**Assignment - 1 To 4**

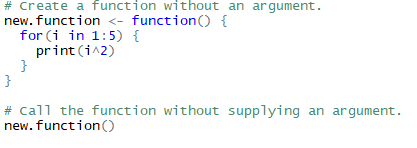
**Task 1:**

**1. How many ways are there to call a function in R?**

*There are 3 ways to call a function in R.*

1. ***Calling a Function without an Argument:***

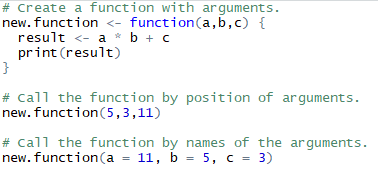
***Example:***



1. ***Calling a Function with Argument Values (by position and by name):***

*The arguments to a function call can be supplied in the same sequence as defined in the function or they can be supplied in a different sequence but assigned to the names of the arguments.*

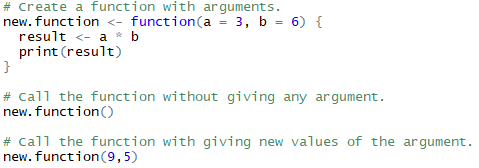
***Example:***



1. ***Calling a Function with Default Argument:***

*We can define the value of the arguments in the function definition and call the function without supplying any argument to get the default result. But we can also call such functions by supplying new values of the argument and get non default result.*

***Example:***



**2. What is the Recycling of elements in a vector?**

**Solution:** *Recycling of elements in a vector in R is, how R automatically recycles, or repeats, elements of the shorter Vector when applying an operation to two vectors that requires them to be the same length.*

*R automatically recycles, or repeats, elements of the shorter one, until it is long enough to match the longer Vector.*

**3. Give an example of recycling of elements.**

***Example:***

*Suppose we have two Vectors c (1,2,4), c (6,0,9,10,13), where the first one is shorter with only 3 elements. Now if we sum these two, we will get a warning message as follows.*

*> c(1,2,4) + c(6,0,9,10,13)  
[1]  7  2 13 11 15*

*Warning message:  
In c(1, 2, 4) + c(6, 0, 9, 10, 13) :  longer object length is not a multiple of shorter object length  
  
Here R , Sum those Vectors by Recycling or repeating the elements in shorter one, until it is long enough to match the longer one as follows..  
  
> c(1,2,4,1,2) + c(6,0,9,10,13)  
[1]  7  2 13 11 15*

**Task 2:**

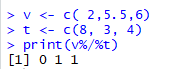
**1. What should be the output of the following Script?**

***v <- c (2,5.5,6)***

***t <- c (8, 3, 4)***

***print (v% / %t)***

***Output:***



**2. You have 25 excel files with names as xx\_1.xlsx, xx\_2.xlsx, ........xx\_25.xlsx in a dir.**

**Write a program to extract the contents of each excel sheet and make it one df.**

**Solution1:**

library(xlsx)

files=list.files(pattern="\_1.csv")

df\_total=data.frame()

for(i in 1:length(files))

{

filename=files[i]

data=read.csv(file = filename,header = T)

df\_total=rbind(df\_total,data)

}

df\_total

***Output:***



**or**

**Solution2:**

files=list.files(pattern="\_1.csv")

for(i in 1:length(files))

{filename=files[i]

data=read.csv(file = filename,header = T)

assign(x = filename,value = data)}

data

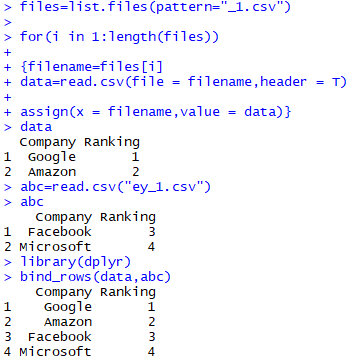
abc=read.csv("ey\_1.csv")

abc

library(dplyr)

bind\_rows(data,abc)

***Output:***



**Task 3:**

**1. Create an m x n matrix with replicate (m, rnorm(n)) with m=10 column vectors of n=10 elements each, constructed with rnorm(n), which creates random normal numbers.**

**Then we transform it into a dataframe (thus 10 observations of 10 variables) and perform an algebraic operation on each element using a nested for loop: at each iteration, every element referred by the two indexes is incremented by a sinusoidal function, compare the vectorized and non-vectorized form of creating the solution and report the system time differences.**

**Solution:**

*m=10; n=10;*

*mymat<-replicate(m, rnorm(n)) # create matrix of normal random numbers*

*mydframe=data.frame(mymat)# transform into data frame*

*for (i in 1:m) {*

*for (j in 1:n) {*

*mydframe[i,j]<-mydframe[i,j] + 10\*sin(0.75\*pi)*

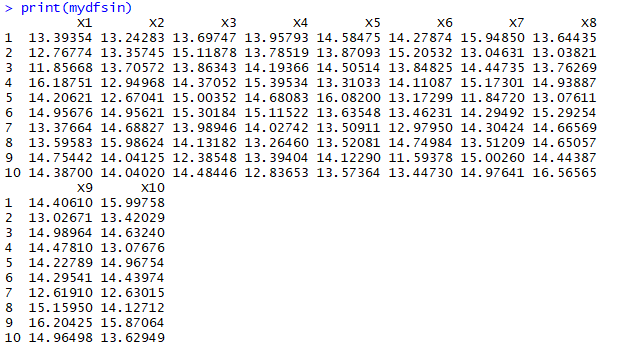
*}*

*}*

*mydfsin = mydframe*

*print(mydfsin)*

***Output:***



**System time differences:**

**Solution:**

*TS = system.time(for (i in 1:m) {*

*for (j in 1:n) {*

*mydframe[i,j]<-mydframe[i,j] + 10\*sin(0.75\*pi)*

*}*

*})*

*print(TS[3])*

***Output:***



**Task 4:**

**1. Define matrix mymat by replicating the sequence 1:5 for 4 times and transforming into a matrix, sum over rows and columns.**

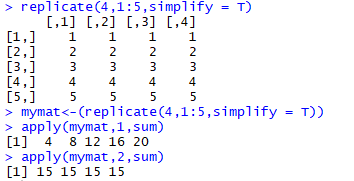
*replicate(4,1:5,simplify = T)*

*mymat<-(replicate(4,1:5,simplify = T))*

*apply(mymat,1,sum)*

*apply(mymat,2,sum)*

***Output:***



**Task 5:**

**1. States = rownames(USArrests)**

***Get states names with ‘w’.***

***Get states names with ‘W’.***

**Solution:**

**Get states names with ‘w’.**

*States = rownames(USArrests)*

*rownames(USArrests)*

*grep("w",rownames(USArrests))*

*States\_with\_w <-grep("w",States)*

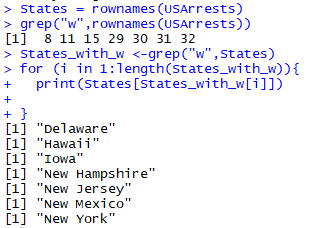
*for (i in 1:length(States\_with\_w))*

*{*

*print(States[States\_with\_w[i]])*

*}*

***Output:***



**Get states names with ‘W’.**

*States = rownames(USArrests)*

*rownames(USArrests)*

*grep("W",rownames(USArrests))*

*States\_with\_W <-grep("W",States)*

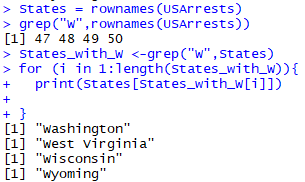
*for (i in 1:length(States\_with\_W))*

*{*

*print(States[States\_with\_W[i]])*

*}*

***Output:***



**2. Prepare a Histogram of the number of characters in each US state.**

**Solution:**

*Characters\_in\_each\_state <- c(0)*

*for(i in 1:50){*

*temp <- States[i]*

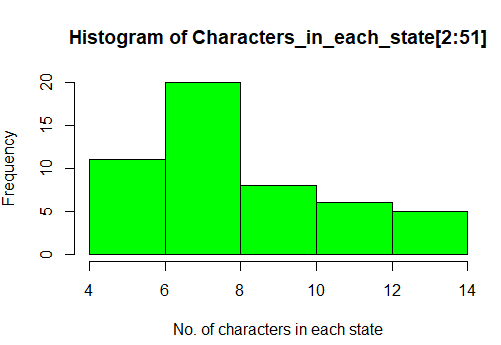
*len <- nchar(temp)*

*Characters\_in\_each\_state <- c(Characters\_in\_each\_state,len)*

*}*

*hist(Characters\_in\_each\_state[2:51],xlab="No. of characters in each state",col = "* *green")*

***Output:***



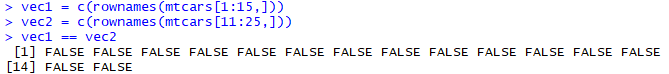
**Task 6:**

**1. Test whether two vectors are exactly equal (element by element).**

***vec1 = c(rownames(mtcars[1:15,]))***

***vec2 = c(rownames(mtcars[11:25,])***

**Solution1:**



**Solution2:**



**2. Sort the character vector in ascending order and descending order.**

***vec1 = c(rownames(mtcars[1:15,]))***

***vec2 = c(rownames(mtcars[11:25,]))***

**Solution:**

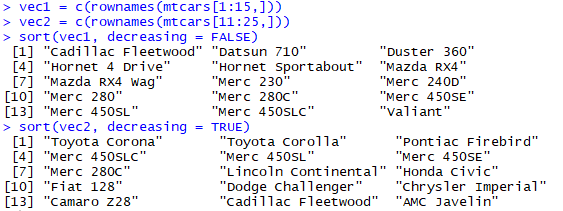
*vec1 = c(rownames(mtcars[1:15,]))*

*vec2 = c(rownames(mtcars[11:25,]))*

*sort(vec1, decreasing = FALSE)*

*sort(vec2, decreasing = TRUE)*

***Output:***



**3. What is the major difference between str() and paste() show an example?**

**Solution:**

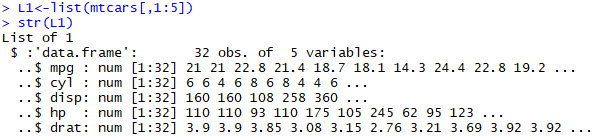
***paste():*** *This is a function which is used to* ***concatenate the strings****.*

***Example:***



***str():*** *This is a function which converts* ***data frame*** *into* ***list*** *format.*

***Example:***



**4. Introduce a separator when concatenating the strings.**

**Solution:**



**Task 7:**

**1. Import the Titanic Dataset from the link => Titanic Data Set.**

**Perform the following:**

***a. Is there any difference in fares by a different class of tickets?***

***Note*** *- Show a boxplot displaying the distribution of fares by class****.***

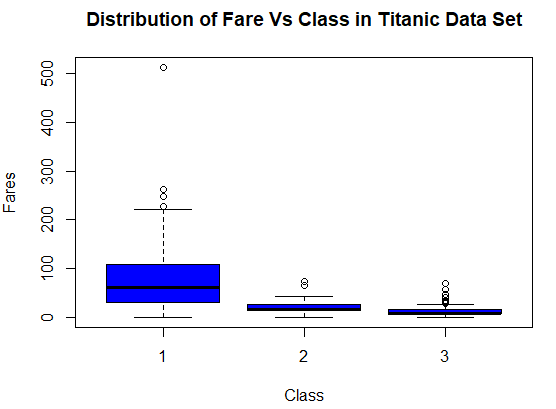
**Solution:**

*Titanic1 = read.csv("Titanic.csv")*

*boxplot(fare ~ pclass, data = Titanic1, xlab = "Class",*

*ylab = "Fares", main = "Distribution of Fare Vs Class in Titanic Data Set", col="Blue")*

***Output:***



***b. Is there any association with Passenger class and gender?***

***Note –*** *Show a stacked bar chart*

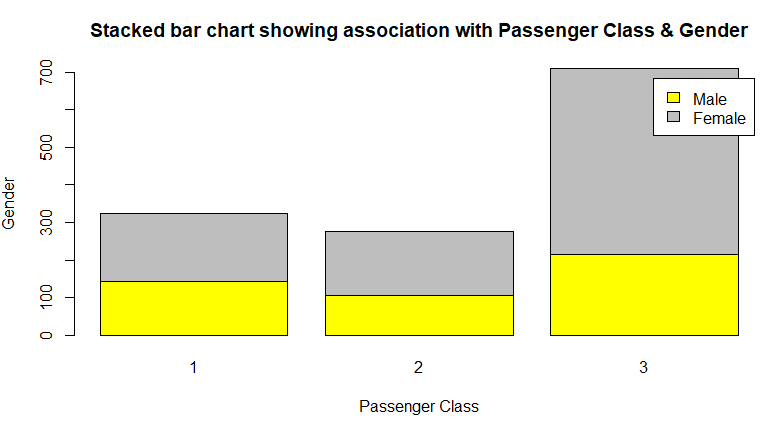
**Solution:**

*Titanic1 = read.csv("Titanic.csv")*

*counts <- table(Titanic1$sex, Titanic1$pclass)*

*barplot(counts, main="Stacked bar chart showing association with Passenger Class & Gender", xlab="Passenger Class",ylab = "Gender",col=c("grey","yellow"),legend.text = c("Female", "Male"))*

***Output:***



**Task 8:**

**1. Create a box and whisker plot by class and count using Titanic dataset.**

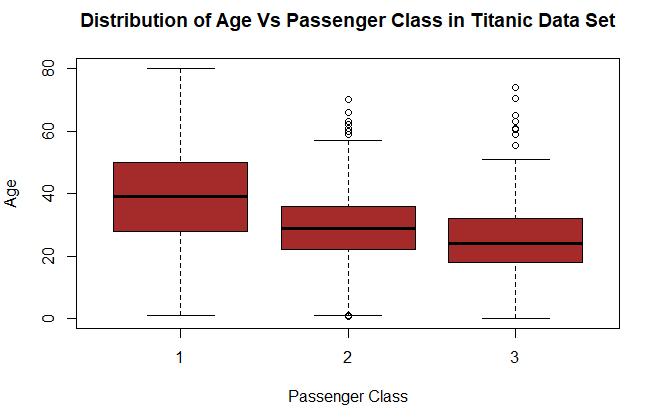
**Solution:**

*Titanic1 = read.csv("Titanic.csv")*

*boxplot(age ~ pclass, data = Titanic1, xlab = "Passenger Class",*

*ylab = "Age", main = "Distribution of Age Vs Passenger Class in Titanic Data Set", col="Brown")*

***Output:***



**Task 9:**

**1. A recent national study showed that approximately 44.7% of college students have used Wikipedia as a source in at least one of their term papers. Let X equal the number of students in a random sample of size n = 31 who have used Wikipedia as a source.**

**Perform the below functions**

***a. Find the probability that X is equal to 17.***

**Solution:**

*dbinom(17, size = 31, prob = 0.447)*

***Output:***



***b. Find the probability that X is at most 13***

**Solution:**

*pbinom(13, size = 31, prob = 0.447)*

***Output:***



***c. Find the probability that X is bigger than 11.***

**Solution:**

*pbinom(11, size = 31, prob = 0.447, lower.tail = FALSE)*

***Output:***



***d. Find the probability that X is at least 15.***

**Solution:**

*pbinom(14, size = 31, prob = 0.447, lower.tail = FALSE)*

***Output:***



***e. Find the probability that X is between 16 and 19, inclusive***

**Solution-1:**

*sum(dbinom(16:19, size = 31, prob = 0.447))*

***Output:***



**Or**

**Solution-2:**

*diff(pbinom(c(19, 15), size = 31, prob = 0.447, lower.tail = FALSE))*

***Output:***



**6. Expected Output**

**Solution document/report shall be in PDF format. Submitted in GitHub.**

**The PDF Doc should have Code with its subsequent result screenshot.**