**DAY-2 EXERCISE**

**ITA0443-STATISTICS WITH R PROGRAMMING**

**I.REVANTH REDDY**

**191911268**

[**https://github.com/Revanthreddy28/R-pro.git**](https://github.com/Revanthreddy28/R-pro.git)

**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:**

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

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

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

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

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

df

**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**

**CODE:**

reshape(df,timevar='Letter',idvar='Id',direction='wide') #reshaping the data from long to wide.

**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**

**CODE:**

Id <- c(1:4)

Age <- c(14,12,15,10)

df1 <- data.frame(Id,Age)

Sex <- c("F","M","M","F")

Code <- letters[1:4]

df2 <- data.frame(Id,Sex,Code

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

M

**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**

**CODE:**

id2 <- 4:1

score <- c(100,98,94,99)

df3 <- data.frame(id2,score)

N=merge(M,df3,by.x='Id',by.y='id2')

N

**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**

**CODE:**N[,c("Sex")]=NULL

N[,c("Code")]=NULL

Stack(N)

**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**

**CODE:**

str(trees)

A <- trees

mean\_tree=apply(trees,2,mean)

max\_tree=apply(trees,2,max)

min\_tree=apply(trees,2,min)

sum\_tree=apply(trees,2,sum)

A=data.frame(mean\_tree,min\_tree,max\_tree,sum\_tree) # The expected table is the transpose of A.

A <- t(A)

A

**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**

**CODE:**

A[order(A[,1]),]

row.names(A)

row.names(A) <- c("mean","min","max","tree")

**Exercise 8**

**Create an empty data frame with column types:**

**&gt;df**

**IntsLogicals Doubles Characters**

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

**CODE:**

df <- data.frame(Ints=integer(), Logicals=logical(),Doubles=double(),Characters=character())

**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:**

XY <- data.frame(X=c(1,2,3,1,4,5,2),Y=c(0,3,2,0,5,9,3))

XY

duplicated(XY) # TRUE means a duplicated row.

unique(XY) #4th and 7th rows will not be displayed.

**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:**

str(Titanic)

Tit <- data.frame(Titanic)

df <- subset(Tit, subset = Class=='1st' & Survived=='No',select=c(Sex,Age,Freq))

df

**MERGING DATAFRAMES**

**Exercise 11 a)**

**Create the following dataframes to merge:**

**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))**

**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:**

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

**Exercise 11 b)**

**Give the dataframes different key variable names:**

**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))**

**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.**

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

**DIFFERENT TYPES OF MERGE IN R**

**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.**

**CODE:**

buildingStats <- merge(buildings, data)

**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:**

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

**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:**

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

**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:**

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

**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)

**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<- 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”))**

**Also, specify the new dataframe as, “allBuidings”.**

**CODE:**

allBuidlings <- rbind(buildings, buildings2)

**Exercise 14**

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

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

**startEfficiency=c(75,87,91))**

**Create a new buildings3 without the extra variables.**

**CODE:**buildings3 <- buildings3[,-3]

**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