#Class\_2\_R\_scripts

Instructor-Nabanita Talukdar, *DBA*

x<-10

class(x) #Finding the class of variable x

class(c(TRUE, FALSE)) #Finding class

class(c("she"))

#Vectors

a<-c(1, 2, 3, 4) #a is a numeric vector

b<-c("one", "two", "three") #b is a character vector

c<-c(TRUE, FALSE) #c is a logical vector

#Refer to elements of vector using numeric positions within brackets

a<-c("k","j", "h", "a", "c", "m")

a[3]

a[c(1,3,5)]

a[2:6]

8.5:4.5 #sequence of numbers from 8.5 to 4.5

c(1, 1:3, c(5,8), 13) #values concatenated into single vector

vector("numeric", 5) #creates a vector of specified type and length

numeric(5) #wrapper function to create a vector

seq.int(3,12) #same as 3:12

seq.int(3,12,2) #specifying that the intermediate values are 2 units apart

length(1:5) #length

#length of character vector

sn<-c("sheena", "leads", "Sheila", "needs")

nchar(sn)

#Indexing Vectors

x<-(1:5)^2

x

x[c(1,3,5)] #indexing method 1

x[c(-2,-4)] #indexing method 2

x[c(TRUE, FALSE, TRUE, FALSE, TRUE)] #indexing method 3

names(x)<-c("one", "four", "nine", "sixteen", "twenty five") #naming each element and returning elements

x[c("one","nine", "twenty five")]

#which function

which(x>10)

which.min(x)

which.max(x)

#rep function

rep(1:5,3)

rep(1:5, each=3)

#array function

vector1<-c(2,9,3)

vector2<-c(10, 16, 17, 13, 11, 15)

z<-array(c(vector1, vector2), dim=c(3,3,2))

z

#array function with optional dimnames

vector1 <- c(2,9,6)

vector2 <- c(10,15,13,16,11,12)

column.names <- c("COL1","COL2","COL3")

row.names <- c("ROW1","ROW2","ROW3")

matrix.names<-c("Matrix1", "Matrix2")

z<-array(c(vector1, vector2), dim=c(3,3,2), dimnames=list(row.names, column.names, matrix.names))

z

#matrix function

y<-matrix (1:20, nrow=5, ncol=4) # creates a 5\*4 matrix

y

# 2\*2 matrix filled by rows

cells <-c(1, 26, 24, 68)

rnames<-c("R1", "R2")

cnames<-c("C1","C2")

mymatrix<- matrix(cells, nrow=2, ncol=2,byrow=TRUE, dimnames=list(rnames, cnames))

mymatrix

#2\*2 matrix filled by columns

cells <-c(1, 26, 24, 68)

rnames<-c("R1", "R2")

cnames<-c("C1","C2")

mymatrix<- matrix(cells, nrow=2, ncol=2,byrow=FALSE, dimnames=list(rnames, cnames))

mymatrix

#Using matrix subscript

x<-matrix(1:10, nrow=2)

x

x[2,] #element in second row selected

x[,2] #element in second column selected

x[1,4] #element in first row and 4th column selected

x[1, c(4,5)] # element in first row and 4th & 5th column selected

#rbind function to combine two matrices by rows

a\_matrix<-matrix(1:12, nrow=4, dimnames = list(c("one", "two", "three", "four"), c("ein", "zwei", "drei")))

a\_matrix

another\_matrix<-matrix(seq.int(2,24,2), nrow=4, dimnames=list(c("five", "six", "seven", "eight"), c("vier", "funf", "sechs")))

another\_matrix

rbind(a\_matrix, another\_matrix) #combining two matrices by rows

cbind(a\_matrix, another\_matrix) #combining two matrices by columns

#Creating a dataframe

a\_data\_frame<-data.frame(x=letters[1:5], rnorm(5)) #rnorm function generates a random value from the normal distribution

a\_data\_frame

#set.seed function for creating simulations or random objects to be produced

set.seed(5) #set the seed of R's random number generator, the random numbers generated continues to be the same

a\_data\_frame<-data.frame(x=letters[1:5], rnorm(5)) #rnorm function generates a random value from the normal distribution

a\_data\_frame

#provide your own row names with row.names vector

b\_data\_frame<-data.frame(x=letters[1:5], rnorm(5), row.names = c("Jackie", "Tito", "Jermaine", "Marlon", "Michael"))

b\_data\_frame

colnames(b\_data\_frame) #to get colnames

dimnames(b\_data\_frame) #to get dimnames

#Sub set function

mtcars

subset(mtcars, disp > 160, select=c(disp,mpg, hp))

#rbind function

set.seed(6)

another\_data\_frame<-data.frame(x=letters[3:7], rnorm(5))

another\_data\_frame

rbind(a\_data\_frame,another\_data\_frame)

#cbind function

cbind(a\_data\_frame, another\_data\_frame)

#merge function

merge(a\_data\_frame, another\_data\_frame, by="x")

#creating a list

a\_list<-list(c(1,1,2,5,14,42), matrix(c(3,-8, 1, -3), nrow=2))

a\_list

#names function to name the elements

names(a\_list)<-c("vectors", "numbers")

a\_list

a\_list$vectors

#length function (length is the number of top-level elements that it contains)

length(a\_list)

a\_list[1] #indexing lists

is.list(a\_list) #returns TRUE if the input is a list and FALSE otherwise

busy\_beaver<-c(1,6,21, 107) #converting between vectors and lists

as.list(busy\_beaver)

#Combining Lists

c(list(a=1, b=2), list(3)) #concatenating lists

#NULL

uk\_bank\_holidays\_2013<-list(Jan = "New Year's Day", Feb = NULL)

uk\_bank\_holidays\_2013$Feb

is.null(NULL) #test for null

is.null(NA) #test for NULL

#Factors

heights<-data.frame(height\_cm=c(153,181,150,172), gender=c("female", "male", "female", "male"))

heights

class(heights$gender)

heights$gender

levels(heights$gender) #levels of the factor

nlevels(heights$gender) #number of levels

#Creating Factor using the factor function

gender\_char<-c("female", "male", "female", "male")

(gender\_char<-factor (gender\_char))

gender\_char

#Changing Factor Levels

factor(gender\_char, levels=c("male", "female"))

#Dropping Factor Levels

getting\_to\_work <-data.frame(mode=c("bike", "car", "bus", "car", "walk", "bike", "car", "bike", "car", "car"), time\_mins=c(25, NA, 22, 65, 28, 15, 24, NA,14))

getting\_to\_work

#remove rows where time\_mins is NA

getting\_to\_work<-subset(getting\_to\_work, !is.na(time\_mins))

getting\_to\_work

#unique function

unique(getting\_to\_work$mode)

#droplevels function

getting\_to\_work$mode<-droplevels(getting\_to\_work$mode)

levels(getting\_to\_work$mode)

#ordered factors

status<-c("Poor", "Improved", "Excellent", "Poor")

status<-factor(status, order=TRUE)

status

#Hands\_on problem 3 ordered factors overriding the default by specifying a levels option in the order Poor, Improved, Excellent

status<-c("Poor", "Improved", "Excellent", "Poor")

status<-factor(status, order=TRUE, levels=c("Poor", "Improved", "Excellent"))

status

#attach function

head(mtcars)

summary(mtcars$mpg)

plot(mtcars$mpg, mtcars$disp)

attach(mtcars)

summary(mpg)

plot(mpg, disp)

detach(mtcars)

#with function

with(mtcars, {

print(summary(mpg))

plot(mpg, disp)

plot(mpg, wt)})

#Importing data from Excel

install.packages("xlsx")

library(xlsx)

#Hands\_on 2 problem Reading csv into R (refer for Assignment\_1 problem 2)

getwd()

mydata2<-read.table("trauma.csv", header=TRUE, sep = ",") #make sure the dataset is in working directory, otherwise you have to provide the path

mydata2

df2 <- read.csv("trauma.csv",header=T)

head(df2)

#Reading spss into R

install.packages("foreign")

library(foreign)

df <- read.spss("Table 14.1 Input Transform.sav", use.value.label=TRUE, to.data.frame=TRUE)

#use.value.labels=TRUE tells the function to convert variables with value labels into R factors with those same levels

head(df)

#Hands-ons 1 problem reading text data (refer for Assignment\_1 problem 1)

df3<- read.table("drink.txt", header=TRUE, sep="\t")

df3