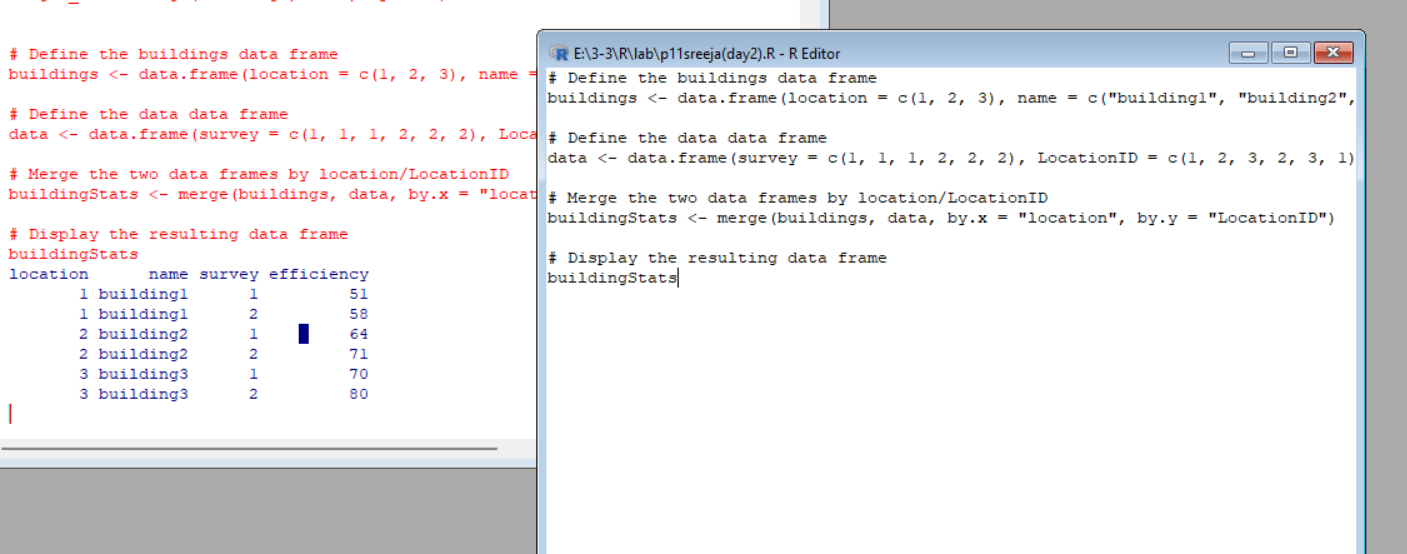
data <- data.frame(survey = c(1, 1, 1, 2, 2, 2), LocationID = c(1, 2, 3, 2, 3, 1), efficiency = c(51, 64, 70, 71, 80, 58))

# Merge the two data frames by location/LocationID

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

# Display the resulting data frame

buildingStats



**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.

**SOURCE CODE:**

# Define the buildings data frame

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

# Define the data data frame

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))

# Merge the two data frames by location/LocationID

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

# Display the resulting data frame

buildingStats

# Define the buildings data frame

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

# Define the data data frame

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))

# Rename the LocationID column in the data data frame to match the location column in the buildings data frame

names(data)[names(data) == "LocationID"] <- "location"

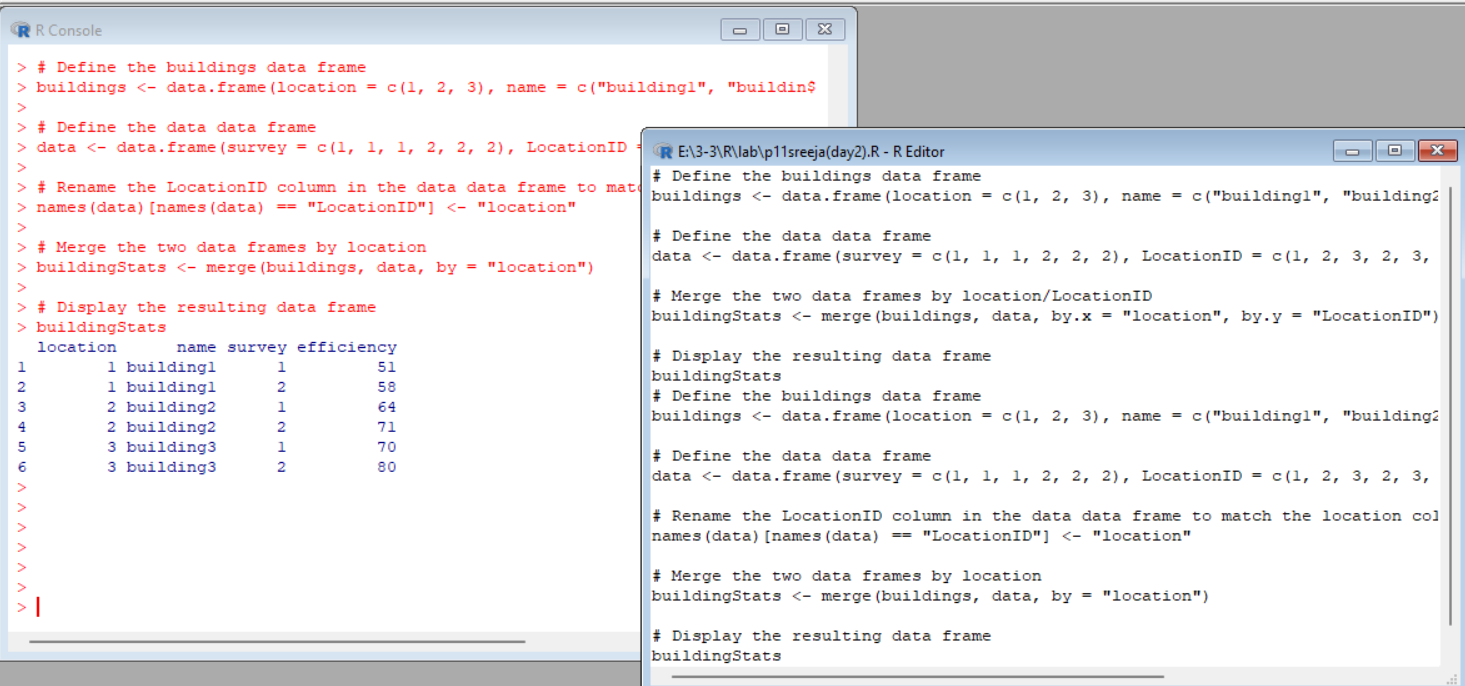
# Merge the two data frames by location

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

# Display the resulting data frame

buildingStats

**OUTPUT:**



**Exercise 12a)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.

**SOURCE CODE:**

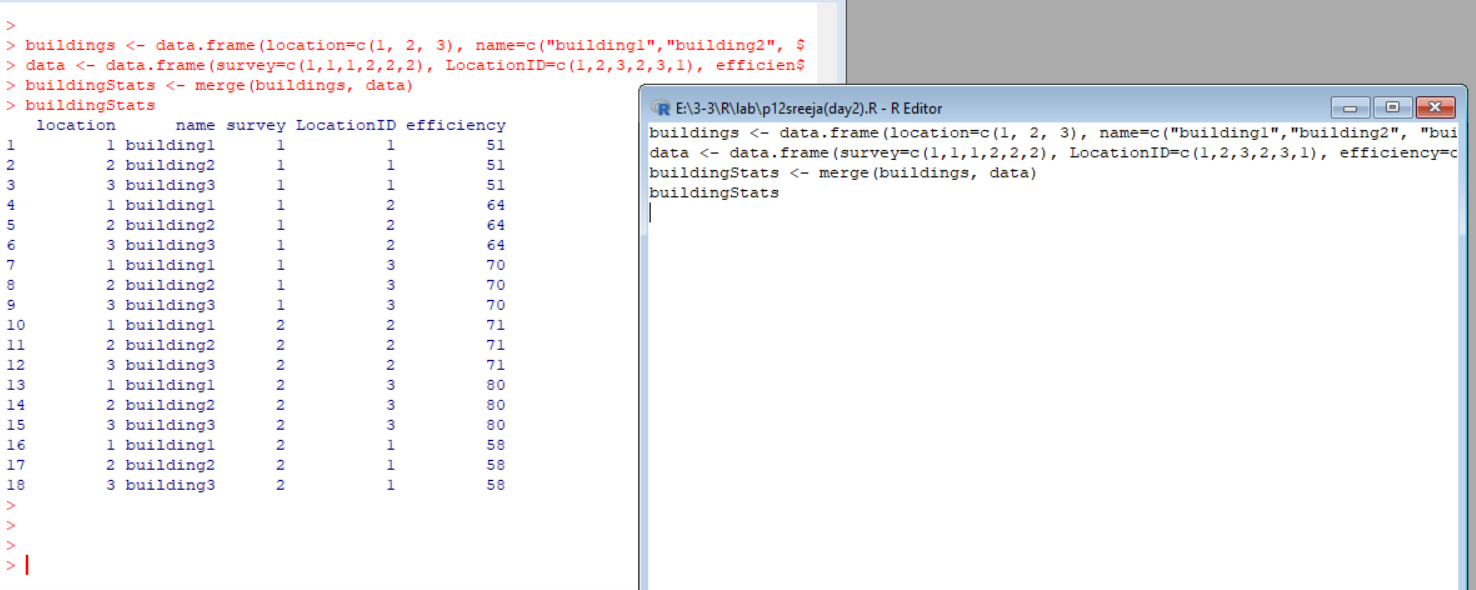
buildings <- data.frame(location=c(1, 2, 3), name=c("building1","building2", "building3"))

data <- data.frame(survey=c(1,1,1,2,2,2), LocationID=c(1,2,3,2,3,1), efficiency=c(51,64,70,71,80,58))

buildingStats <- merge(buildings, data)

buildingStats

**OUTPUT**

****

**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”.

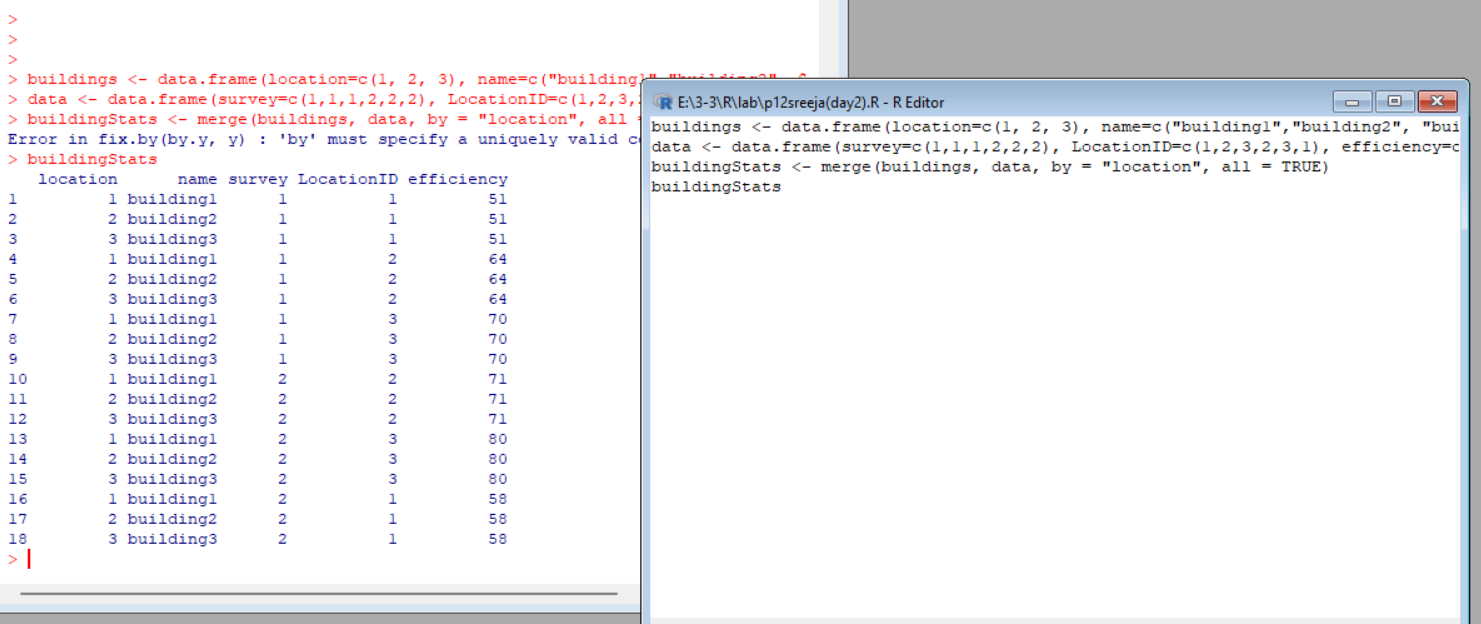
**SOURCE CODE:**

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

data <- data.frame(survey=c(1,1,1,2,2,2), LocationID=c(1,2,3,2,3,1), efficiency=c(51,64,70,71,80,58))

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

buildingStats



**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.

**SOURCE CODE:**

**# Define the buildings data frame**

# Define the buildings data frame

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

# Define the data data frame

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 = "location", all.x = TRUE)

buildingStats

**OUTPUT:**

**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.

**SOURCE CODE:**

**OUTPUT:**

**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”?

**SOURCE CODE:**

**OUTPUT:**

**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”.

**SOURCE CODE:**

# Define the buildings data frame

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

# Define the buildings2 data frame

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

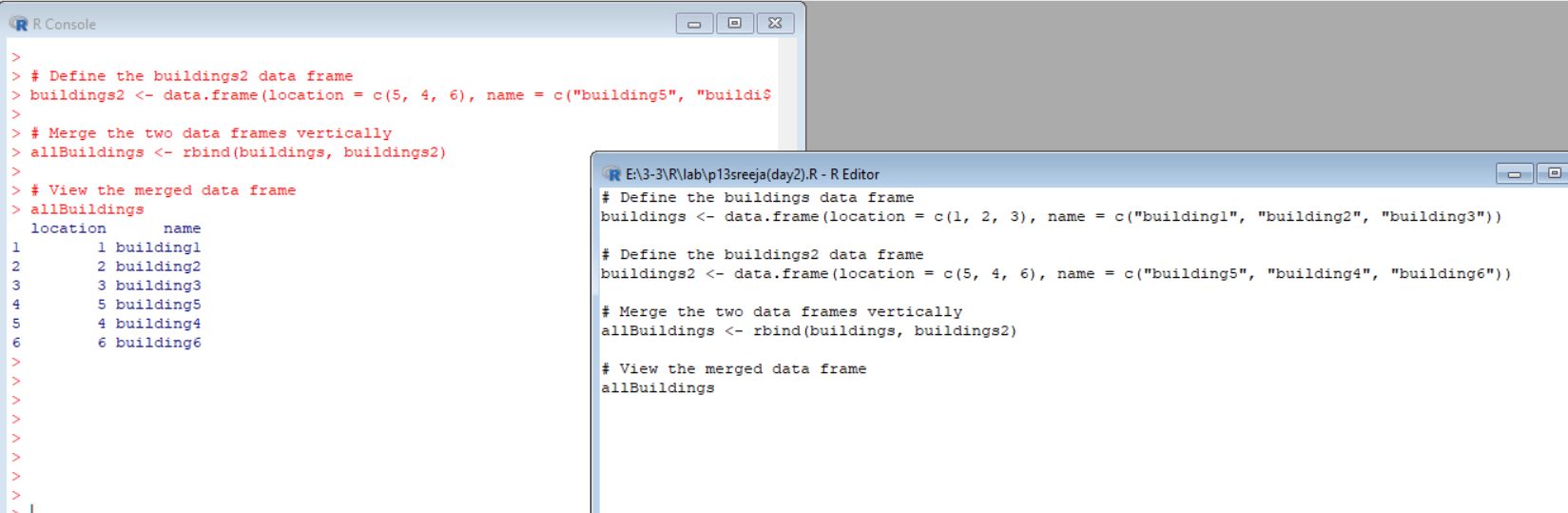
# Merge the two data frames vertically

allBuildings <- rbind(buildings, buildings2)

# View the merged data frame

allBuildings

**OUTPUT:**

****

**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.

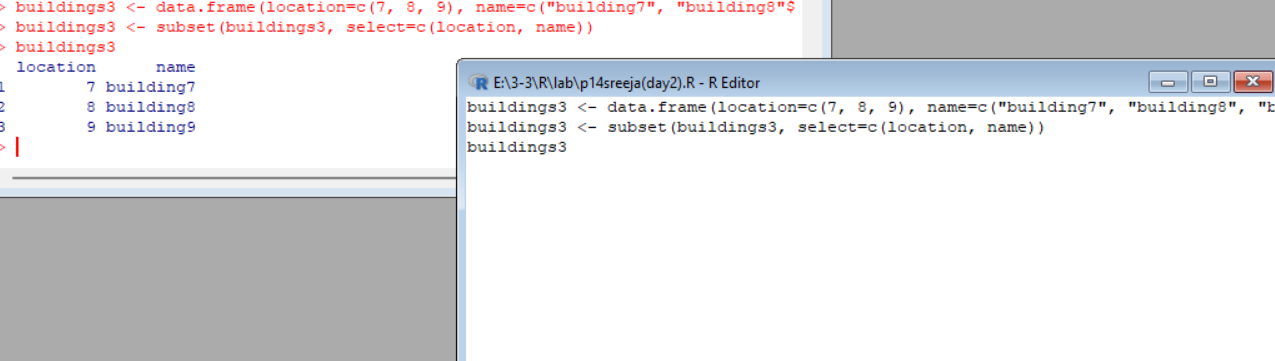
**SOURCE CODE:**

buildings3 <- data.frame(location=c(7, 8, 9), name=c("building7", "building8", "building9"), startEfficiency=c(75,87,91))

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

buildings3

**OUTPUT:**

****

**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.

**SOURCE CODE:**

**# Create buildings3 with extra variable**

buildings3 <- data.frame(location=c(7, 8, 9), name=c("building7", "building8", "building9"), startEfficiency=c(75,87,91))

# Append extra variable to buildings and buildings2

buildings$startEfficiency <- NA

buildings2$startEfficiency <- NA

buildings <- cbind(buildings, startEfficiency = buildings3$startEfficiency)

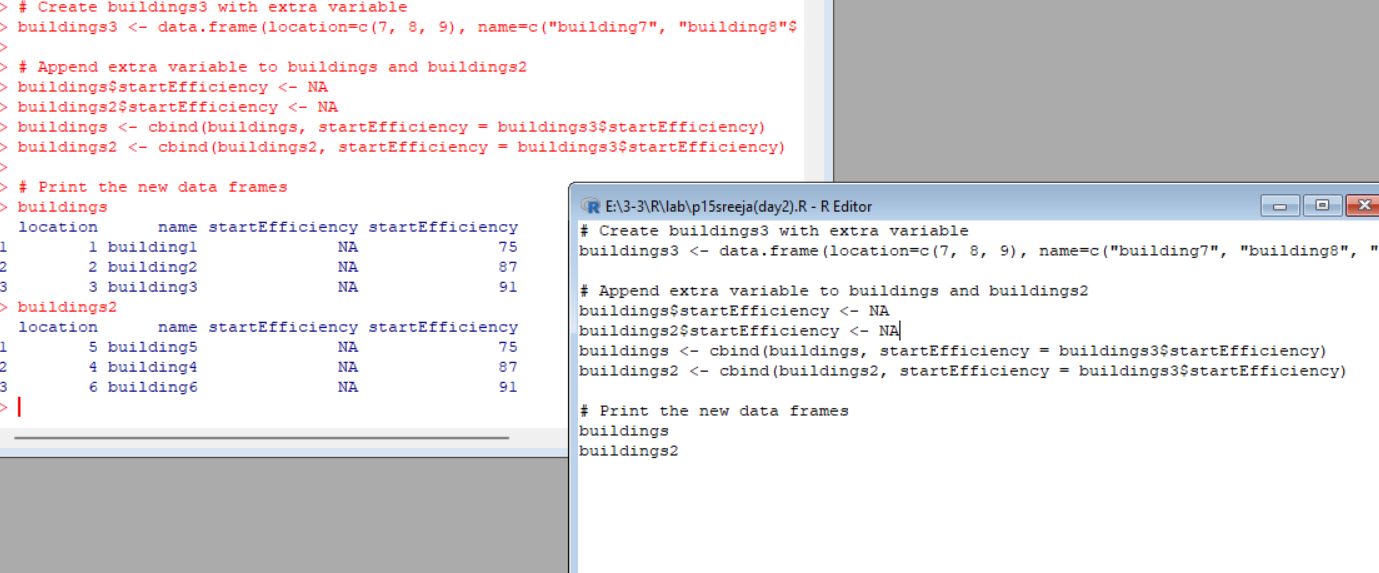
buildings2 <- cbind(buildings2, startEfficiency = buildings3$startEfficiency)

# Print the new data frames

buildings

buildings2

**OUTPUT:**

****

**Exercise: 16**

Construct the following data frame ‘country’.

a) Reshape in R from wide to long:

Reshape the above data frame from wide to long format in R.

b) Reshape in R from long to wide:

**SOURCE CODE:**

library(tidyr)

# creating the 'country' data frame

country <- data.frame(countries = c("A", "B", "C"),

value.population\_in\_million = c(100, 200, 120),

value.gdp\_percapita = c(2000, 7000, 15000))

country

# reshaping from wide to long format

country\_long <- gather(country, key = "variable", value = "value", -countries)

country\_long

# reshaping from long to wide format

country\_wide <- spread(country\_long, key = variable, value = value)

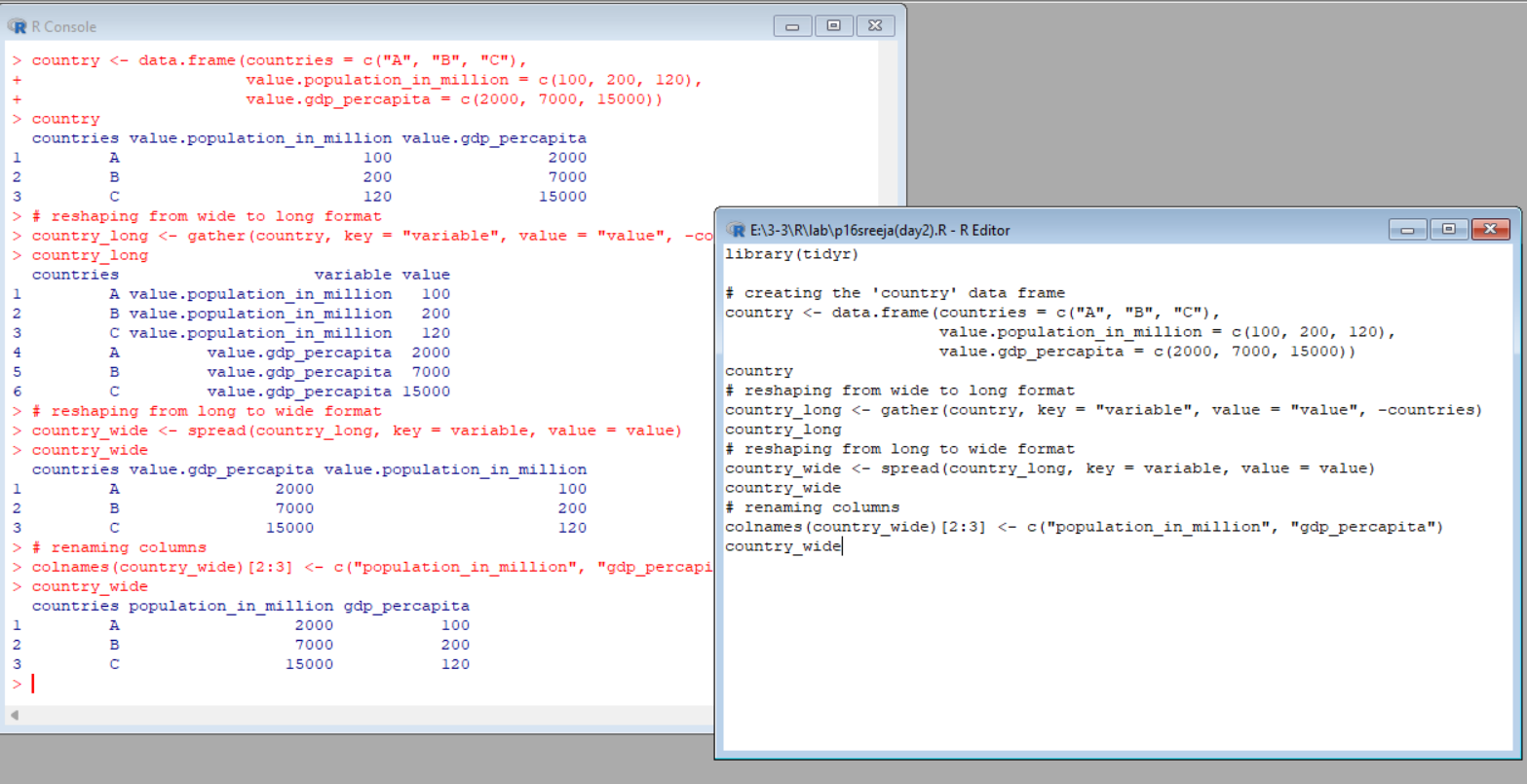
country\_wide

# renaming columns

colnames(country\_wide)[2:3] <- c("population\_in\_million", "gdp\_percapita")

country\_wide

**OUTPUT:**

****

**Exercises 17 :**

1. Melt airquality data set and display as a long – format data ?

2. Melt airquality data and specify month and day to be “ID variables” ?

3. Cast the molten airquality data set .

4. Use cast function appropriately and compute the average of Ozone, Solar.R , Wind

and temperature per month ?

**SOURCE CODE:**

library(reshape2)

# Melt airquality data set

airquality\_melted <- melt(airquality)

# Display as long-format data

head(airquality\_melted)

# Melt airquality data and specify month and day to be "ID variables"

airquality\_melted <- melt(airquality, id.vars = c("Month", "Day"))

# Display as long-format data

head(airquality\_melted)

# Cast the molten airquality data set

airquality\_casted <- dcast(airquality\_melted, Month + Day ~ variable)

# Display as wide-format data

head(airquality\_casted)

# Compute the average of Ozone, Solar.R, Wind, and Temp per month

airquality\_avg <- dcast(airquality\_melted, Month ~ variable, fun.aggregate = mean, value.var = "value")

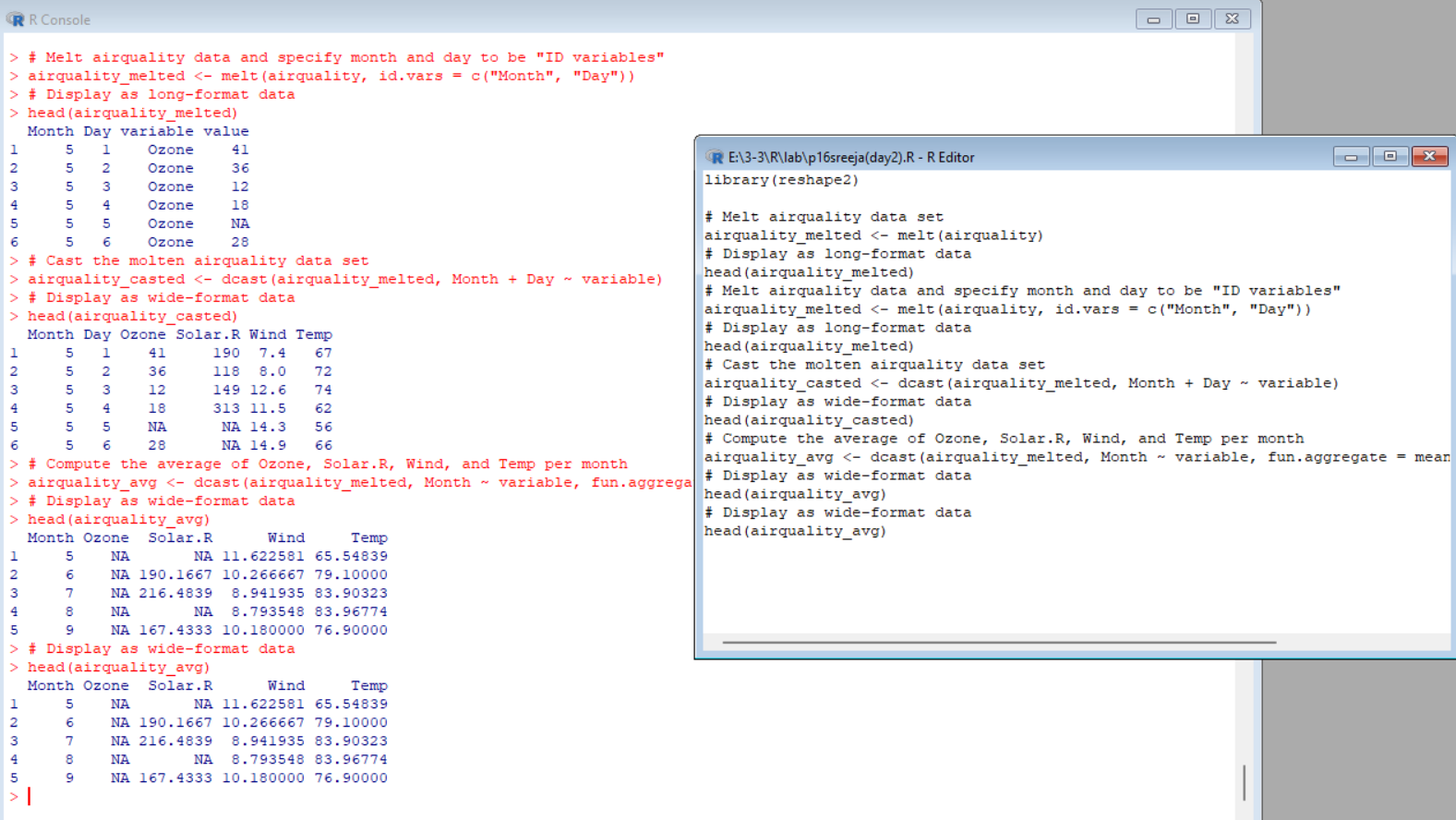
# Display as wide-format data

head(airquality\_avg)

# Display as wide-format data

head(airquality\_avg)

**OUTPUT:**

****