THE KNOWLEDGE UNIVERSITY

Faculty of Computing

Year 2 Semester 1 (2025)

IT2120 - Probability and Statistics Lab Sheet 02

(Introduction to R)

Before starting the lab sheet, you need to create a folder in your desktop and save all your working inside the folder. Set the working directory to that folder using the following command:

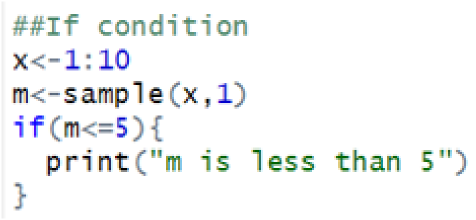
setwd(”paste the path of the folder”)
Eg:- setwd(”E:2018PS_ProrataLab Sheets_PS_ProrataLab 02”)

Try out the following commands which we can use to create a sample of data and see the difference.

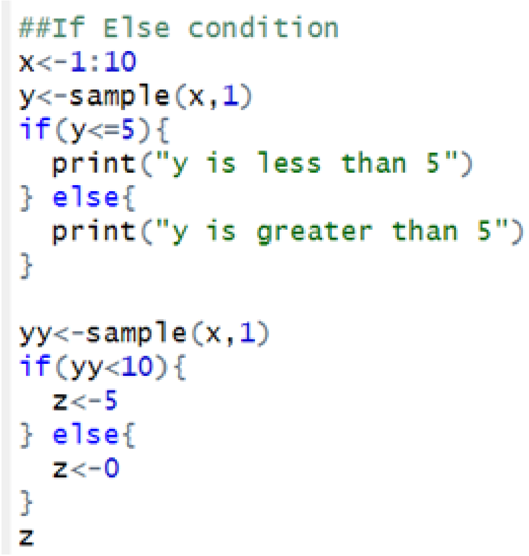
```
print(sample(1:3))
print(sample(1:3,size=3,replace=FALSE))
print(sample(c(2,5,3),size=4,replace=TRUE))
print(sample(c(2,5,3),size=4,replace=FALSE))
print(sample(1:2,size=10,prob=c(0.3,0.7),replace=TRUE))
sample(1:3)
sample(1:3,size=3,replace=FALSE)
sample(c(2,5,3),size=4,replace=TRUE)
sample(c(2,5,3),size=4,replace=FALSE)
sample(1:2,size=10,prob=c(0.3,0.7),replace=TRUE)
```

Control Statements

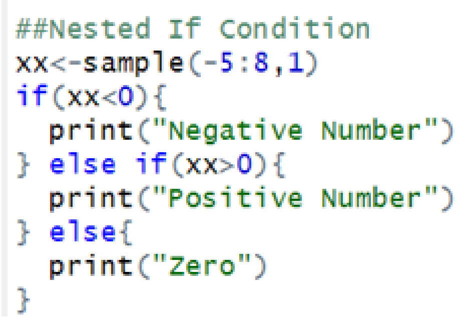
Conditional statements and loops exist in R; the same as in other programming languages. Given below are the syntax for conditional statements (If, If else & Nested if) and loops (For loop & while loop). Try them by your self.



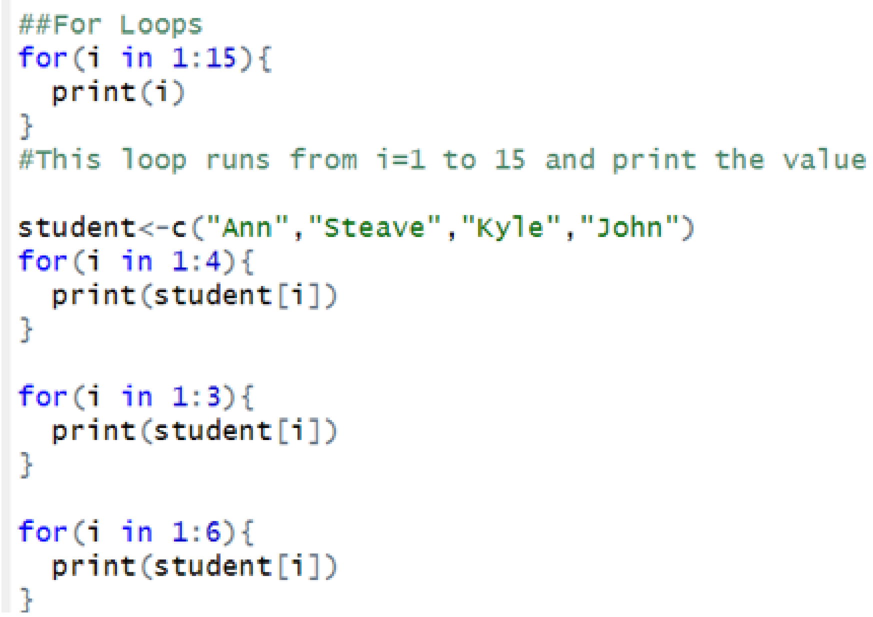
```r
##If condition
x<-1:10
m<-sample(x,1)
if(m<=5){
  print("m is less than 5")
}
```



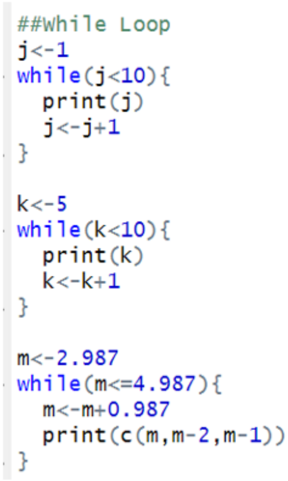
```r
##If Else condition
x<-1:10
y<-sample(x,1)
if(y<=5){
  print("y is less than 5")
} else{
  print("y is greater than 5")
}

yy<-sample(x,1)
if(yy<10){
  z<-5
} else{
  z<-0
}
z
```



```r
##Nested If Condition
xx<-sample(-5:8,1)
if(xx<0){
  print("Negative Number")
} else if(xx>0){
  print("Positive Number")
} else{
  print("Zero")
}
```



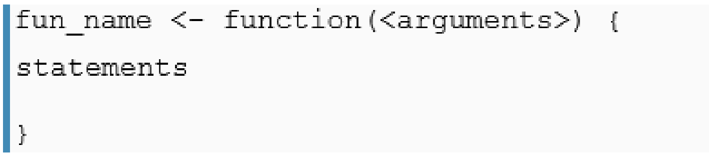
```r
##For Loops
for(i in 1:15){
  print(i)
}
#This loop runs from i=1 to 15 and print the value

student<-c("Ann","Steave","Kyle","John")
for(i in 1:4){
  print(student[i])
}

for(i in 1:3){
  print(student[i])
}

for(i in 1:6){
  print(student[i])
}
```



```
##While Loop
j<-1
while(j<10){
  print(j)
  j<-j+1
}

k<-5
while(k<10){
  print(k)
  k<-k+1
}

m<-2.987
while(m<=4.987){
  m<-m+0.987
  print(c(m,m-2,m-1))
}
```

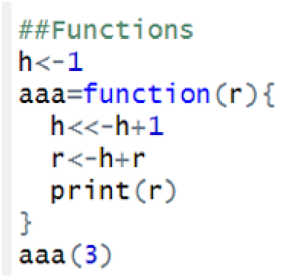
Functions

Functions are created using the function() directive and are stored as R objects just like anything else. They are R objects of class “function”.



```
fun_name <- function(<arguments>) {
statements

}
```

When specifying arguments, the function can be defined with default values for the arguments. Try out the following example.



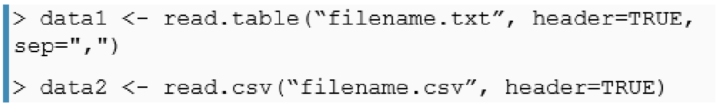
```
##Functions
h<-1
aaa=function(r){
  h<<-h+1
  r<-h+r
  print(r)
}
aaa(3)
```

Importing and Exporting

R can read files on your machine and create data files and graphics. Paths to these files are computed relative to the **working directory.** Paths are specified in the format appropriate for the machine.

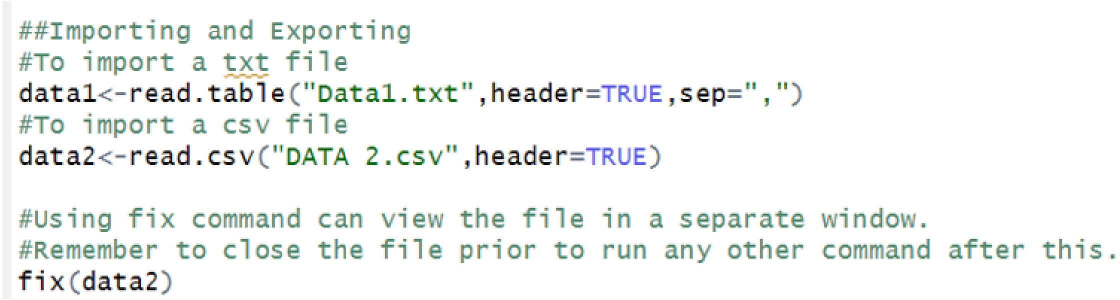
R supports basic data types when importing data and other file types can be imported using dedicated packages (e.g.: xlsx package for importing Excel 2010 onwards).

read.table() can be used to import data from a basic file type (text file) and read.csv() for importing data from CSV (comma separated values) files. The dataset will be imported as a data frame. The following 2 commands do the same job.

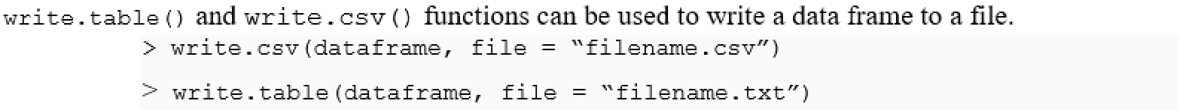


```
> data1 <- read.table("filename.txt", header=TRUE,
sep=",")
> data2 <- read.csv("filename.csv", header=TRUE)
```

Try to import “Data1” file and “DATA 2” file using the following commands.



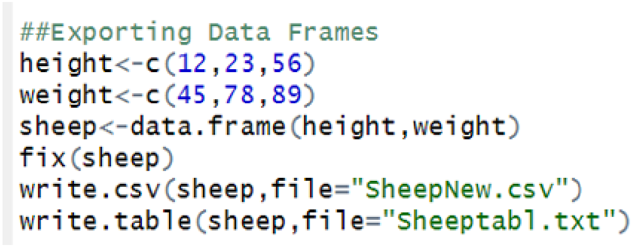
```
##Importing and Exporting
#To import a txt file
data1<-read.table("Data1.txt",header=TRUE,sep=",")
#To import a csv file
data2<-read.csv("DATA 2.csv",header=TRUE)

#Using fix command can view the file in a separate window.
#Remember to close the file prior to run any other command after this.
fix(data2)
```



write.table() and write.csv() functions can be used to write a data frame to a file.

```
> write.csv(dataframe, file = "filename.csv")
> write.table(dataframe, file = "filename.txt")
```

Create two variables called “Height” and “Weight” and create a data frame called “Sheep” with those two variables. Export the data frame as a csv file and text file.



```
##Exporting Data Frames
height<-c(12,23,56)
weight<-c(45,78,89)
sheep<-data.frame(height,weight)
fix(sheep)
write.csv(sheep,file="SheepNew.csv")
write.table(sheep,file="Sheeptabl.txt")
```

Exercise

Instructions: Create a folder in your desktop with your registration number (Eg: “IT. . . . . . .”). You need to save the R script file and take screenshots of the command prompt with answers and save it in a word document inside the folder. Save both R script file and word document with your registration number (Eg: “IT. . . . . . .”). After you finish the exercise, zip the folder and upload the zip file to the submission link.

1. Without using R, determine the result of the following computation.
   x ¡- c(1, 2, 3)
   x[1] / x[2]ˆ3 - 1 + 2 * x[3] - x[2 - 1]

2. Consider vector 1: 15. Write **R** command that determines how many elements in the vector are exactly divisible by 3.

3. Write a loop structure to scan through an integer vector to determine the index of the maximum value.

4. Do the question 03 without using a loop.