DATA ANALYTICS WITH R, EXCEL AND TABLEU

Assignment 1 (Session 1-5)

Name – Abhilash Singh

Task 1

1. How many ways are there to call a function in R?
Answer: There are mainly three ways to call a function in R.

a. Call(name,..)
b. Is.call(x)
c. As.call(x)

2. What is the Recycling of elements in a vector?
Answer: When applying two vectors in R, it requires to be same length. R automatically repeats elements of the shorter one till one vector become long enough to match the longer vector. This is called Recycling of elements in vector.

3. Give an example of recycling of elements.
Answer:
>c(2,4,6)+c(5,6,7,8,9)
>c(7,10,16,18,13)

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)

Answer:

```
> v <- c( 2,5.5,6)
> t <- c(8, 3, 4)
> print(v%/%t)
[1] 0 1 1
```

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.
Answer:
setwd("c:/R/mergeme") 0r specific file path name files=list.files(pattern=".xlsx") for(i in 1:length(files)) {filename=files[i] data=read.xlsx(file = filename,header = T) assign(x = filename,value = data)} #Suppose the columns are the same for each file, #you can bind them together in one dataframe with bind_rows from dplyr: library(dplyr) #one more option is as follows df<-lapply(files, read.xlsx) %>% bind_rows()

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.**

Answer:

```
#Vectorized
form
                set.seed(42)


                #create matrix
                mat_1<- replicate(10,rnorm(10))


                #transform into data frame
                df_1= data.frame(mat_1)
                df_1<- df_1 + 10*sin(0.75*pi)


                #non-vectorized form
                set.seed(42)
                #create matrix
                mat_1<- replicate(10,rnorm(10))
                #transform into data frame
                df_1= data.frame(mat_1)


                for(i in 1:10){
                  for(j in 1:10){
                    df_1[i,j]<- df_1[i,j] + 10*sin(0.75*pi)
                    print(df_1)
                    }
                }


                #time difference


                system.time(
                      df_1[i,j]<- df_1[i,j] + 10*sin(0.75*pi)
                )


                system.time(
                  for(i in 1:10){
                  for(j in 1:10){
                    df_1[i,j]<- df_1[i,j] + 10*sin(0.75*pi)
                   }
                  }
                 )
```

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.**

Answer:

mymat <- matrix(rep(1:5, 4), ncol = 4)

mymat [,1] [,2] [,3] [,4] [1,] 1 1 1 1 [2,] 2 2 2 2 [3,] 3 3 3 3 [4,] 4 4 4 4 [5,] 5 5 5 5
apply(mymat, 1, sum) [1] 4 8 12 16 20

apply(mymat, 2, sum) [1] 15 15 15 15

Task 5

1. States = rownames(US Arrests)
Get states names with 'w'.
Get states names with 'W'.

Answer:

```
States =
rownames(USArrests)
                    rownames(USArrests)
                    #for w
                    grep("w",rownames(USArrests))
                    x<-grep("w",States)
                    for (i in 1:length(x)){
                      print(States[x[i]])
                    }
                    #for W
                    grep("W",rownames(USArrests))
                    y<-grep("W",States)
                    for (i in 1:length(y)){
                      print(States[y[i]])
                    }
```

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

```
answer <- c(0)
for(i in 1:50){
  temp <- States[i]
  len <- nchar(temp)
  answer <- c(answer,len)

}
# As 1st element we have added is 0
# which we do not want in output so we are getting rid of it
hist(answer[2:51],xlab="No. of characters in each state",col = "red")
```

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,]))

Answer: No, both vectors are not exactly equal element by element. Only number of elements are equal but not value. Output are:

a.
```
>vec1 = c(rownames(mtcars[1:15,]))
> vec1
 [1] "Mazda RX4"          "Mazda RX4 Wag"      "Datsun 710"
"Hornet 4 Drive"     "Hornet Sportabout"  "Valiant"
 [7] "Duster 360"         "Merc 240D"          "Merc 230"
"Merc 280"           "Merc 280C"          "Merc 450SE"
[13] "Merc 450SL"         "Merc 450SLC"        "Cadillac Fleetwood"
```

b.
```
>vec2 = c(rownames(mtcars[11:25,]))
> vec2
 [1] "Merc 280C"           "Merc 450SE"          "Merc 450SL"
"Merc 450SLC"         "Cadillac Fleetwood"  "Lincoln Continental"
 [7] "Chrysler Imperial"   "Fiat 128"            "Honda Civic"
"Toyota Corolla"      "Toyota Corona"       "Dodge Challenger"
[13] "AMC Javelin"         "Camaro Z28"          "Pontiac Firebird"
```

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

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

Ascending order:
```
print(sort(vec1))
 [1] "Cadillac Fleetwood" "Datsun 710"         "Duster 360"         "Horne
t 4 Drive"     "Hornet Sportabout"  "Mazda RX4"
 [7] "Mazda RX4 Wag"      "Merc 230"           "Merc 240D"          "Merc
280"           "Merc 280C"          "Merc 450SE"
[13] "Merc 450SL"         "Merc 450SLC"        "Valiant"
```

Descending Order:
```
> print(sort(vec1, decreasing =TRUE))
 [1] "Valiant"            "Merc 450SLC"        "Merc 450SL"         "Merc
450SE"         "Merc 280C"          "Merc 280"
 [7] "Merc 240D"          "Merc 230"           "Mazda RX4 Wag"      "Mazda
RX4"          "Hornet Sportabout"  "Hornet 4 Drive"
[13] "Duster 360"         "Datsun 710"         "Cadillac Fleetwood"
```

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

Ascending Order:
```
>print(sort(vec2))
 [1] "AMC Javelin"         "Cadillac Fleetwood"  "Camaro Z28"          "Ch
rysler Imperial"   "Dodge Challenger"    "Fiat 128"
 [7] "Honda Civic"         "Lincoln Continental" "Merc 280C"           "Me
rc 450SE"          "Merc 450SL"          "Merc 450SLC"
[13] "Pontiac Firebird"    "Toyota Corolla"      "Toyota Corona"
```

Descending Order:
```
> print(sort(vec2, decreasing =TRUE))
```

```
 [1] "Toyota Corona"       "Toyota Corolla"      "Pontiac Firebird"    "Me
rc 450SLC"         "Merc 450SL"          "Merc 450SE"
 [7] "Merc 280C"           "Lincoln Continental" "Honda Civic"         "Fi
at 128"            "Dodge Challenger"    "Chrysler Imperial"
[13] "Camaro Z28"          "Cadillac Fleetwood"  "AMC Javelin"
```

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

`paste0(..., collapse = NULL)`is a wrapper for `paste(..., sep = "", collapse = NULL)`, which means there is no separator. In other words, with `paste0()` you can not apply some sort of separator, while you do have that option with `paste()`, whereas a single space is the default. `str_c(..., sep = "", collapse = NULL)` is equivalent to `paste()`, which means you do have the option to customize your desired separator. The difference is for `str_c()` the default is no separator, so it acts just like `paste0()` as a default. `Paste()` and `paste0()` are both functions from the base package, whereas `str_c()` comes from the stringr package.

4. Introduce a separator when concatenating the strings.

Answer:

```
# Example R program to concatenate two strings

str1 = 'Hello'
str2 = 'Sir!'

# concatenate two strings using paste function
result = paste(str1,str2)
```

```
[1] "Hello Sir!"
```