SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

ITA 0443 - STATISTICS WITH R PROGRAMMING FOR REAL TIME PROBLEM

DAY 2 – LAB EXERCISES

Regno :- 192110320

Name :- U.Mohan Srinivas

1. Demonstrate Vector Recycling in R.

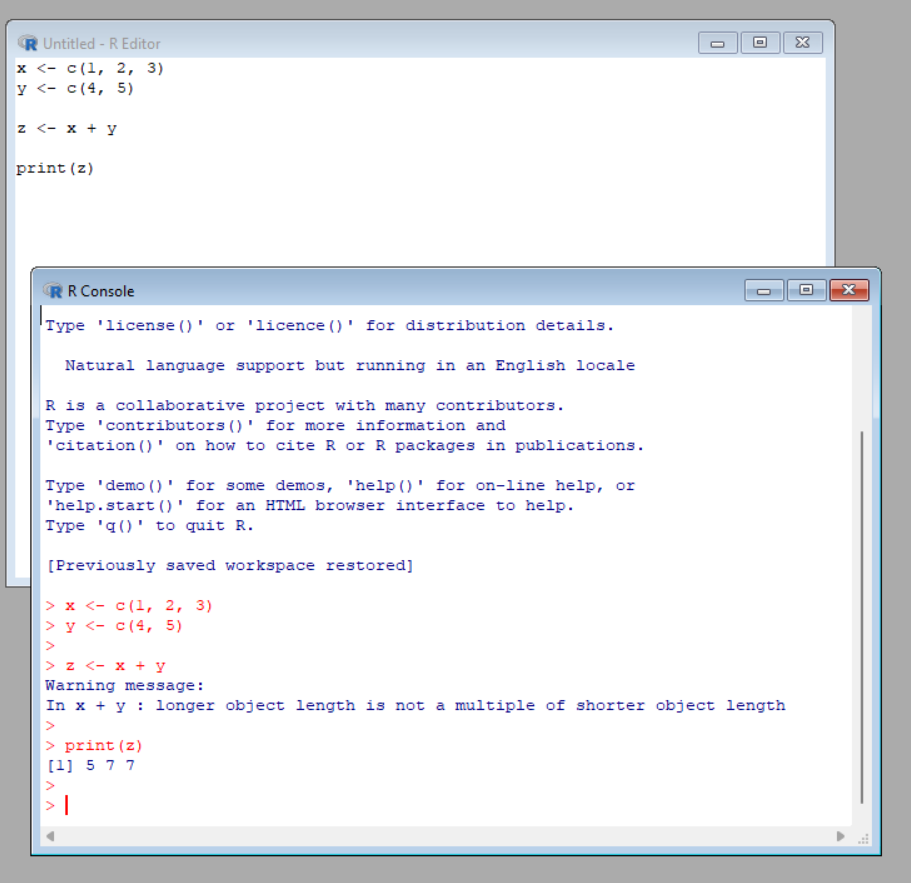
SOURCE CODE:

x <- c(1, 2, 3)

y <- c(4, 5)

z <- x + y

print(z)



2. Demonstrate the usage of apply function in R

SOURCE CODE:

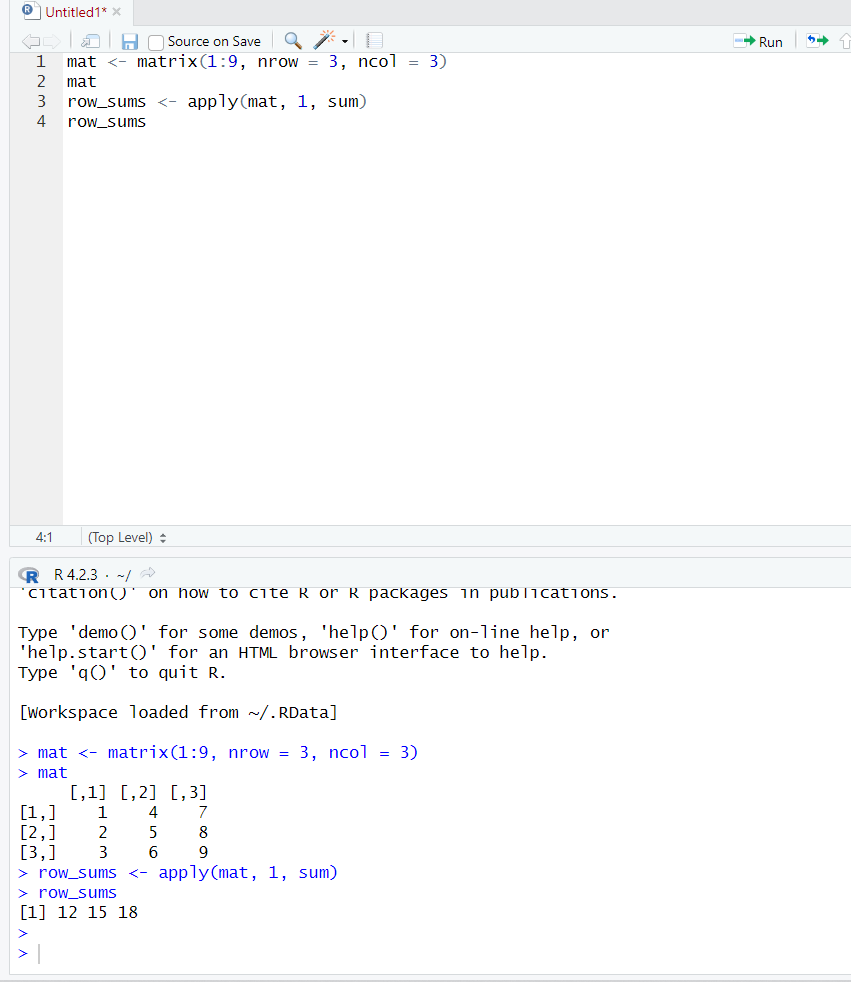
mat <- matrix(1:9, nrow = 3, ncol = 3)

mat

row\_sums <- apply(mat, 1, sum)

row\_sums

OUTPUT:



3. Demonstrate the usage of lapply function in R

SOURCE CODE:

my\_list <- list(1, 2, 3, 4, 5)

square <- function(x) {

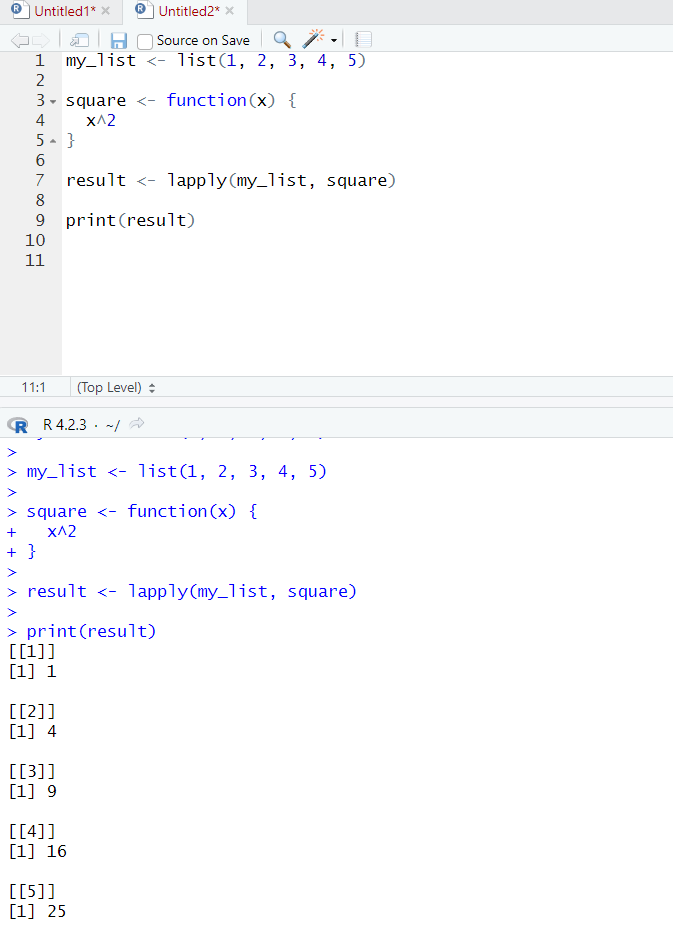
x^2

}

result <- lapply(my\_list, square)

print(result)

OUTPUT:



4. Demonstrate the usage of sapply function in R

SOURCE CODE:

my\_vector <- c(1, 2, 3, 4, 5)

squared\_vector <- sapply(my\_vector, function(x) x^2)

print(squared\_vector)

OUTPUT:



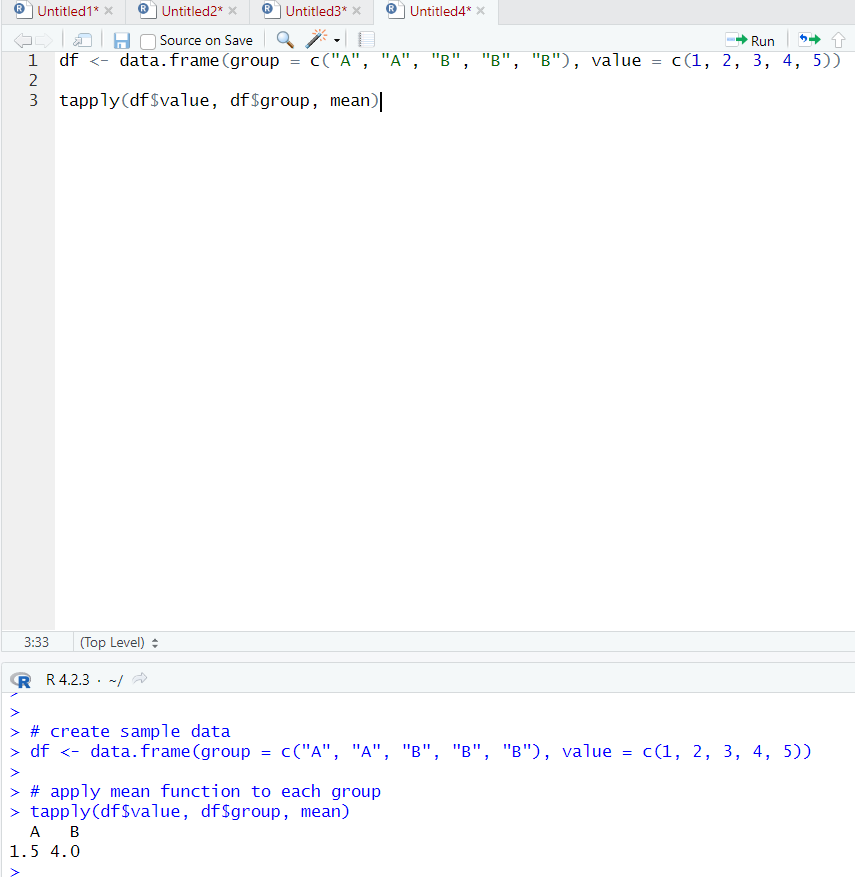
5. Demonstrate the usage of tapply function in R

SOURCE CODE:

df <- data.frame(group = c("A", "A", "B", "B", "B"), value = c(1, 2, 3, 4, 5))

tapply(df$value, df$group, mean)

OUTPUT:



6. Demonstrate the usage of mapply function in R

SOURCE CODE:

x <- c(1, 2, 3, 4)

y <- c(5, 6, 7, 8)

add <- function(a, b) {

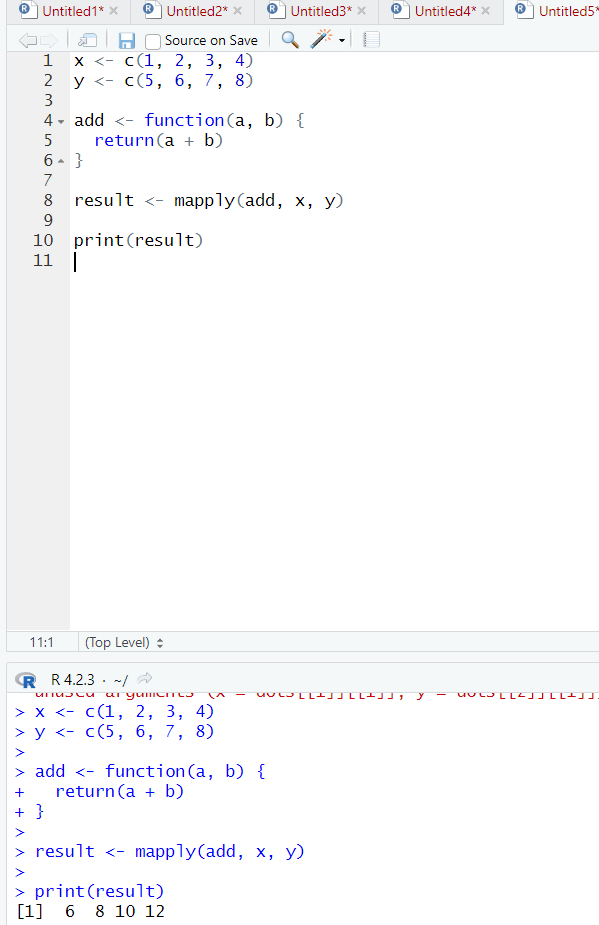
return(a + b)

}

result <- mapply(add, x, y)

print(result)

OUTPUT:



7. Sum of Natural Numbers using Recursion

SOURCE CODE:

sum\_natural <- function(n) {

if (n == 1) {

return(1)

} else {

return(n + sum\_natural(n-1))

}

}

sum\_natural(5)

OUTPUT:



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

SOURCE CODE:

fibonacci <- function(n) {

if (n <= 1) {

return(n)

} else {

return(fibonacci(n - 1) + fibonacci(n - 2))

}

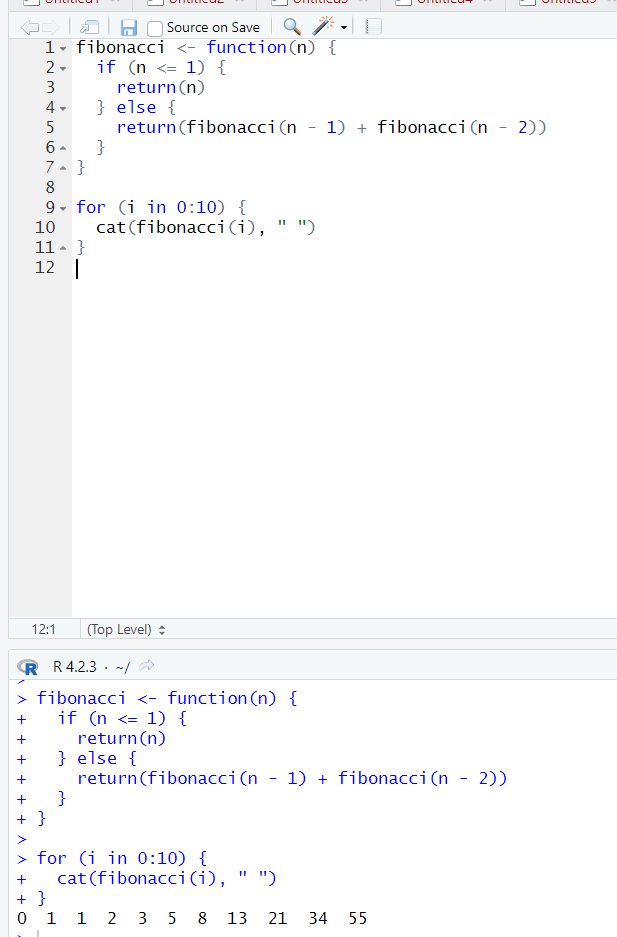
}

for (i in 0:10) {

cat(fibonacci(i), " ")

}

OUTPUT:



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

SOURCE CODE:

factorial <- function(n) {

if (n == 0) {

return(1)

} else {

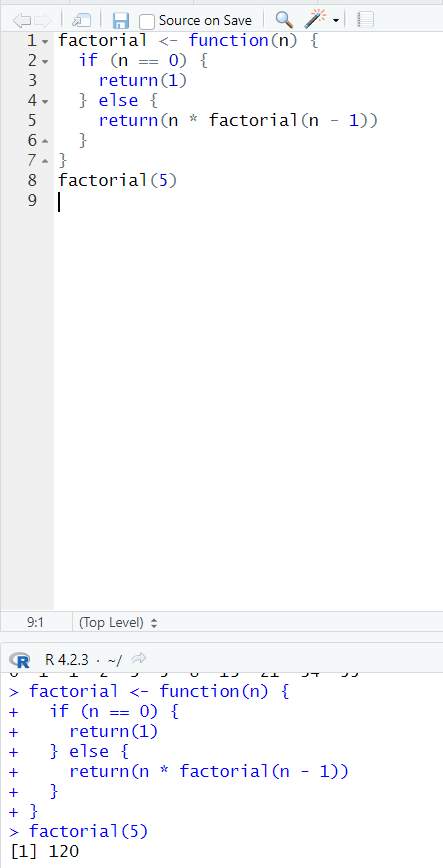
return(n \* factorial(n - 1))

}

}

factorial(5)

OUTPUT:



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:

x <- c(1, 2, 3)

y <- c(4, 5)

z <- x + y

print(z)

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),

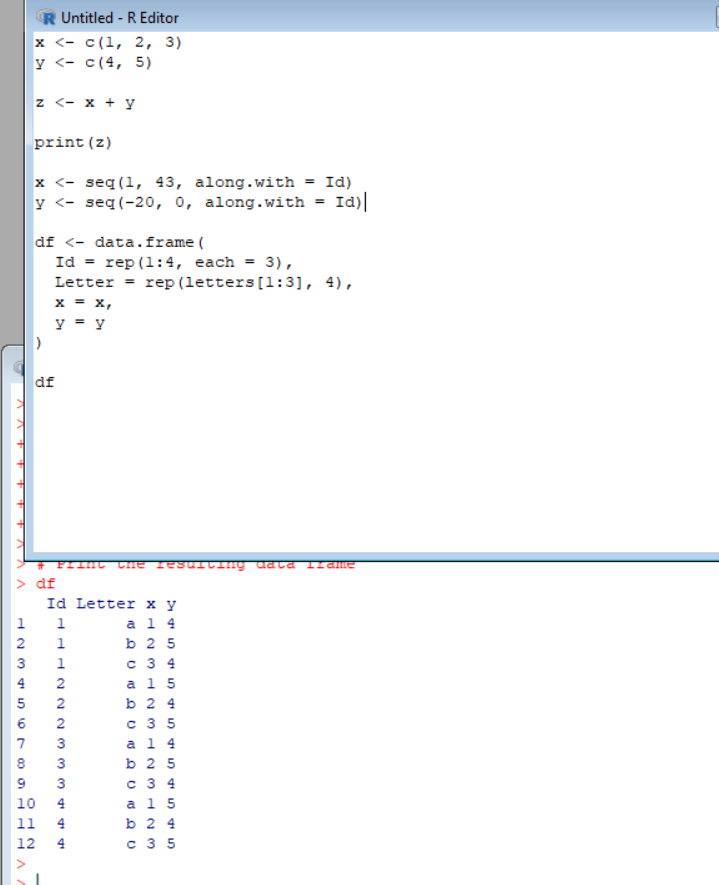
x = x,

y = y

)

df

OUTPUT:



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

SOURCE CODE:

new\_df <- data.frame(

Id = c(1, 2, 3, 4),

x = c(1, 2, 3, 4),

ay = c(1.00000, 12.45455, 23.90909, 35.36364),

ax = c(-20.000000, -14.545455, -9.090909, -3.636364),

by = c(4.818182, 16.272727, 27.727273, 39.181818),

bx = c(-18.181818, -12.727273, -7.272727, -1.818182),

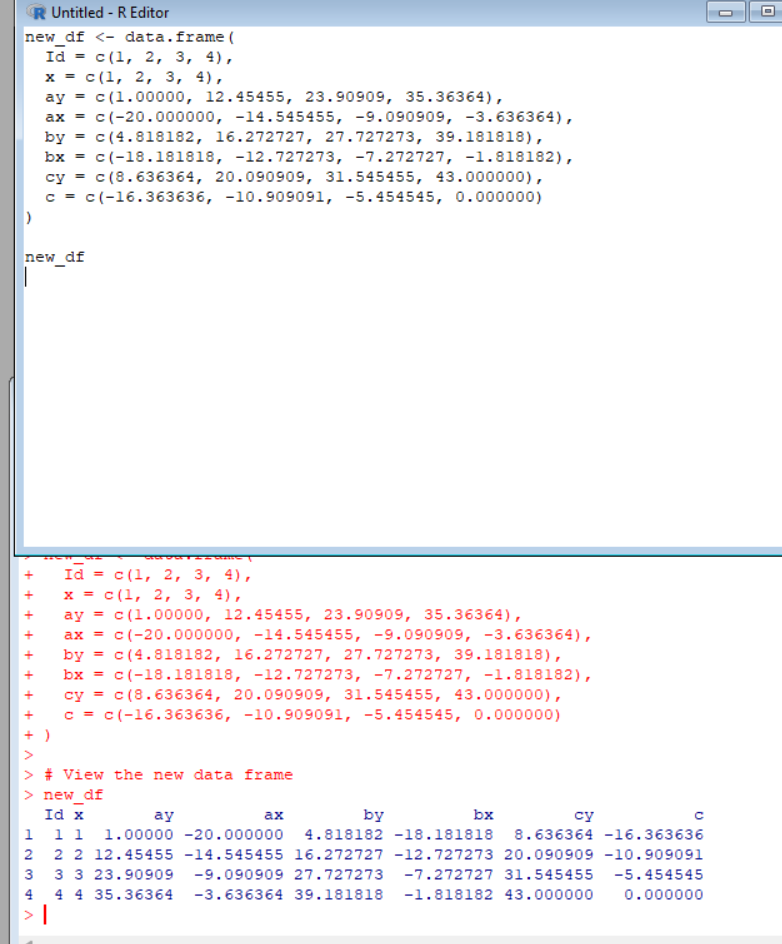
cy = c(8.636364, 20.090909, 31.545455, 43.000000),

c = c(-16.363636, -10.909091, -5.454545, 0.000000)

)

new\_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:

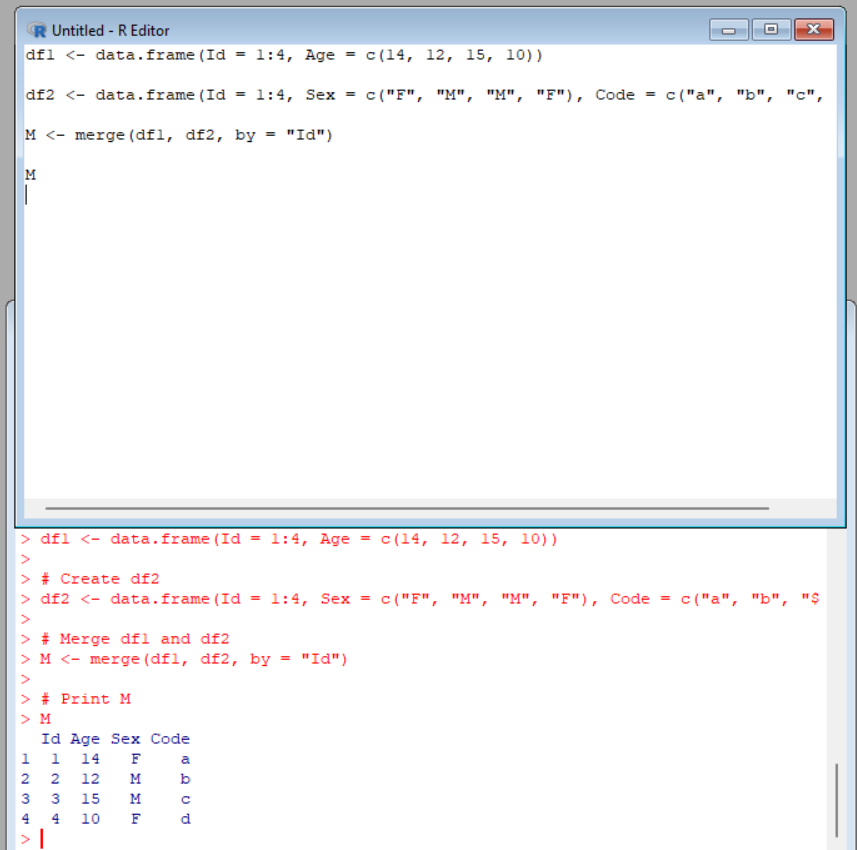
df1 <- data.frame(Id = 1:4, Age = c(14, 12, 15, 10))

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

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:

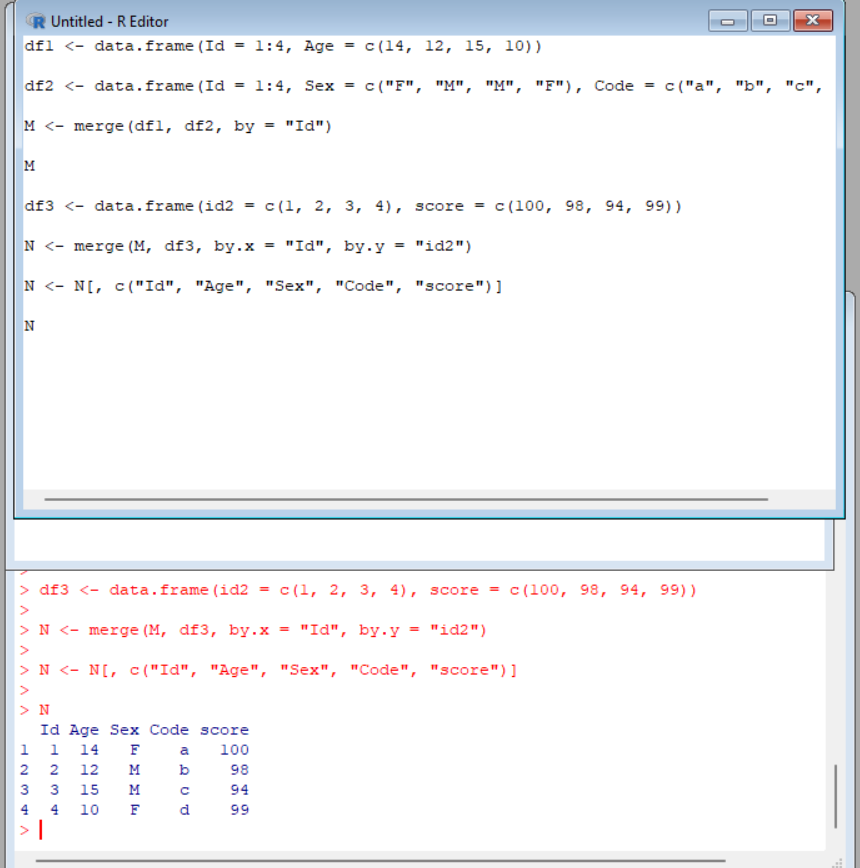
df3 <- data.frame(id2 = c(1, 2, 3, 4), score = c(100, 98, 94, 99))

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

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

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:

# Remove the variables Sex and Code

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

# Reshape the data frame to long format

library(reshape2)

N\_long <- melt(N, id.vars = "Id", variable.name = "ind", value.name = "values")

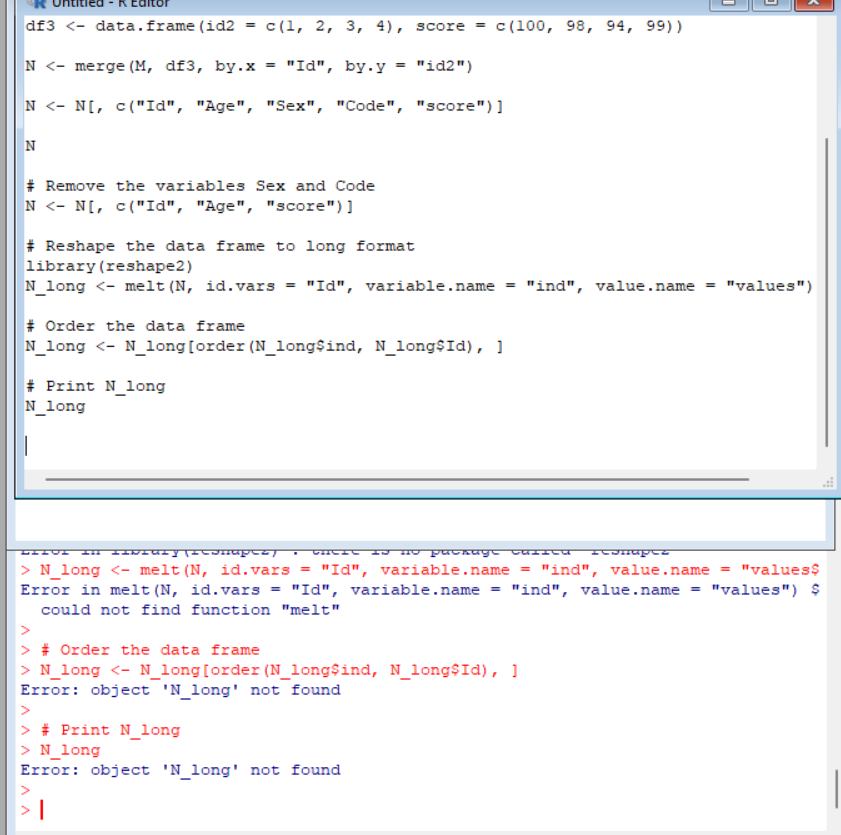
# Order the data frame

N\_long <- N\_long[order(N\_long$ind, N\_long$Id), ]

# Print N\_long

N\_long

OUTPUT:



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

SOURCE CODE:

data(trees)

class(trees)

trees <- as.data.frame(trees)

mean\_girth <- mean(trees$Girth)

mean\_height <- mean(trees$Height)

mean\_volume <- mean(trees$Volume)

min\_girth <- min(trees$Girth)

min\_height <- min(trees$Height)

min\_volume <- min(trees$Volume)

max\_girth <- max(trees$Girth)

max\_height <- max(trees$Height)

max\_volume <- max(trees$Volume)

sum\_girth <- sum(trees$Girth)

sum\_height <- sum(trees$Height)

sum\_volume <- sum(trees$Volume)

A <- data.frame(

Girth = c(mean\_girth, min\_girth, max\_girth, sum\_girth),

Height = c(mean\_height, min\_height, max\_height, sum\_height),

Volume = c(mean\_volume, min\_volume, max\_volume, sum\_volume),

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

)

A

OUTPUT:

