Module 01 Lab 01

HD Sheets

2024-10-01

Module 01 Lab 01

For each problem below, create the cells needed and enter your solution

Problem 1

a.) Use seq to create a vector of values x from 10 to 1, decreasing in steps of 1/2

```
x=seq(from=10, to=1, by=(-0.5))
x
```

b.) Find the length of this vector, using an R function

```
length(x)
```

c.) Find the index of the element in x equal to 7.5

```
index_1<-which(x==7.5)
index_1
```

d.) Remove the value 7.5 from the vector x, so the length decreases by 1

```
x<-x[-index_1]
x
```

e.) Sort the vector into both increasing form and decreasing form

```
sort(x,decreasing=FALSE)
sort(x,decreasing=TRUE)
```

Problem 2

Read the help manual for the function runif()

a.) Generate a vector z of 10 random uniform values between 1 and 5

```
z<-runif(10,1,5)
z
```

b.) Find the minimum and maximum

```
min(z)
max(z)
```

c.) Find the indices of the minimum and maximum

```
min_index<-which(z==min(z))
min_index
max_index<-which(z==max(z))
max_index
```

d.) Compute the vector of the square values in z, store this in a vector called z2

```
z2<-(z^2)
z2
```

e.) Find out if any of the squared vectors are less than 4

```
any(z2<4)
```

f.) Find out of all the squared vectors are less than 22

```
all(z2<22)
```

g.) find out how many of the squared vectors are greater than 15 (hint, try summing the test statement)

```
sum(z2>15)
```

Problem 3

Read the help manual for rnorm to generate random normal values

a.) create a vector of 9 values, all of which are equal to 3

```
nine_values<-rnorm(9, mean = 3, sd = 0)
nine_values
```

b.) Put this into a 3 by 3 matrix

```
matrix=matrix(nine_values,3,3)
matrix
```

c.) Label the columns as “a”, “b”,“c”, and label the rows 1 to 3

```
colnames(matrix)=c("a","b","c")
rownames(matrix)=c("1","2","3")
```

d.) create a vector of 12 random normal values

```
twelve_values<-rnorm(12,mean=0,sd=1)
twelve_values
```

e.) put the random normal values into 4 x 3 matrix

```
matrix2=matrix(twelve_values,4,3)
matrix2
```

f.) Create a 3 x 3 identity matrix, you will need to look up how to do this in R, google search “R identity matrix”

```
ident_matrix<-diag(3)
ident_matrix
```

g.) Multiply the matrix from part b by the identify matrix created in part f

```
ident_matrix%*%matrix
matrix%*%ident_matrix
```

h.) find the row and column numbers of the largest value in the matrix created in step e

```
dim(matrix2)
max_matrix2<-max(matrix2)
max_indices<-which(matrix2 == max_matrix2, arr.ind = TRUE)
max_indices
```

i.) What is the average value of the 2nd column of the matrix from step e

```
matrix2[,2]
mean(matrix2[,2])
```

j.) Multiply the matrix from part e by the matrix from part b

```
matrix2%*%matrix
```

Problem 4

Explain the difference between vectors and lists in R

-how are each created? # vectors are typically created with the “c()” or “combine” function; this allows you to create a set of like values c(1,2,3,4,5). In swirl practice problems, they showed creation of vectors using seq and “:” as well. Lists are created with the “list()” function; this allows you to create a set of values of different data structes list1<- list(name = “katarina”, favorite_numbers =c(3,33,77). -What is different about the two #vectors are sets of values that are of the same data type while lists can hold multiple different objects and data structures. -Create a list of the values 1,2,3. Can you muliple the list by 2?

```
list1<-list(1,2,3)
list1
#list1*2
#you cannot multiply the list by 2
```

Problem 5

a.) Create a list containing the items “bob”, “sally”, c(1,2,3,4), seq(1,5,by=2), c(“jah”,“nah”,“hah”) in it

```
new_list<-list("bob","sally",c(1,2,3,4), seq(1,5, by=2), c("jah","nah","hah"))
new_list
```

b.) How would you reference the word “nah”?

```
new_list[[5]][[2]]
```

c.) How would you refernce the y in “sally”

```
y <-substr(new_list[[2]],5,5)
y
#I tried multiple times to do: y <- new_list[[2]][5] and I kept receiving NA as the outp
ut. I had to do some research to find str_sub which provides you with the opportunity to
extract or replace substrings in a character vector.
```

d.) Extract the vector 1,2,3,4 and store it in a new variable. Find the mean

```
new_vector<-new_list[[3]]
new_vector
mean(new_vector)
```

e.) Create a new list from the one in part 1 by removing “bob” and “sally” from the list

```
new_list2<-new_list[-c(1,2)]
new_list2
```

Problem 6

Use ?swiss to look at the built-in data set swiss

a.) What type of data type is this?

```
?swiss
class(swiss)
#swiss is a data frame.
```

b.) How many rows and columns does it have (use a function to determine this)

```
dim(swiss)
#47 rows, 6 columns
```

c.) Use the summary() and str() functions on this object, explain what they mean

```
summary(swiss)
str(swiss)
#summary() gives an overview of what the data contains. For this data frame it shows the
min, 1st Q, Median, Mean, 3rd Q, and Max of the 6 main variables. str() shows the struct
ure of the data frame, offering the details of what each category contains. For this exa
mple, it shows the data types of the 6 categories along with its contents.
```

e.) Print out the column of Agriculture values, but no other columns

```
agr<-swiss[,"Agriculture"]
agr
```

d.) Which region has the highest education value?

```
data(swiss)
edu<-swiss$Education
edu
max_edu<-max(edu)
max_index_edu<-which(swiss$Education == max_edu)
max_index_edu
region<-rownames(swiss)[max_index_edu]
region
#V. De Geneve has the highest education value.
```

e.) Which region has the lowest infant mortality?

```
data(swiss)
inf_mort<-swiss$Infant.Mortality
inf_mort
min_inf_mort<-min(inf_mort)
min_index_infm<-which(swiss$Infant.Mortality==min_inf_mort)
min_index_infm
region<-rownames(swiss)[min_index_infm]
region
#La Vallee has the lowest infant mortality.
```

Problem 7

a.) Use file.choose() to get the full name and file path of the example file sales.csv that is included in the Module 02 content

-Do not use file.choose() in the RMD file, run it in the console. Including file.choose() in an RMD will cause a knit errror

b.) set infile to the path you found and load the sales.csv file

```
# this is my path name for the download,  change it to the file name you get
# using file.choose()

infile="/Users/katarinadouglas-blake/Desktop/DSE5002/Module 02/sales.csv"

sales_df=read.csv(infile)
```

c.) Use head(), str() and summary() on this data set. How many orders are there?

```
head(sales_df)
str(sales_df)
summary(sales_df)
# there are 9994 orders
```

d.) Use the unique() function on the Segment column to find out the number of distinct segments there are in the data

```
dist_seg<-unique(sales_df$Segment)
dist_seg
#there are three distinct segments in the data.
```

e.) How many different customers are there?

```
diff_cust<-unique(sales_df$Customer.Name)
length(diff_cust)
# there are 793 different customers.
```

f.) Find the highest profit and the index of the highest profit

```
prof<-sales_df$Profit
prof[1:10]
max_prof<-max(prof)
max_prof
max_index_prof<-which(sales_df$Profit == max_prof)
max_index_prof
#the max profit is $8,399.976 at an index of [6827]
```

g.) What was the name of the customer with the highest profit order?

```
prof_cust<-(sales_df$Customer.Name[max_prof])
prof_cust
#the name of the customer with the highest profit order was Sharelle Roach.
```

h.) Set the Segment variable to be a factor

```
sales_df$Segment<-factor(sales_df$Segment)
sales_df$Segment[1:10]
```

i.) use the lubridate package to set the Order.Data and Ship.Date varialbe to be lubridate style date variables

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
sales_df$Order.Date = trimws(as.character(sales_df$Order.Date))
sales_df$Ship.Date = trimws(as.character(sales_df$Ship.Date))
sales_df$Order.Date.l=mdy(sales_df$Order.Date)
sales_df$Ship.Date.l=mdy(sales_df$Ship.Date)
```

```
head(sales_df)
```

j.) compute the shipping delay from each order, using order date and ship data what were the longest and shortest shipping delays?

```
sales_df$shippingdelay<-as.numeric(sales_df$Ship.Date.l - sales_df$Order.Date.l)
longest_delay<-max(sales_df$shippingdelay, na.rm = TRUE)
                   cat("The longest shipping delay was", longest_delay, "days \n")
shortest_delay<-min(sales_df$shippingdelay, na.rm = TRUE)
                    cat("The shortest shipping delay was", shortest_delay, "days \n")
longest_delay
shortest_delay
#The longest shipping delay was 7 days.The shortest shipping delay was 0 days.
```

k.) Were orders received on Saturday and Sunday? Were any shipped on Sunday?

```
wknd_order<-any(wday(sales_df$Order.Date.l) %in% c(6&7))
wknd_order
sunday_shipment<-any(wday(sales_df$Ship.Date.l)==7)
sunday_shipment
```

l.) Convert the Customer.Name to all lower case

```
sales_df$Customer.Name<-tolower(sales_df$Customer.Name)
sales_df$Customer.Name
```

m.) Search the customer names and determine how many contain the name John

```
library(stringr)
sum(str_detect(sales_df$Customer.Name, regex("John",ignore_case = TRUE)))
# 138 customer names contain the name John.
```

n.) How many times is the word “table” in the product names -careful, str_detect() might find “table” in “portable”, also be careful with case

```
count_Table<-sum(str_count(sales_df$Product.Name, "(?i)\\btable\\b"))
count_Table
# the word "table" is in the product names 230 times.
```

o.) Use tapply to find the mean profit value, grouped by State

```
mean_profit<- tapply(sales_df$Profit, sales_df$State, mean)
mean_profit
```

```
  https://www.statology.org/tapply-r/
```

p.) use tapply to find the median profit value, grouped by Segment

```
median_prof <- tapply(sales_df$Profit, sales_df$Segment, median)
median_prof
```

q.) use table to get the number of orders per segment

```
seg_orders<-table(sales_df$Segment)
seg_orders
#Consumers had 5191 orders, Corporate segment had 3020 order, and Home Office had 1783 o
rders.
```