KAUNAS UNIVERSITY OF TECHNOLOGY

Faculty of Mathematics
and Natural Sciences

Theory of Probability and Statistics

Laboratories work report
Labs 1-6

Name:Alexandros Veremis
Submission date: 2020-06-10

(Erasmus- Spring 2020)

KAUNAS

LAB1

1) Write the SUM and the PRODUCT rules.

2) At the restaurant the waiter informs that you have (a) two choices for appetizers: Ceasar or Greek salads; (b) three choices for main course: meat, fish or vegetable dish; and (c) two for dessert: ice cream or cake.

a) How many possible choices do you have to pick one dish?

b) How many possible choices do you have for your complete meal? Illustrate possible meals by a tree diagram.

3) Write the formula for the number of PERMUTATIONS. Find corresponding R's function and explain its usage.

4) Write the formula for the number of ARRANGEMENTS of m outcomes of n possibilities. Find corresponding R's function and explain its usage.

5) Write the formula for the number of COMBINATIONS of m outcomes of n possibilities. Find corresponding R's function and explain its usage.

6) Explain the difference between COMBINATIONS and ARRANGEMENTS.

Solutions:

1)

Product rule: if you can choose object A in n ways
AND object B in m ways → then you can choose object A or B in
n*m ways

Sum rule: if you can choose object A in n ways
OR object B in m ways then you can choose object A or B in
n+m ways

2)

a)needs the Sum rule because we choose one object of the whole 2+3+2=7.
We don't do any combinations
b)needs the product rule because we now choose a whole meal which is a combination of 3 different dishes :2*3*2=12

3)

```
install.packages('gtools)
#load library
library(gtools)
```

Permutations ,arrangements and combinations can be found in gtools!

```
#Permutations : P(n)=n!
#or factorial(n)
```

```
fact <-function(x){
  return(factorial(x))
}
fact(6)

x<-factorial(5)
x
```

factorial is the function that returns the Factorial of the variable that we call it with and it helps us with permutation calculations ,if we don’t want to call permutations from gtools
#nrow(permutations(n=3,r=3,v=c(1,2,3),repeats.allowed=FALSE))
The permutation function from gtools
we have to make sure that n=r
#A permutation is an ordered combination

4)Arrangements:

```
A <-function(m, n){ #m out of n
  a<-factorial(n)/factorial(n-m)
  return (a)
}
#nrow(permutations(n=4,r=2,v=c(1,2,3,4),repeats.allowed=FALSE))
```

In permutations we see that n=r while in arrangements n<>r
Arrangements are a way to calculate the total outcomes of an event where order of the outcomes DOES matter

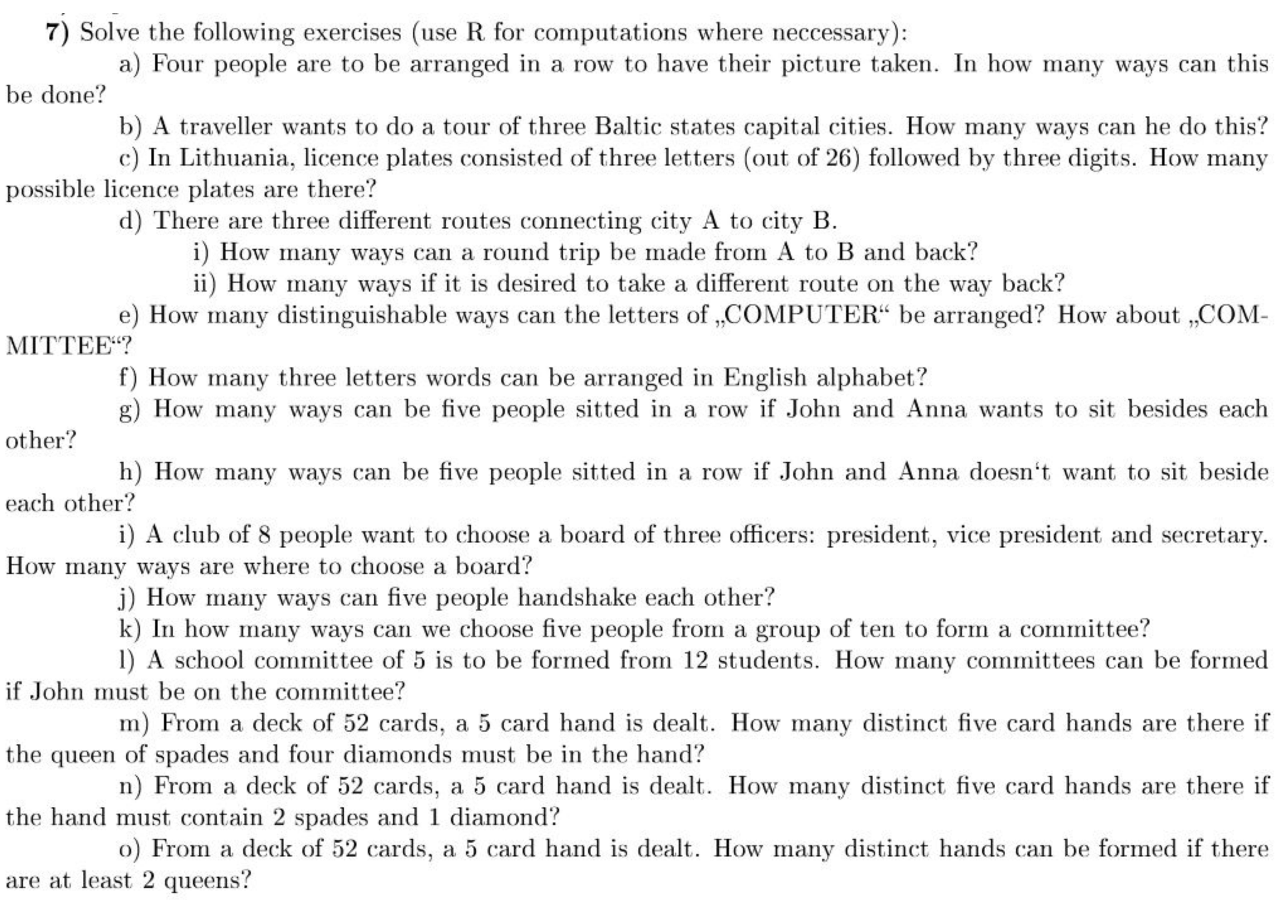
5) Combinations:

```
C <-function(m, n){ #m out of n
  a<-factorial(n)/(factorial(n-m)*factorial(m))
  return (a)
}
#nrow(combinations(3,2))
```

Combinations are a way to calculate the total outcomes of an event where order of the outcomes does not matter

6)

Arrangements -order is important ,while in combinations order doesn't play any role.



7) Solve the following exercises (use R for computations where neccessary):

a) Four people are to be arranged in a row to have their picture taken. In how many ways can this be done?

b) A traveller wants to do a tour of three Baltic states capital cities. How many ways can he do this?

c) In Lithuania, licence plates consisted of three letters (out of 26) followed by three digits. How many possible licence plates are there?

d) There are three different routes connecting city A to city B.

i) How many ways can a round trip be made from A to B and back?

ii) How many ways if it is desired to take a different route on the way back?

e) How many distinguishable ways can the letters of „COMPUTER“ be arranged? How about „COMMITTEE“?

f) How many three letters words can be arranged in English alphabet?

g) How many ways can be five people sitted in a row if John and Anna wants to sit besides each other?

h) How many ways can be five people sitted in a row if John and Anna doesn‘t want to sit beside each other?

i) A club of 8 people want to choose a board of three officers: president, vice president and secretary. How many ways are where to choose a board?

j) How many ways can five people handshake each other?

k) In how many ways can we choose five people from a group of ten to form a committee?

l) A school committee of 5 is to be formed from 12 students. How many committees can be formed if John must be on the committee?

m) From a deck of 52 cards, a 5 card hand is dealt. How many distinct five card hands are there if the queen of spades and four diamonds must be in the hand?

n) From a deck of 52 cards, a 5 card hand is dealt. How many distinct five card hands are there if the hand must contain 2 spades and 1 diamond?

o) From a deck of 52 cards, a 5 card hand is dealt. How many distinct hands can be formed if there are at least 2 queens?

7)

a)	24	
b)	6	
c)	17576000	
d)	i)	9
	ii)	6
e)	COMPUTER:40320	
	COMMITEE:5040	
f)	17576	
g)	48	
h)	72	
i)	336	
j)	10	
k)	252	
l)	330	
m)	19600	
n)	329550	
o)	117600	

Program Code:

```
#this is a comment lw1

#P1

#Product rule:  if you can choose object A in n ways
#AND object  B in m ways  then u can choose object A or B
#n*m ways

#Sum rule:  if you can choose object A in n ways
#or object  B in m ways  then u can choose object A or B
#n+m ways

#P2
# a)2+3+2=7
#b) 2*3*2=12

#P3
#P(n)=n!
factorial(5)

fact <-function(x){
  return(factorial(x))
}
fact(6)

factorial(5)->x
x
#factorial is the function and returns the
#factorial of the variable that we call it with
#and it helps us with permutation calculations

#P4
A <-function(m, n){ #m out of n
  a<-factorial(n)/factorial(n-m)
  return (a)
}
A(2,3)

#P5
C <-function(m, n){ #m out of n
  a<-factorial(n)/(factorial(n-m)*factorial(m))
  return (a)
}
C(2,3)
choose(3,2)
library("utils")
```

```
combn(c("a","b","c"),2)
obje <- c("a","b","c")

combn(obje,2)
install.packages("gtools")
library("gtools")

combinations(3,2)
x<-12.21
y<-'abc'
cc<-c(1,2,78)
cc
class(cc)
M<-matrix(c(1,7,45,-1),ncol=2,byrow=TRUE)
l<-list("1",1,c(1,5))
l
ll<-as.list(cc)
ll

#P6
#Arrangements -order is important
#while in combinations order doesn't play any role
#P7
#a)
x<-factorial(4)
st<-")Four people want to take a picture in a row ==> that many ways"
message(st)
message(x)
#because it is like arrangement 4 out of 4 ==> permutation of 4
#b)
factorial(3)
#same as a) ==> permutation of 3
#c)
x<-26^3 * 10^3
x
#product rule because we have to take all letters and numbers
#d)
#i) product rule again taking all the ways
3*3
#ii) product rule taking all the ways except for one
3*2

#e)
factorial(8) #permutaTion of all the letters of COMPUTER
factorial(8)/(2*2*2) #pemutation of all the letters of COMMITTEE
#dividing by those who exist more than one time
#f)
26^3
```

```
#product rule like in Lithuanian license plates
#Dataframes
S<-seq(1,11,2)
M<-matrix(S,ncol=2)
N<-c("john","stuart","sarah")
N2<-c(0.1,0.2,0.3)
cbind(M,N2)
colnames(cbind(M,N2))

#apo edw xekinane ta dika moy
#g)
factorial(4)*factorial(2)
#h)
factorial(5)-(factorial(4)*factorial(2))
#the ways they can all 5 sit together minus the ways John and Anna are next to each other
#i)
A(3,8)
#j)
C(2,5)
#order not important and pairs of 2 from 5 people
#k)
C(5,10)
#order not important
#l)
C(4,11)
#m)
C(3,50)
#n)
C(2,13)*C(1,13)*C(2,26)
#because it doesnt say at least, so I thought exactly 2 and 1.
#o)
C(2,4)*C(3,50)
#because it says at least 2 queens so we use 50
```

LAB 2

1) Consult variuos sources (R ducumentation, search engines, your lecturer) and shortly describe:
  a) R and RStudio.
  b) Data formats, basic operators, basic functions, basic functions for manipulationg with data.
  c) Possibilities of base graphics system. Other ways/packages to plot data?

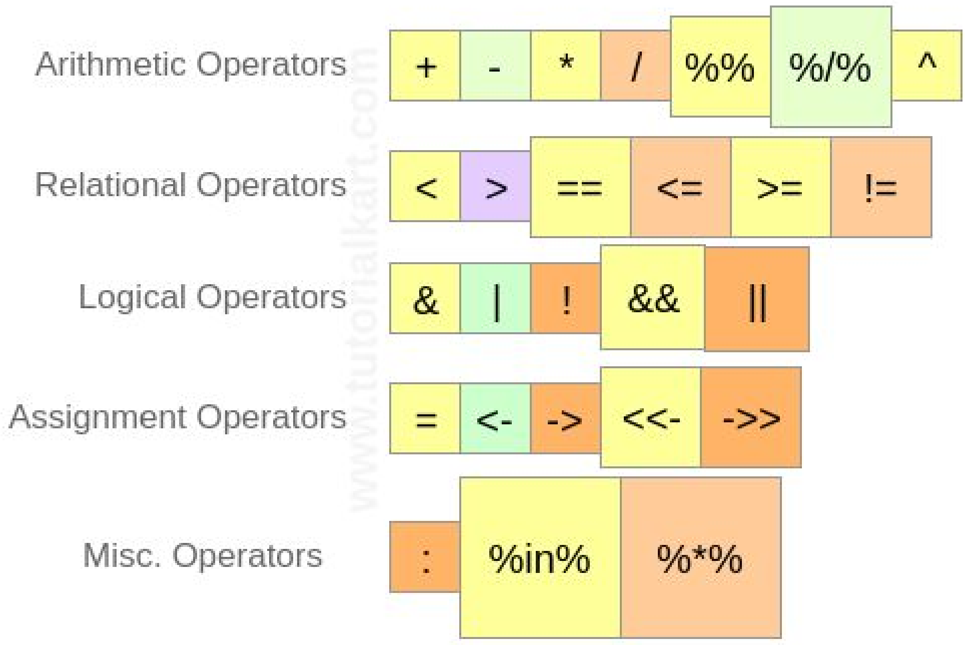
a)R is a programming language and free software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing. The R language is widely used among statisticians and data miners for developing statistical software and data analysis. Polls, data mining surveys, and studies of scholarly literature databases show substantial increases in popularity; as of February 2020, R ranks 13th in the TIOBE index, a measure of popularity of programming languages.
A GNU package,the official R software environment is written primarily in C, Fortran, and R itself(thus, it is partially self-hosting) and is freely available under the GNU General Public License. Pre-compiled executables are provided for various operating systems. Although R has a command line interface, there are several third-party graphical user interfaces, such as RStudio, an integrated development environment, and Jupyter, a notebook interface.
R is an implementation of the S programming language combined with lexical scoping semantics, inspired by Scheme. S was created by John Chambers in 1976, while at Bell Labs. There are some important differences, but much of the code written for S runs unaltered.
R was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, and is developed by the R Development Core Team (of which, as of August 2018, Chambers was a member). R is named partly after the first names of the first two R authors and partly as a play on the name of S. The project was conceived in 1992, with an initial version released in 1995 and a stable beta version (v1.0) on 29th February, 2000
The RStudio IDE is partly written in the C++ programming language and uses the Qt framework for its graphical user interface. The bigger percentage of the code is written in Java. JavaScript is also amongst the languages used.

Work on the RStudio IDE started around December 2010, and the first public beta version (v0.92) was officially announced in February 2011. Version 1.0 was released on 1 November 2016. Version 1.1 was released on 9 October 2017.

In April 2018, RStudio Inc. announced that it will provide operational and infrastructure support to Ursa Labs in support of the Labs focus on building a new data science runtime powered by Apache Arrow.

In April 2019, RStudio Inc. released a new product, the RStudio Job Launcher. The Job Launcher is an adjunct to RStudio Server. The launcher provides the ability to start processes within various batch processing systems (e.g. Slurm) and container orchestration platforms (e.g. Kubernetes). This function is only available in RStudio Server Pro (fee-based application).

b and c)



R has a wide variety of data types including scalars, vectors (numerical, character, logical), matrices, data frames, and lists.

<u>Vectors:</u>

a <- c(1,2,5.3,6,-2,4) # numeric vector
b <- c("one","two","three") # character vector
c <- c(TRUE,TRUE,TRUE,FALSE,TRUE,FALSE) #logical vector
Refer to elements of a vector using subscripts.
a[c(2,4)] # 2nd and 4th elements of vector

<u>Matrices:</u>

All columns in a matrix must have the same mode(numeric, character, etc.) and the same length.
The general format is:
mymatrix <- matrix(vector, nrow=r, ncol=c, byrow=FALSE,
  dimnames=list(char_vector_rownames, char_vector_colnames))
byrow=TRUE indicates that the matrix should be filled by rows.
byrow=FALSE indicates that the matrix should be filled by columns (the default).
dimnames provides optional labels for the columns and rows.
# generates 5 x 4 numeric matrix
y<-matrix(1:20, nrow=5,ncol=4)
# another example
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))
Identify rows, columns or elements using subscripts.
x[,4] # 4th column of matrix

```
x[3,] # 3rd row of matrix
x[2:4,1:3] # rows 2,3,4 of columns 1,2,3
```

Arrays:

Arrays are similar to matrices but can have more than two dimensions.

Data Frames:

A data frame is more general than a matrix, in that different columns can have different modes (numeric, character, factor, etc.). This is similar to SAS and SPSS datasets.

```
d <- c(1,2,3,4)
e <- c("red", "white", "red", NA)
f <- c(TRUE,TRUE,TRUE,FALSE)
mydata <- data.frame(d,e,f)
names(mydata) <- c("ID","Color","Passed") # variable names
```

There are a variety of ways to identify the elements of a data frame .

```
myframe[3:5] # columns 3,4,5 of data frame
myframe[c("ID","Age")] # columns ID and Age from data frame
myframe$X1 # variable x1 in the data frame
```

Lists:

An ordered collection of objects (components). A list allows you to gather a variety of (possibly unrelated) objects under one name.

```
# example of a list with 4 components -
# a string, a numeric vector, a matrix, and a scaler
w <- list(name="Fred", mynumbers=a, mymatrix=y, age=5.3)

# example of a list containing two lists
v <- c(list1,list2)
```

Identify elements of a list using the [[]] convention.

```
mylist[[2]] # 2nd component of the list
mylist[["mynumbers"]] # component named mynumbers in list
```

Factors:

Tell R that a variable is nominal by making it a factor. The factor stores the nominal values as a vector of integers in the range [1... k] (where k is the number of unique values in the nominal variable), and an internal vector of character strings (the original values) mapped to these integers.

```
# variable gender with 20 "male" entries and
# 30 "female" entries
gender <- c(rep("male",20), rep("female", 30))
gender <- factor(gender)
# stores gender as 20 1s and 30 2s and associates
# 1=female, 2=male internally (alphabetically)
# R now treats gender as a nominal variable
summary(gender)
```

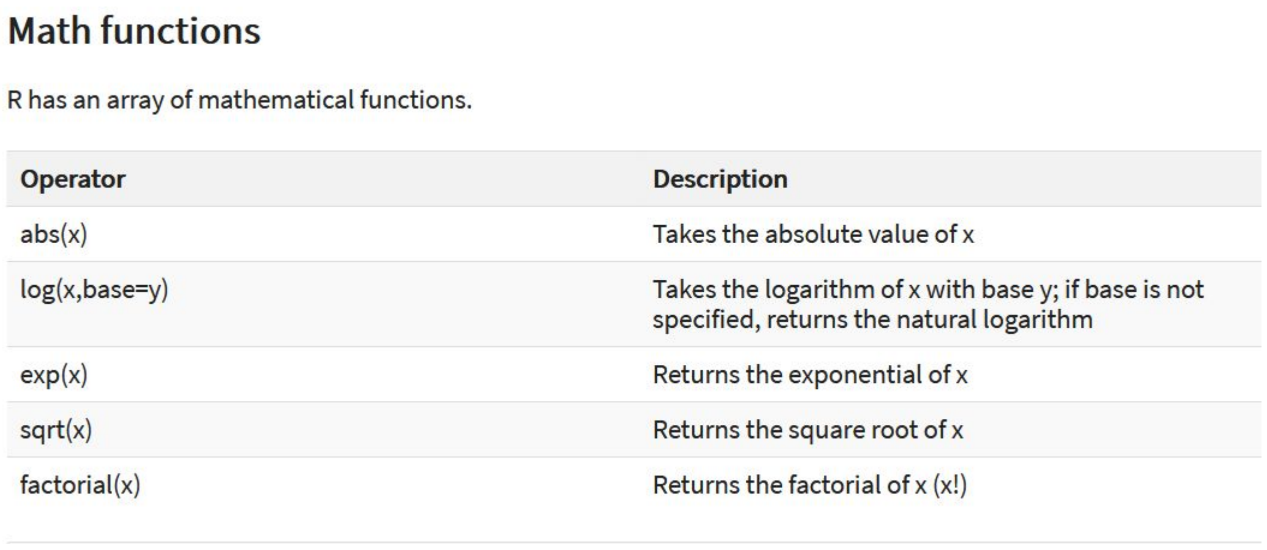
An ordered factor is used to represent an ordinal variable.

```
# variable rating coded as "large", "medium", "small'
rating <- ordered(rating)
# recodes rating to 1,2,3 and associates
# 1=large, 2=medium, 3=small internally
# R now treats rating as ordinal
```

R will treat factors as nominal variables and ordered factors as ordinal variables in statistical procedures and graphical analyses. You can use options in the factor( ) and ordered( ) functions to control the mapping of integers to strings (overiding the alphabetical ordering). You can also use factors to create value labels.

<u>Useful Functions:</u>

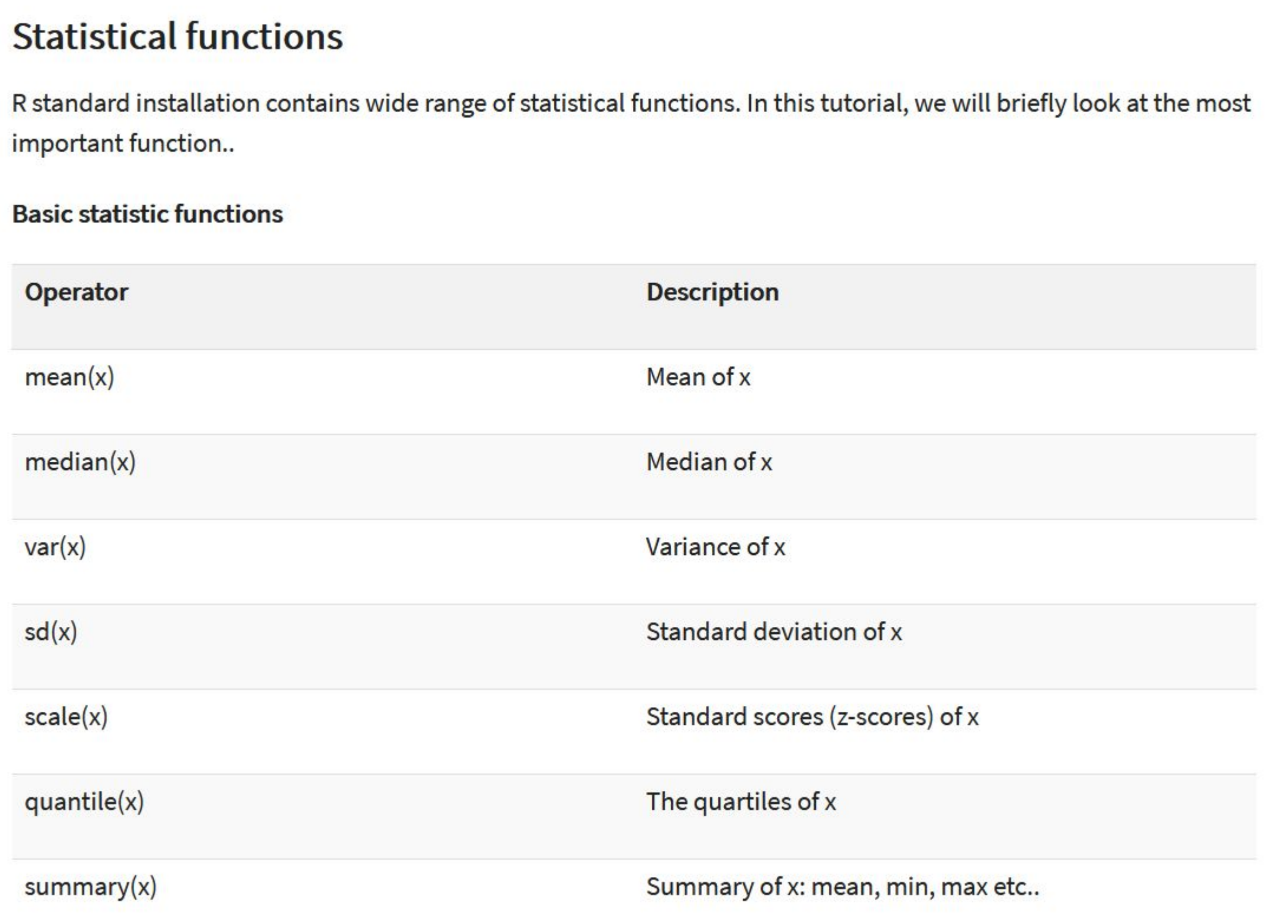
```
length(object) # number of elements or components
str(object)      # structure of an object
class(object)  # class or type of an object
names(object)  # names

c(object,object,...)       # combine objects into a vector
cbind(object, object, ...) # combine objects as columns
rbind(object, object, ...) # combine objects as rows

object  # prints the object

ls()       # list current objects
rm(object) # delete an object

newobject <- edit(object) # edit copy and save as newobject
fix(object)                  # edit in place
```



Math functions

R has an array of mathematical functions.

Operator	Description
abs(x)	Takes the absolute value of x
log(x,base=y)	Takes the logarithm of x with base y; if base is not specified, returns the natural logarithm
exp(x)	Returns the exponential of x
sqrt(x)	Returns the square root of x
factorial(x)	Returns the factorial of x (x!)



Statistical functions

R standard installation contains wide range of statistical functions. In this tutorial, we will briefly look at the most important function..

Basic statistic functions

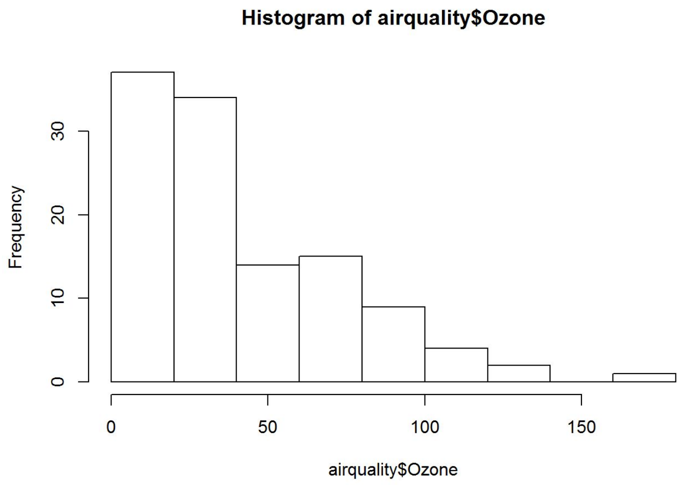
Operator	Description
mean(x)	Mean of x
median(x)	Median of x
var(x)	Variance of x
sd(x)	Standard deviation of x
scale(x)	Standard scores (z-scores) of x
quantile(x)	The quartiles of x
summary(x)	Summary of x: mean, min, max etc..

<u>The Base Plotting System in R:</u>
Plotting System:
The core plotting and graphics engine in R is encapsulated in the following packages:
graphics: contains plotting functions for the “base” graphing systems, including plot, hist, boxplot and many others.
grDevices: contains all the code implementing the various graphics devices, including X11, PDF, PostScript, PNG, etc.
The lattice plotting system is implemented using the following packages:
lattice: contains code for producing Trellis graphics, which are independent of the graphics system; includes functions like xyplot, bwplot, levelplot
grid: implements a different graphing system independent of the system; the lattice package builds on top of grid; we seldom call functions from the grid package directly

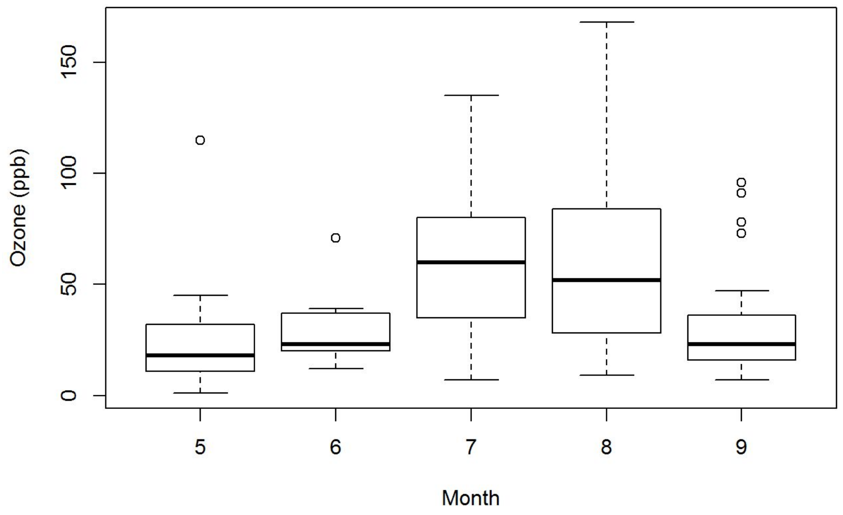
Base Graphics:
Base graphics are used most commonly and are a very powerful system for creating 2-D graphics.There are two phases to creating a base plot:
Initializing a new plot
Annotating (adding to) an existing plot
Calling plot(x, y) or hist(x) will launch a graphics device (if one is not already open) and draw a new plot on the device
If the arguments to plot are not of some special class, then the default method for plot is called; this function has many arguments, letting you set the title, x axis label, y axis label, etc.

The base graphics system has many parameters that can set and tweaked; these parameters are documented in ?par.

```
#Some examples:
library(datasets)
hist(airquality$Ozone)  ## Draw a new plot
```



```
airquality <- transform(airquality, Month = factor(Month))
boxplot(Ozone ~ Month, airquality, xlab = "Month", ylab = "Ozone (ppb)")
```



Some Important Base Graphics Parameters:

Many base plotting functions share a set of parameters. Here are a few key ones:

- pch: the plotting symbol (default is open circle)
- lty: the line type (default is solid line), can be dashed, dotted, etc.
- lwd: the line width, specified as an integer multiple

- col: the plotting color, specified as a number, string, or hex code; the colors() function gives you a vector of colors by name
- xlab: character string for the x-axis label
- ylab: character string for the y-axis label

Base Plotting Functions:

- plot: make a scatterplot, or other type of plot depending on the class of the object being plotted
- lines: add lines to a plot, given a vector x values and a corresponding vector of y values (or a 2-column matrix); this function just connects the dots
- points: add points to a plot
- text: add text labels to a plot using specified x, y coordinates
- title: add annotations to x, y axis labels, title, subtitle, outer margin
- mtext: add arbitrary text to the margins (inner or outer) of the plot
- axis: adding axis ticks/labels

More R packages in order to plot and manipulate data can be found here:
https://www.guru99.com/r-functions-programming.html
https://rstudio-pubs-static.s3.amazonaws.com/84527_6b8334fd3d9348579681b24d156e7e9d.html
https://www.computerworld.com/article/2921176/great-r-packages-for-data-import-wrangling-visualization.html
https://mode.com/blog/r-data-visualization-packages/

2)

```
library(rSymPy)
```

```
x <- Var("x")
x+x
```

```
## [1] "2*x"
```

```
x*x/x
```

```
## [1] "x"
```

```
y <- Var("x**3")
x/y
```

```
## [1] "x**(-2)"
```

```
z <- sympy("2.5*x**2")
z + y
```

```
## [1] "2.5*x**2 + x**3"
```

```
sympy("sqrt(8).evalf()")  # evaluate an expression
```

```
## [1] "2.82842712474619"
```

```
sympy("one = cos(1)**2 + sin(1)**2")
```

```
## [1] "cos(1)**2 + sin(1)**2"
```

```
sympy("(one - 1).evalf()")  # rounding errors
```

```
## [1] "-.0e-124"
```

```
sympy("(one - 1).evalf(chop=True)")  # rouding this type of roundoff errors
```

```
## [1] "0"
```

```
sympy("Eq(x**2+2*x+1,(x+1)**2)") # create an equation
```

```
## [1] "1 + 2*x + x**2 == (1 + x)**2"
```

```
sympy("a = x**2+2*x+1")
```

```
## [1] "1 + 2*x + x**2"
```

```
sympy("b = (x+1)**2")
```

```
## [1] "(1 + x)**2"
```

```
"0" == sympy("simplify(a-b)")  # if they are equal, the result is zero
```

```
## [1] TRUE
```

```
# simplify works in other tasks:
sympy("simplify((x**3 + x**2 - x - 1)/(x**2 + 2*x + 1))")
```

```
## [1] "-1 + x"
```

```
sympy("(x + 2)*(x - 3)")
```

```
## [1] "-(2 + x)*(3 - x)"
```

```
sympy("expand((x + 2)*(x - 3))")
```

```
## [1] "-6 - x + x**2"
```

```
sympy("factor(x**3 - x**2 + x - 1)")
```

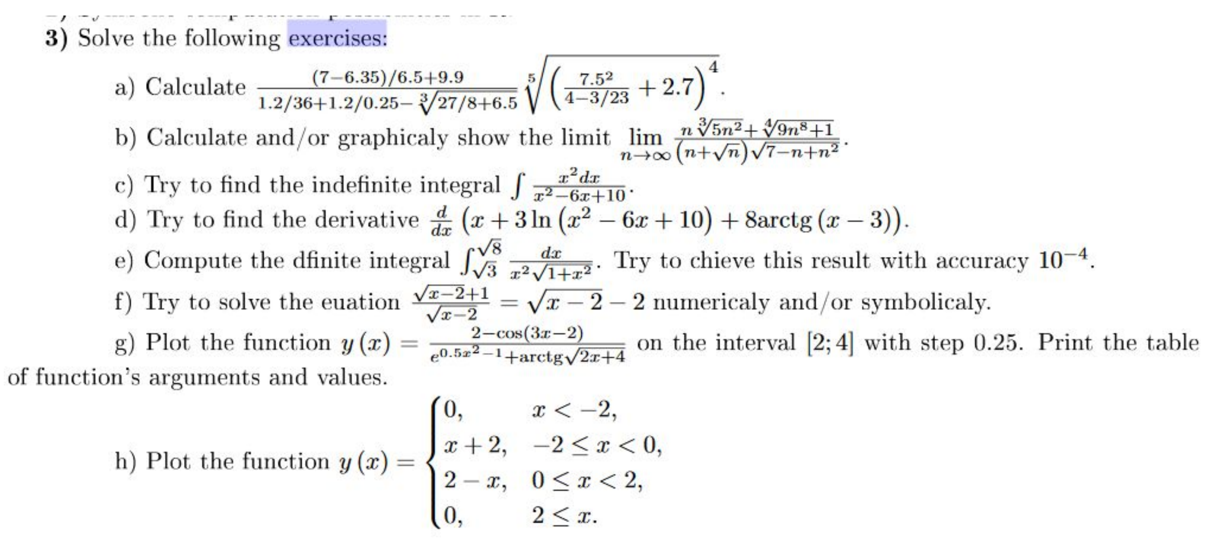
```
## [1] "-(1 + x**2)*(1 - x)"
```

```
y <- Var("y")
z <- Var("z")
sympy("collect(x*y + x - 3 + 2*x**2 - z*x**2 + x**3, x)")  # organize equation around var 'x'
```

```
## [1] "-3 + x*(1 + y) + x**2*(2 - z) + x**3"
```

```
sympy("(x*y**2 - 2*x*y*z + x*z**2 + y**2 - 2*y*z + z**2)/(x**2 - 1)")
```

```
## [1] "-(-2*y*z - 2*x*y*z + y**2 + z**2 + x*y**2 + x*z**2)/(1 - x**2)"
```



3) Solve the following exercises:

a) Calculate $\frac{(7-6.35)/6.5+9.9}{1.2/36+1.2/0.25-\sqrt[3]{27/8+6.5}}\sqrt[5]{\left(\frac{7.5^2}{4-3/23}+2.7\right)^4}$.

b) Calculate and/or graphicaly show the limit $\lim_{n\to\infty}\frac{n\sqrt[3]{5n^2}+\sqrt[4]{9n^8+1}}{(n+\sqrt{n})\sqrt{7-n+n^2}}$.

c) Try to find the indefinite integral $\int\frac{x^2dx}{x^2-6x+10}$.

d) Try to find the derivative $\frac{d}{dx}\left(x+3\ln\left(x^2-6x+10\right)+8\operatorname{arctg}(x-3)\right)$.

e) Compute the dfinite integral $\int_{\sqrt{3}}^{\sqrt{8}}\frac{dx}{x^2\sqrt{1+x^2}}$. Try to chieve this result with accuracy 10^{-4}.

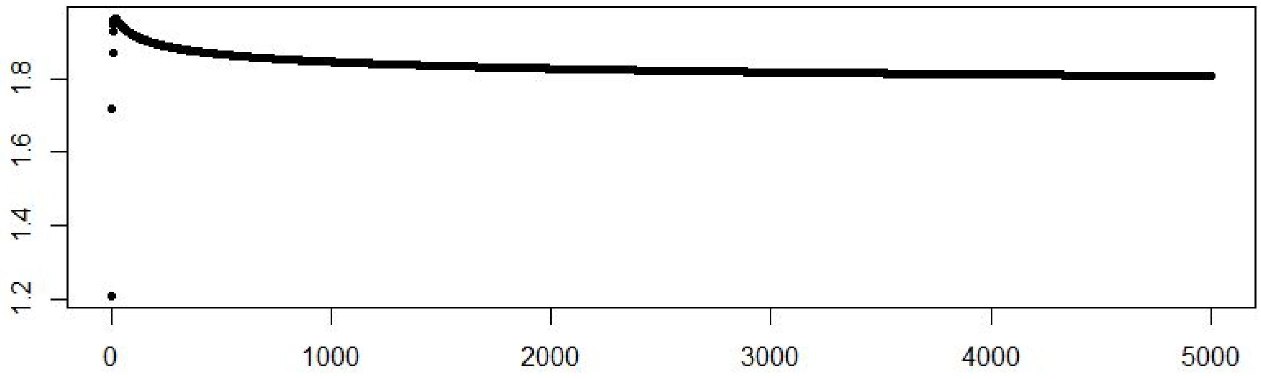
f) Try to solve the euation $\frac{\sqrt{x-2}+1}{\sqrt{x-2}}=\sqrt{x-2}-2$ numericaly and/or symbolicaly.

g) Plot the function $y(x)=\frac{2-\cos(3x-2)}{e^{0.5x^2-1}+\operatorname{arctg}\sqrt{2x+4}}$ on the interval $[2;4]$ with step 0.25. Print the table of function's arguments and values.

h) Plot the function $y(x)=\begin{cases}0, & x<-2,\\ x+2, & -2\le x<0,\\ 2-x, & 0\le x<2,\\ 0, & 2\le x.\end{cases}$

a) 36.28644

b) We can see from the plotting below that the limit → 1.8 when x→ INF.



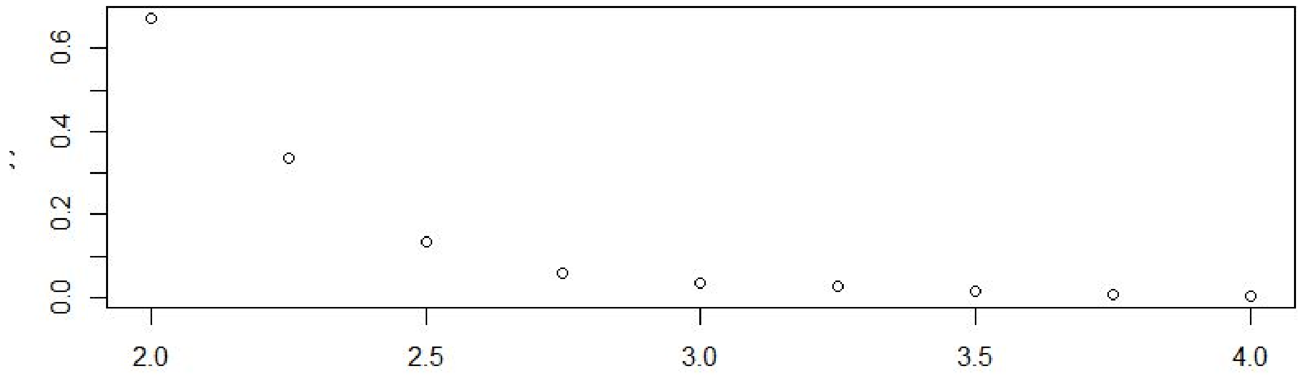
c) 3 * log(x^2 -6 * x + 10) + 8 * atan((2 * x - 6) / 2) + x + C

d)1 + 3 * ((2 * x - 6)/(x^2 - 6 * x + 10)) + 8 * (1/(1 + (x - 3)^2))

e)0.09404037 with absolute error < 1e-15 which is less than 10^(-4) and that is desired.

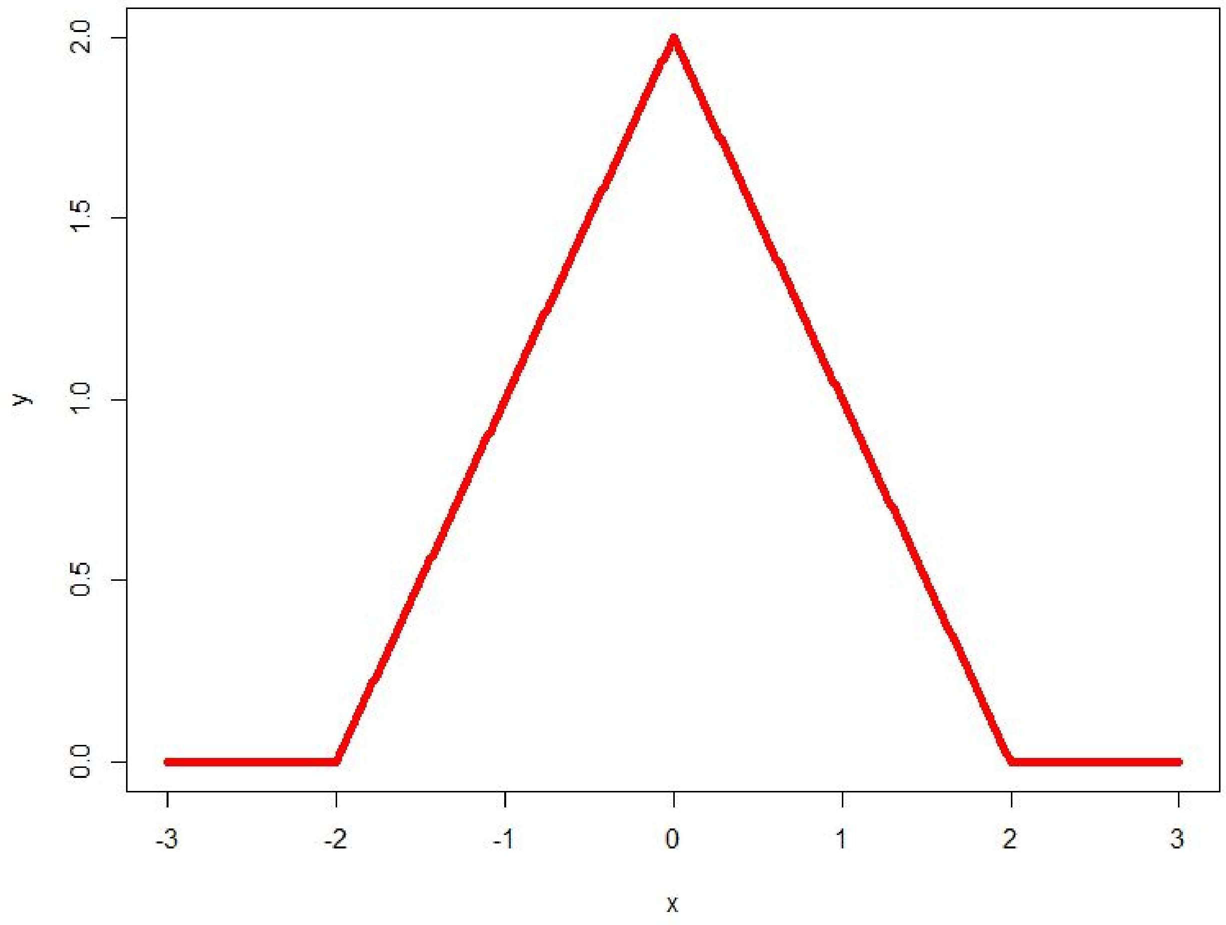
f) 12.90837

g)



```
> TBL
  col x       col y
1  2.00 0.671937584
2  2.25 0.334633097
3  2.50 0.134206801
4  2.75 0.057515300
5  3.00 0.036244875
6  3.25 0.025762873
7  3.50 0.015355442
8  3.75 0.007148812
9  4.00 0.002585857
```

h)



Program Code:

```
#3a
a3a <- (27/8+6.50)^(1/3)
kati<- (((7.5)^2)/(4-3/23) +2.7)^(4/5)
a3 <- (((7-6.35)/6.5 +9.9)/(1.2/36 + 1.2/0.25 -a3a))*kati
a3

#3b
f2 <-function(x){
  return((x*(5*x^2)^(1/3)+(9*x^8+1)^(1/4))/((x+x^(1/2))*((7-x+(x*x))^(1/2))))
}
x<-seq(0,5000,2)
fn<-apply(matrix(x),1,f2)
plot(x,fn,type="p",pch=20)
fn[length(fn)]
#3c)

install.packages("mosaicCalc")
library("mosaicCalc")
F = antiD((x^2)/(x^2-6*x+10) ~ x)
F

func <-function(x){(x^2/(x^2-6*x+10))}
#integrate(func,lower=0,upper=10)
#symbolic computations

install.packages("Ryacas")
install.packages("xml2")
library(Ryacas)
library(xml2)
yacasInstall()
x<-Sym('x')
f <-function(x){
  return(x*x/(x*x-6*x+10))
}
res<-yacas(integrate(f(x),x))
res

#3d)
F = D((x+3*log(x^2-6*x+10)+8*atan(x-3)) ~ x)
F

#3e)
f <- function(x) {1/((x^2)*sqrt(1+x^2))}
integrate(f, lower = sqrt(3), upper = sqrt(8))
```

```
#3f)
fnToFindRoot = function(x) {
  ((sqrt(x-2)+1)/sqrt(x-2)-sqrt(x-2)+2)
}

install.packages("rootSolve")
library("rootSolve")
uniroot.all(fnToFindRoot, c(2.03,13))

#3g
y<-function(x){
  return((2-cos(3*x-2))/
           (exp((x^2)/2-1)+
           atan(sqrt(2*x+4))))
}
x<-seq(2,4,0.25)
yy<-apply(matrix(x),1,y)
plot(x,yy)
TBL<-rbind(x,yy)
TBL
TBL<-data.frame(t(rbind(x,yy)))
names(TBL)<-c("col x","col y")
TBL
x<-TBL$'col x'
y<-TBL[,2]
y<-TBL[1:2,2]
TBL

#3h)
#h problem
f <- function(x){
  if(x< -2)
           return (0)
  if( x < 0)
           return (x+2)
  if ( x < 2)
           return (2-x)
  return(0)
}

x<-seq(-3,3,0.01)
n<- length(x)
#y<-vector(mode="numeric",length=n)
y<- numeric(length=n)
for(i in seq(1,n)){
  y[i]<-f(x[i])
}
```

```
plot(x,y,type="b",pch=20,col="red")

positions <- matrix(c(1,2,3,3),
                    byrow=TRUE,
                    ncol=2)
layout(positions)

layout(1)
par(mar=c(3.1,3.1,0.1,0.1))
plot(x,y,type="b",pch=20,lwd=2,xlab="",ylab="")
mtext("x axis",1,3)
mtext("y axis",2,3)
```

LAB 3

LW 3. Discrete random variable

1) There are 8 balls in a box (3 of them are white); 3 balls are selected at random. Let ξ be the number of the
white balls taken. Write down the distribution law (a table) and the distribution function $F_\xi(x)$ of ξ. Sketch the graph of $F_\xi(x)$. Find expected value $E\xi$, variance $D\xi$, probability $P(\xi < 2)$. Analyse two cases separately:

a) random sampling without replacements.
b) random sampling with replacement.

2) There are 7 balls in a box (2 of them are white); 3 balls are selected at random. Let ξ be the number of the white balls taken. Write down the distribution law (a table) and the distribution function $F_\xi(x)$ of ξ. Sketch the graph of $F_\xi(x)$. Find expected value $E\xi$, variance $D\xi$, probability $P(\xi < 2)$. Analyse two cases separately:

a) random sampling without replacements.
b) random sampling with replacement.

3) There are 7 balls in a box (5 of them are white); 2 balls are selected at random. Let ξ be the number of the white balls taken. Write down the distribution law (a table) and the distribution function $F_\xi(x)$ of ξ. Sketch the graph of $F_\xi(x)$. Find expected value $E\xi$, variance $D\xi$, probability $P(\xi < 2)$. Analyse two cases separately:

a) random sampling without replacements.
b) random sampling with replacement.

Output:

```
> # Problem 1
> # a) sampling without replacements
Check if all probs sum to 1:
> message(sum(probs))
1
> message('Distribution law:')
Distribution law:
> DL <- rbind(ksi, probs)
> DL
           [,1]      [,2]      [,3]       [,4]
ksi   0.0000000 1.0000000 2.0000000 3.00000000
probs 0.1785714 0.5357143 0.2678571 0.01785714
> message('Distribution function:')
Distribution function:
> cs <- cumsum(probs)
> DF <- rbind(c(ksi, Inf), c(0, cs))
> DF
     [,1]      [,2]      [,3]      [,4] [,5]
[1,]    0 1.0000000 2.0000000 3.0000000  Inf
[2,]    0 0.1785714 0.7142857 0.9821429    1
> plot(c(ksi, Inf), c(0, cs))
> Pxl2<- probs[1]+probs[2]
> Pxl2
[1] 0.7142857
> # Numerical characteristics
> EX <- sum(ksi*probs)
> DX <- sum((ksi-EX)^2*probs)
> message('EX, DX =')
EX, DX =
> EX
[1] 1.125
> DX
[1] 0.5022321
>
> # b) sampling with replacements
Check if all probs sum to 1:
> message(sum(probs))
1
> message('Distribution law:')
Distribution law:
> DL <- rbind(ksi, probs)
> DL
           [,1]      [,2]      [,3]       [,4]
ksi   0.0000000 1.0000000 2.0000000 3.00000000
probs 0.2441406 0.4394531 0.2636719 0.05273438
> message('Distribution function:')
Distribution function:
> cs <- cumsum(probs)
> DF <- rbind(c(ksi, Inf), c(0, cs))
> DF
     [,1]      [,2]      [,3]      [,4] [,5]
[1,]    0 1.0000000 2.0000000 3.0000000  Inf
[2,]    0 0.2441406 0.6835938 0.9472656    1
> plot(c(ksi, Inf), c(0, cs))
> Pxl2<- probs[1]+probs[2]
> Pxl2
[1] 0.6835938
> # Numerical characteristics
> EX <- sum(ksi*probs)
> DX <- sum((ksi-EX)^2*probs)
> message('EX, DX =')
EX, DX =
> EX
[1] 1.125
> DX
[1] 0.703125
```

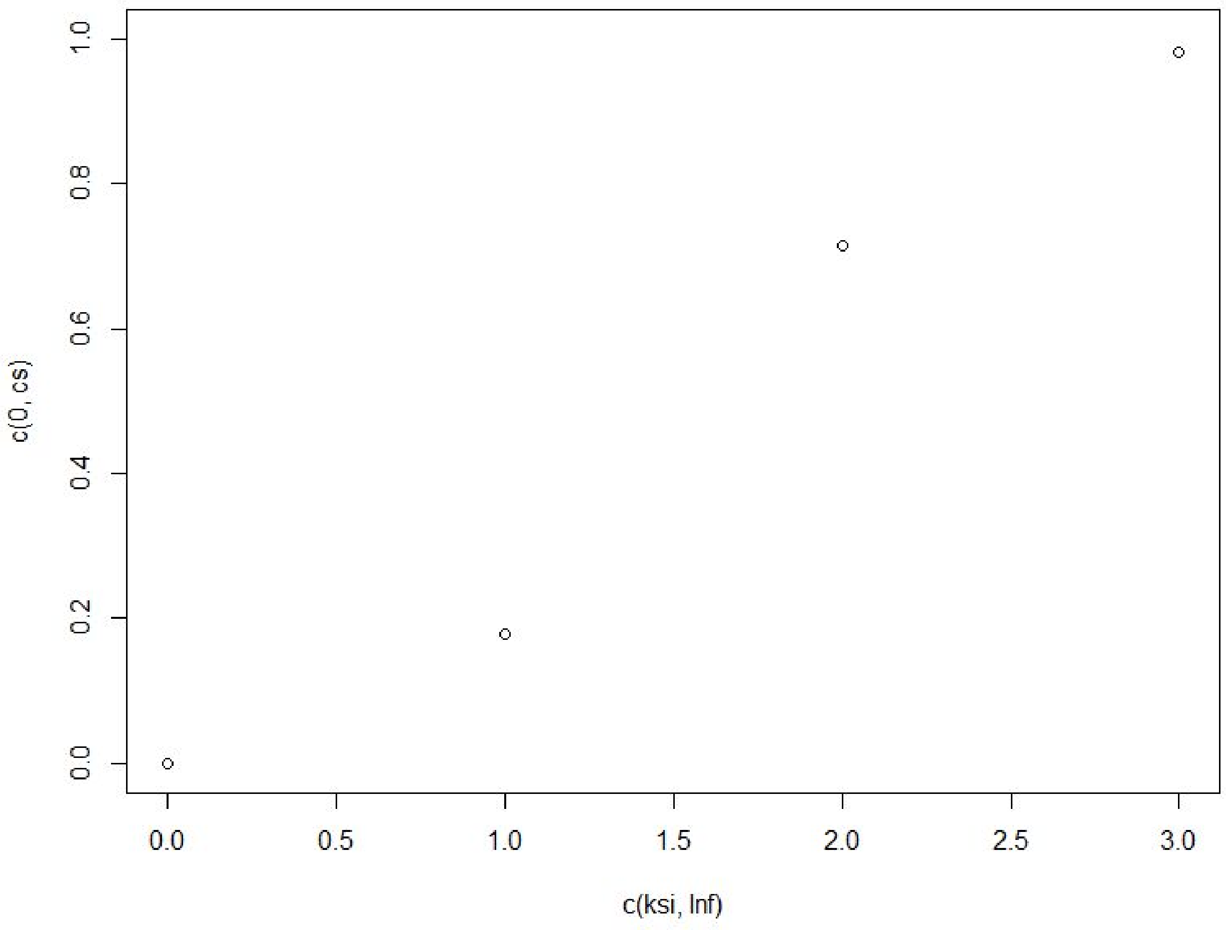
```
> # Problem 2

Check if all probs sum to 1:
> message(sum(probs))
1
> message('Distribution law:')
Distribution law:
> DL <- rbind(ksi, probs)
> DL
           [,1]      [,2]      [,3] [,4]
ksi   0.0000000 1.0000000 2.0000000    3
probs 0.2857143 0.5714286 0.1428571    0
> message('Distribution function:')
Distribution function:
> cs <- cumsum(probs)
> DF <- rbind(c(ksi, Inf), c(0, cs))
> DF
     [,1]      [,2]      [,3] [,4] [,5]
[1,]    0 1.0000000 2.0000000    3  Inf
[2,]    0 0.2857143 0.8571429    1    1
> plot(c(ksi, Inf), c(0, cs))
> Pxl2<-probs[1]+probs[2]
> Pxl2
[1] 0.8571429
> # Numerical characteristics
> EX <- sum(ksi*probs)
> DX <- sum((ksi-EX)^2*probs)
> message('EX, DX =')
EX, DX =
> EX
[1] 0.8571429
> DX
[1] 0.4081633
>
> # b) sampling with replacements
Check if all probs sum to 1:
> message(sum(probs))
1
> message('Distribution law:')
Distribution law:
> DL <- rbind(ksi, probs)
> DL
           [,1]      [,2]      [,3]        [,4]
```

```
ksi   0.0000000 1.0000000 2.0000000 3.00000000
probs 0.3644315 0.4373178 0.1749271 0.02332362
> message('Distribution function:')
Distribution function:
> cs <- cumsum(probs)
> DF <- rbind(c(ksi, Inf), c(0, cs))
> DF
     [,1]      [,2]      [,3]      [,4] [,5]
[1,]    0 1.0000000 2.0000000 3.0000000  Inf
[2,]    0 0.3644315 0.8017493 0.9766764    1
> plot(c(ksi, Inf), c(0, cs))
> Pxl2<- probs[1]+probs[2]
> Pxl2
[1] 0.8017493
> # Numerical characteristics
> EX <- sum(ksi*probs)
> DX <- sum((ksi-EX)^2*probs)
> message('EX, DX =')
EX, DX =
> EX
[1] 0.8571429
> DX
[1] 0.6122449

> # Problem 3
>
> N <- 7 # No of balls in the box
> N.W <- 5 # No of white balls
> N.B <- N-N.W # No of non-white balls
> N.taken <- 2
>
> # a) sampling without replacements

Check if all probs sum to 1:
> message(sum(probs))
1
> message('Distribution law:')
Distribution law:
> DL <- rbind(ksi, probs)
> DL
            [,1]      [,2]      [,3]
ksi   0.00000000 1.0000000 2.0000000
probs 0.04761905 0.4761905 0.4761905
> message('Distribution function:')
```

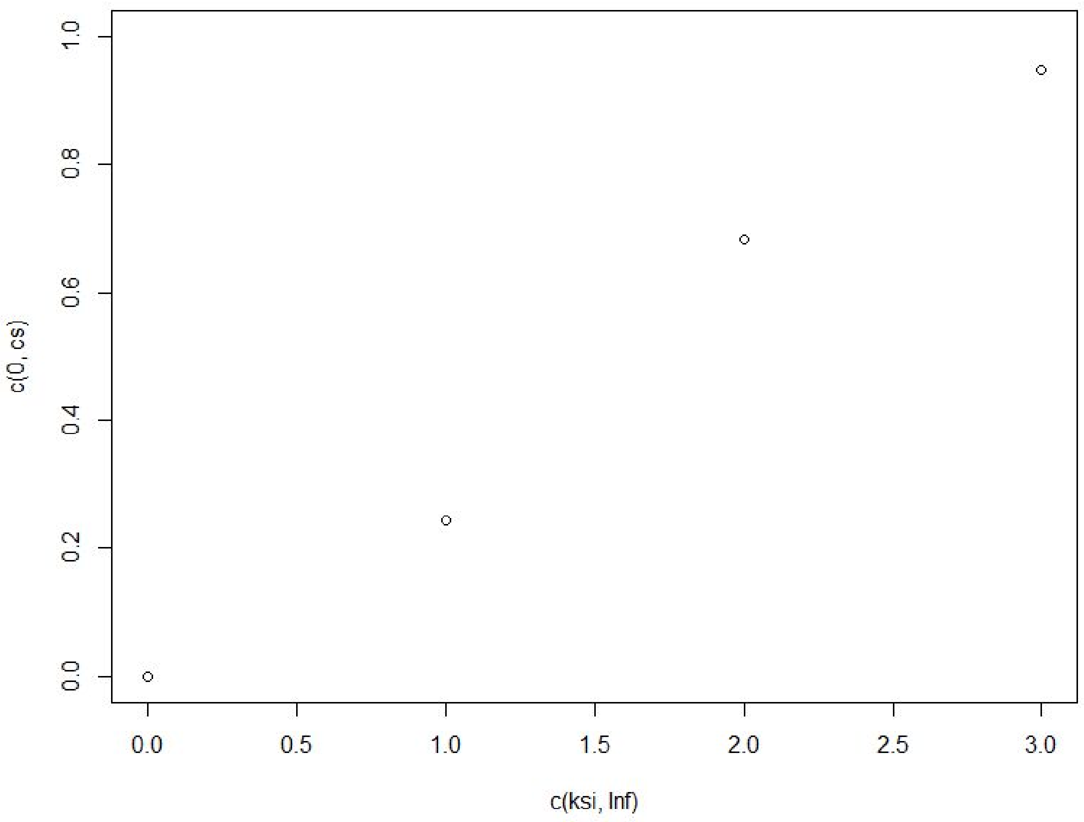
```
Distribution function:
> cs <- cumsum(probs)
> DF <- rbind(c(ksi, Inf), c(0, cs))
> DF
     [,1]       [,2]      [,3] [,4]
[1,]    0 1.00000000 2.0000000  Inf
[2,]    0 0.04761905 0.5238095    1
> plot(c(ksi, Inf), c(0, cs))
> Pxl2<-probs[1]+probs[2]
> Pxl2
[1] 0.5238095
> # Numerical characteristics
> EX <- sum(ksi*probs)
> DX <- sum((ksi-EX)^2*probs)
> message('EX, DX =')
EX, DX =
> EX
[1] 1.428571
> DX
[1] 0.3401361
>
>
> # b) sampling with replacements

Check if all probs sum to 1:
> message(sum(probs))
1
> message('Distribution law:')
Distribution law:
> DL <- rbind(ksi, probs)
> DL
            [,1]      [,2]      [,3]
ksi   0.00000000 1.0000000 2.0000000
probs 0.08163265 0.4081633 0.5102041
> message('Distribution function:')
Distribution function:
> cs <- cumsum(probs)
> DF <- rbind(c(ksi, Inf), c(0, cs))
> DF
     [,1]       [,2]      [,3] [,4]
[1,]    0 1.00000000 2.0000000  Inf
[2,]    0 0.08163265 0.4897959    1
> plot(c(ksi, Inf), c(0, cs))
> Pxl2<- probs[1]+probs[2]
> Pxl2
[1] 0.4897959
> # Numerical characteristics
> EX <- sum(ksi*probs)
> DX <- sum((ksi-EX)^2*probs)
> message('EX, DX =')
EX, DX =
> EX
[1] 1.428571
> DX
[1] 0.4081633
```

Figures:

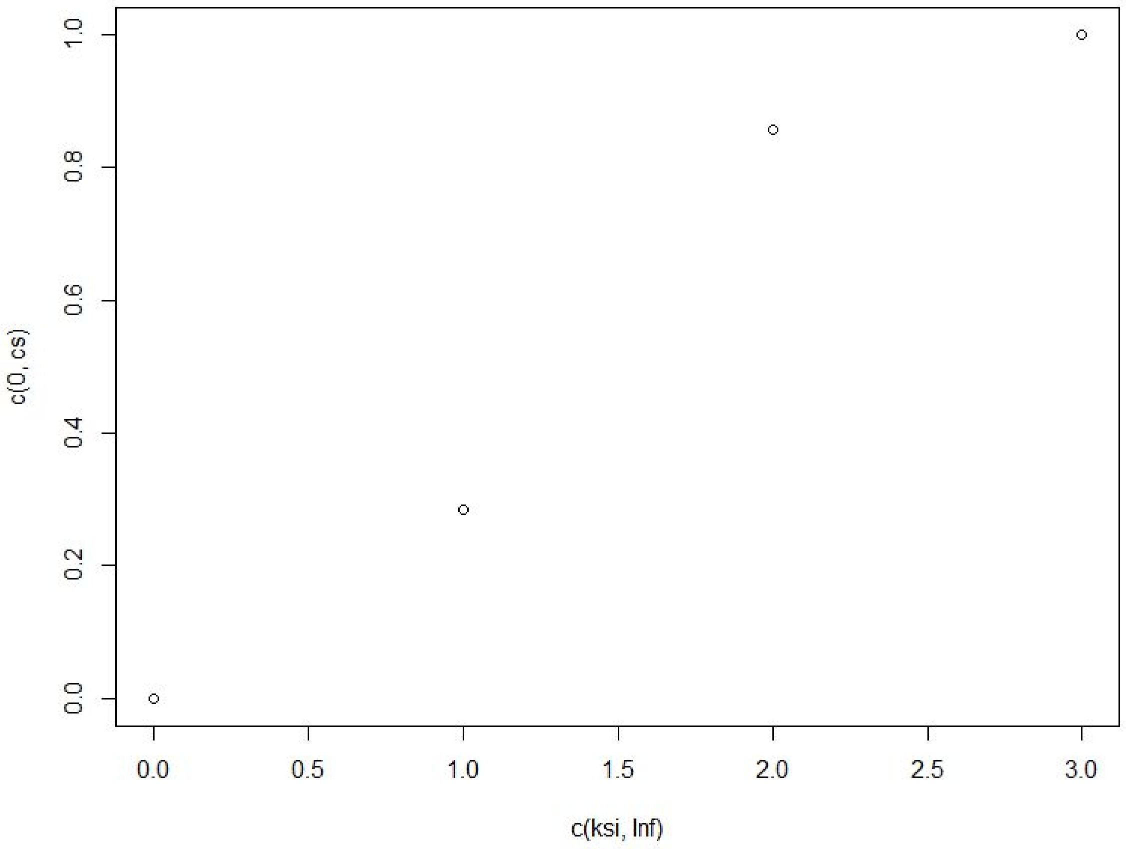
Problem 1: a)



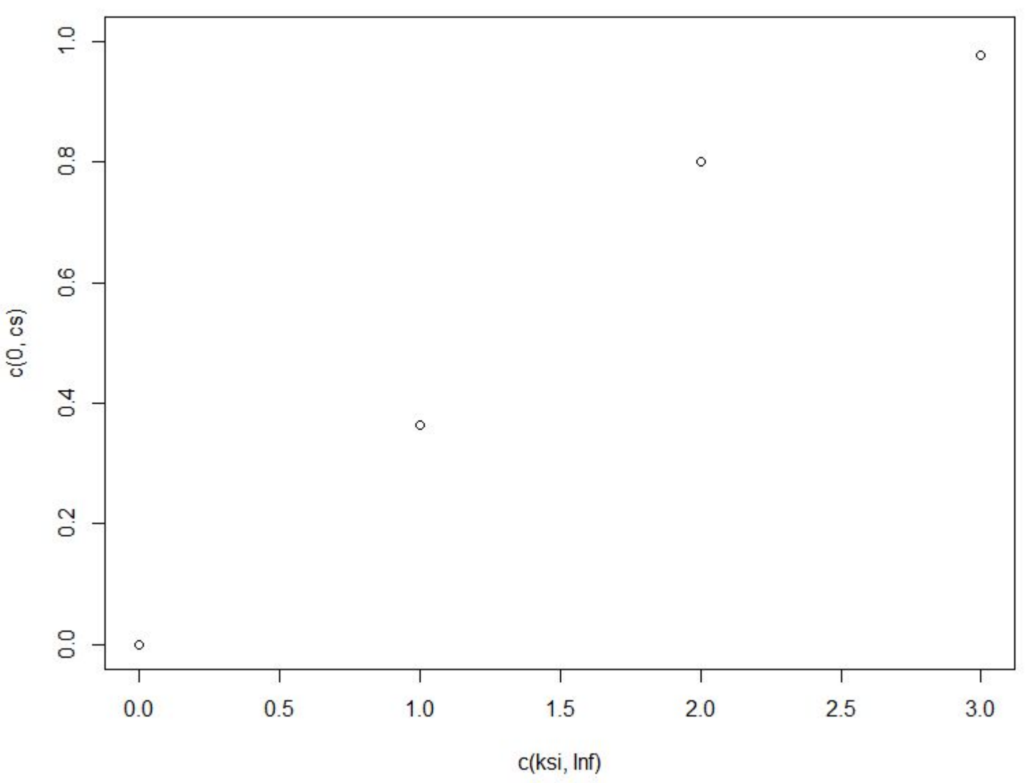
Problem 1: b)



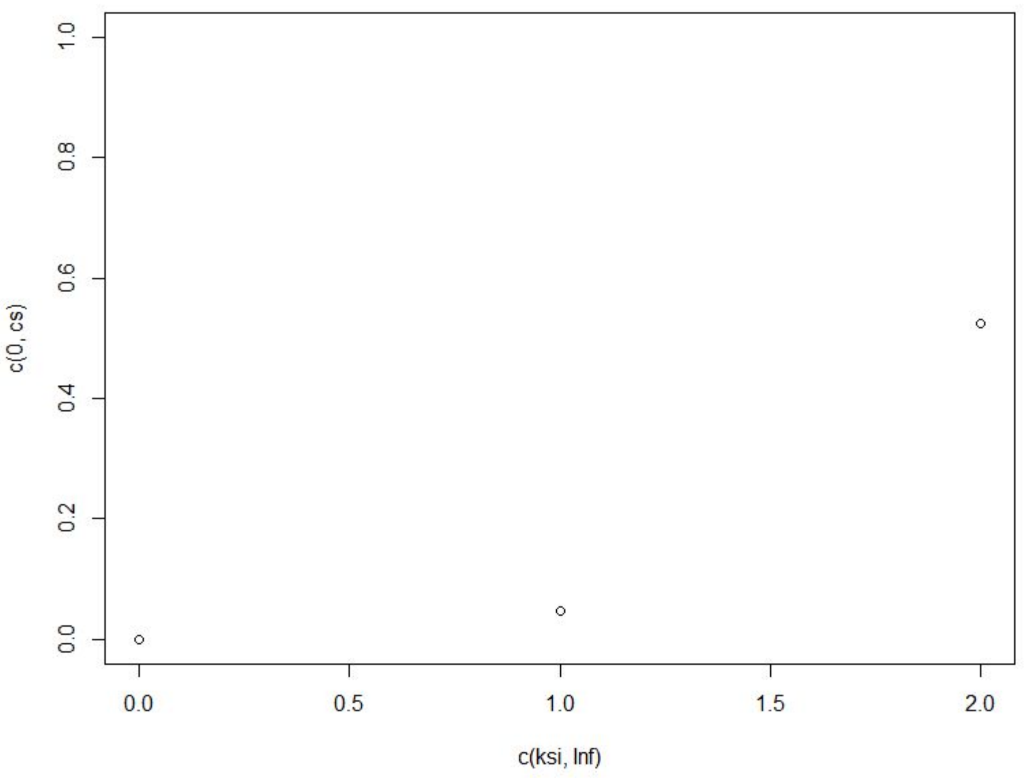
Problem 2 : a)



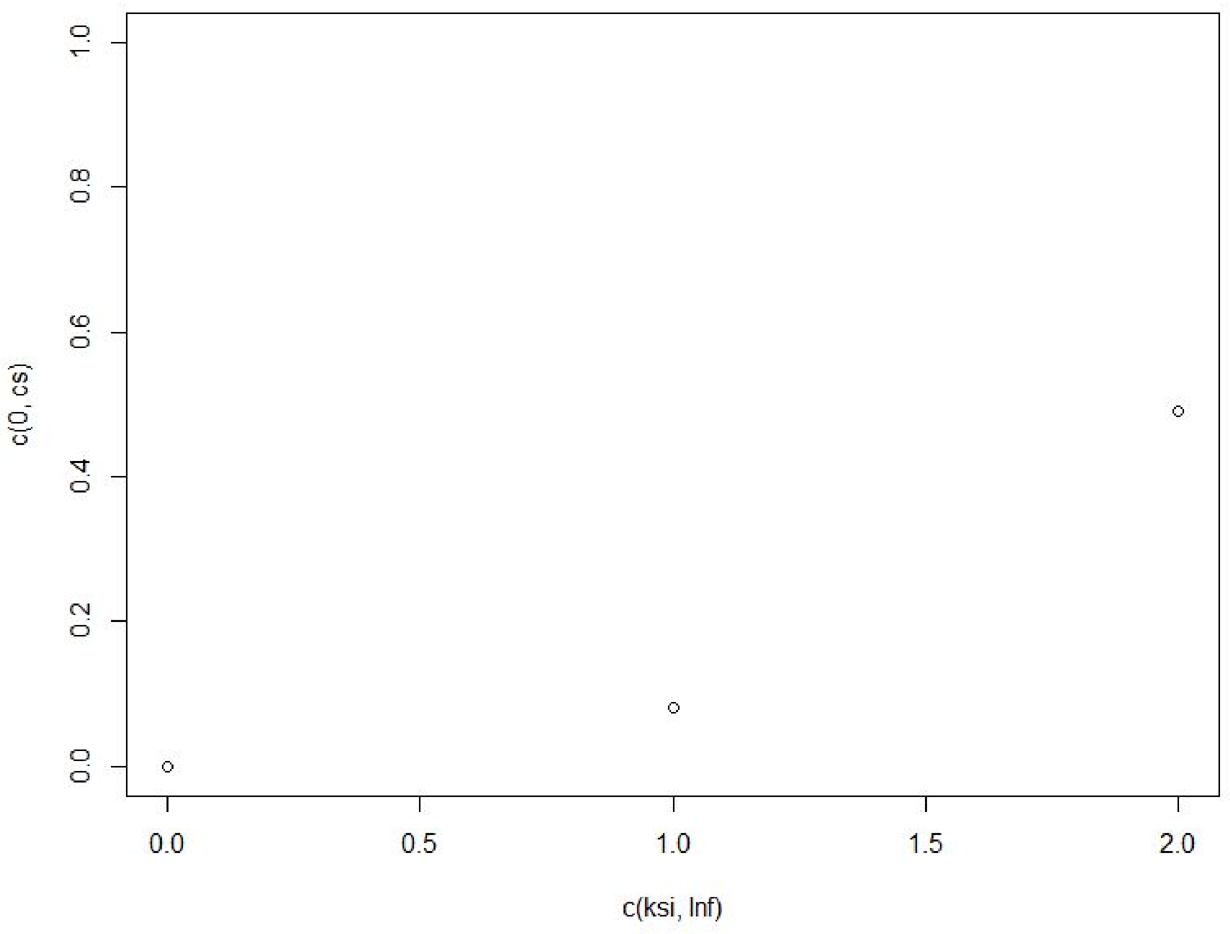
Problem 2 : b)



Problem 3 : a)



Problem 3 : b)



Program Code:

```
# Problem 1

N <- 8 # No of balls in the box
N.W <- 3 # No of white balls
N.B <- N-N.W # No of non-white balls
N.taken <- 3

# a) sampling without replacements
ksi <- 0:N.taken
#probs <- c(5/8*4/7*3/6,
#          ...)
probs <- c(N.B/N * (N.B-1)/(N-1) * (N.B-2)/(N-2),
           N.W/N * N.B/(N-1) * (N.B-1)/(N-2)+
           N.B/N * N.W/(N-1) * (N.B-1)/(N-2)+
           N.B/N * (N.B-1)/(N-1) * N.W/(N-2),
           N.W/N * N.B/(N-1) * (N.W-1)/(N-2)+
           N.W/N * (N.W-1)/(N-1) * N.B/(N-2)+
           N.B/N * N.W/(N-1) * (N.W-1)/(N-2),
           N.W/N * (N.W-1)/(N-1) * (N.W-2)/(N-2))
message('Check if all probs sum to 1:')
message(sum(probs))
message('Distribution law:')
DL <- rbind(ksi, probs)
DL
message('Distribution function:')
cs <- cumsum(probs)
DF <- rbind(c(ksi, Inf), c(0, cs))
DF
plot(c(ksi, Inf), c(0, cs))
Pxl2<- probs[1]+probs[2]
Pxl2
# Numerical characteristics
EX <- sum(ksi*probs)
DX <- sum((ksi-EX)^2*probs)
message('EX, DX =')
EX
DX

# b) sampling with replacements
ksi <- 0:N.taken
# probs <- c((5/8)^3, ..)
C <- function(n, k) {
  return(factorial(n)/
           (factorial(n-k)*factorial(k)))
```

```
}
probs <- numeric(length=N.taken+1)
for (i in 0:N.taken)
  probs[i+1] <- C(N.taken, i)*
  (N.W/N)^i *
  (N.B/N)^(N.taken-i)
message('Check if all probs sum to 1:')
message(sum(probs))
message('Distribution law:')
DL <- rbind(ksi, probs)
DL
message('Distribution function:')
cs <- cumsum(probs)
DF <- rbind(c(ksi, Inf), c(0, cs))
DF
plot(c(ksi, Inf), c(0, cs))
Pxl2<- probs[1]+probs[2]
Pxl2
# Numerical characteristics
EX <- sum(ksi*probs)
DX <- sum((ksi-EX)^2*probs)
message('EX, DX =')
EX
DX

# Problem 2

N <- 7 # No of balls in the box
N.W <- 2# No of white balls
N.B <- N-N.W # No of non-white balls
N.taken <- 3

# a) sampling without replacements
ksi <- 0:N.taken
#probs <- c(5/7*4/6*3/5,
#          ...)
probs <- c(N.B/N * (N.B-1)/(N-1) * (N.B-2)/(N-2),
          N.W/N * N.B/(N-1) * (N.B-1)/(N-2)+N.B/N * N.W/(N-1) * (N.B-1)/(N-2)+N.B/N * (N.B-1)/(N-1) *
N.W/(N-2),
          N.W/N * N.B/(N-1) * (N.W-1)/(N-2)+N.W/N * (N.W-1)/(N-1) * N.B/(N-2)+ N.B/N * N.W/(N-1) *
(N.W-1)/(N-2),
          0)
message('Check if all probs sum to 1:')
message(sum(probs))
message('Distribution law:')
DL <- rbind(ksi, probs)
DL
message('Distribution function:')
```

```
cs <- cumsum(probs)
DF <- rbind(c(ksi, Inf), c(0, cs))
DF
plot(c(ksi, Inf), c(0, cs))
Pxl2<-probs[1]+probs[2]
Pxl2
# Numerical characteristics
EX <- sum(ksi*probs)
DX <- sum((ksi-EX)^2*probs)
message('EX, DX =')
EX
DX

# b) sampling with replacements
ksi <- 0:N.taken
# probs <- c((5/7)^3, ..)
C <- function(n, k) {
  return(factorial(n)/
           (factorial(n-k)*factorial(k)))
}
probs <- numeric(length=N.taken+1)
for (i in 0:N.taken)
  probs[i+1] <- C(N.taken, i)*
  (N.W/N)^i *
  (N.B/N)^(N.taken-i)
message('Check if all probs sum to 1:')
message(sum(probs))
message('Distribution law:')
DL <- rbind(ksi, probs)
DL
message('Distribution function:')
cs <- cumsum(probs)
DF <- rbind(c(ksi, Inf), c(0, cs))
DF
plot(c(ksi, Inf), c(0, cs))
Pxl2<- probs[1]+probs[2]
Pxl2
# Numerical characteristics
EX <- sum(ksi*probs)
DX <- sum((ksi-EX)^2*probs)
message('EX, DX =')
EX
DX
```

```
# Problem 3

N <- 7 # No of balls in the box
N.W <- 5 # No of white balls
N.B <- N-N.W # No of non-white balls
N.taken <- 2

# a) sampling without replacements
ksi <- 0:N.taken
#probs <- c(2/7*1/6,
#           ...)
probs <- c(N.B/N * (N.B-1)/(N-1),
           N.W/N * N.B/(N-1) +N.B/N * N.W/(N-1),
           N.W/N * (N.W-1)/(N-1))

message('Check if all probs sum to 1:')
message(sum(probs))
message('Distribution law:')
DL <- rbind(ksi, probs)
DL
message('Distribution function:')
cs <- cumsum(probs)
DF <- rbind(c(ksi, Inf), c(0, cs))
DF
plot(c(ksi, Inf), c(0, cs))
Pxl2<-probs[1]+probs[2]
Pxl2
# Numerical characteristics
EX <- sum(ksi*probs)
DX <- sum((ksi-EX)^2*probs)
message('EX, DX =')
EX
DX


# b) sampling with replacements
ksi <- 0:N.taken
# probs <- c((2/7)^2, ..)
C <- function(n, k) {
  return(factorial(n)/
           (factorial(n-k)*factorial(k)))
}
probs <- numeric(length=N.taken+1)
for (i in 0:N.taken)
  probs[i+1] <- C(N.taken, i)*
  (N.W/N)^i *
  (N.B/N)^(N.taken-i)
```

```
message('Check if all probs sum to 1:')
message(sum(probs))
message('Distribution law:')
DL <- rbind(ksi, probs)
DL
message('Distribution function:')
cs <- cumsum(probs)
DF <- rbind(c(ksi, Inf), c(0, cs))
DF
plot(c(ksi, Inf), c(0, cs))
Pxl2<- probs[1]+probs[2]
Pxl2
# Numerical characteristics
EX <- sum(ksi*probs)
DX <- sum((ksi-EX)^2*probs)
message('EX, DX =')
EX
DX
```

LAB 4

LW 4. Statistical description of discrete data

1) 15 randomly chosen families were surveyed to reveal the number of children: 0, 1, 1, 0, 2, 1, 2, 3, 2, 2, 2, 1, 4, 3, 3. Describe the data statistically.

a) Variational series.

b) Sample size, mean, dispersion, standard deviation, quartiles.

c) Distribution law.

d) Frequency polygon, histogram, PDF, CDF, boxplot.

2) Import datafile rooms.txt that consists number of rooms of sold flats. Find variational series, statistical array, plot frequency polygon and histogram, find cumulative frequencies, proportionate frequencies, cumulative proportionate frequencies. Find and plot empirical distribution function, calculate mean (average) value, variance, and standard deviation.

Program Code:

```
sink('Analysis.txt')

#number of children
X<- c( 0,1,1,0,2,1,2,3,2,2,2,1,4,3,3)
print('Statistical data / Sample')
print(X)
#a)variational series
VS<-sort(X)

#b)other numerical charateristics
#sample size
n<-length(X)
#average value
m<-mean(X)
VS
m
n
#spread
DX<-var(X)
sX <-sd(X)

Me <-median(X)
Me<-quantile(X,0.5)
Q1<-quantile(X,0.25)
Q3<-quantile(X,0.75)

#c) distr. law
summary(X)
str(X)
tbl=table(X) #distribution law in table format

nms<-sort(unique(X))
```

```
freqs<-as.numeric(tbl)
df<-data.frame(t(freqs))
names(df) <-paste("NoC = ",nms,sep='')
row.names(df)<-"Freq."
df

#d) Frequency polygon
layout(matrix(c(1,2,3,4,5,5),byrow=TRUE,ncol=2))
plot(nms,freqs,type='l',col='red', main='freq polygon')
#Histogram
barplot(freqs/n,names.arg=as.character(nms),type='l',col='red', main='histogram')
#empirical PDF
plot(nms,freqs/n,type='l',col='red', main='Empirical PDF')
#empirical CDF
nms2<-c(-1,nms,5)
cfreqs<-c(0,cumsum(freqs)/n,1)
plot(nms2,cfreqs,type='s',col='red', main='Empirical CDF')
layout(1)
boxplot(X,horizontal =TRUE,xlab='No of children',ylab ='Boxplot')

print('')
print('the boxplot graphically summarizes center and spread of the data')
library('e1071')

assymetry1<-skewness(X)
Mo <-nms[which(max(freqs)==freqs)]
print(paste('The most common number of children in a family is ',Mo,sep=''))
if(assymetry1<0)
  print(paste('Having less than ',Mo,'children is more common.',sep=''))
if(assymetry1>0)
  print(paste('Having more than ',Mo,'children is more common.',sep=''))




#number of children in the roomsLW4.txt
X<-read.csv(file='roomsLW4.txt', sep='\t', dec=',', header=TRUE)
X<-X$ROOMS
print('Statistical data / Sample')
print(X)
#a)variational series
VS<-sort(X)

#b)other numerical charateristics
#sample size
n<-length(X)
#average value
```

```
m<-mean(X)
#spread
DX<-var(X)
sX <-sd(X)

Me <-median(X)
Me<-quantile(X,0.5)
Q1<-quantile(X,0.25)
Q3<-quantile(X,0.75)

#c) distr. law
summary(X)
str(X)
tbl=table(X) #distribution law in table format

nms<-sort(unique(X))
freqs<-as.numeric(tbl)
df<-data.frame(t(freqs/n))
names(df) <-paste("NoR = ",nms,sep='')
row.names(df)<-"Freq."
df

#d) Frequency polygon
layout(matrix(c(1,2,3,4,5,5),byrow=TRUE,ncol=2))
plot(nms,freqs,type='l',col='red', main='freq polygon')
#Histogram
barplot(freqs/n,names.arg=as.character(nms),type='l',col='red', main='histogram')
#empirical PDF
plot(nms,freqs/n,type='l',col='red', main='Empirical PDF')
#empirical CDF
nms2<-c(-1,nms,5)
cfreqs<-c(0,cumsum(freqs)/n,1)
plot(nms2,cfreqs,type='s',col='red', main='Empirical CDF')
layout(1)
boxplot(X,horizontal =TRUE,xlab='No of rooms',ylab ='Boxplot')

print('')
print('the boxplot graphically summarizes center and spread of the data')
library('e1071')

assymetry1<-skewness(X)
Mo <-nms[which(max(freqs)==freqs)]
print(paste('The most common number of rooms in flats is ',Mo,sep=''))
if(assymetry1<0)
  print(paste('Having less than ',Mo,'rooms is more common.',sep=''))
if(assymetry1>0)
  print(paste('Having more than ',Mo,'rooms is more common.',sep=''))
```

Output:

<u>Number of children in a family:</u>

```
 "Statistical data / Sample"
 [1] 0 1 1 0 2 1 2 3 2 2 2 1 4 3 3
Variational series
 [1] 0 0 1 1 1 1 2 2 2 2 2 3 3 3 4
```

Average: 1.8

Size of sample: 15

The majority is from Q1 to Q3 rooms where Q1= 1 and Q3=2.5

FIVE NUMBER SUMMARY:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.0	1.0	2.0	1.8	2.5	4.0

The distribution law is:

NoC	0	1	2	3	4
Freq.	2	4	5	3	1

```
[1] "The most common number of children in a family is 2"
[1] "Having more than 2 children is more common."
```

<u>Number of Rooms:</u>

"Statistical data / Sample"

```
  [1] 1 2 2 3 3 3 3 2 2 1 1 3 1 2 4 1 1 2 2 2 1 2 2 2 1 2 1 3 2 2 3 1 2 2 1 2 2 1 3 3 2 2 1 2 1 4
3 2 2
 [50] 1 1 2 4 3 2 1 2 3 2 4 1 2 2 1 3 2 2 3 3 3 2 2 2 1 1 2 2 4 3 2 1 4 5 2 2 1 3 2 3 1 2 1 3 1
2 2 3 2
 [99] 1 1
```

Variational series

```
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [64] 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 5
```

Average: 2.08

Size of sample: 100

The majority is from Q1 to Q3 rooms where Q1= 1 and Q3=3

FIVE NUMBER SUMMARY:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.0	1.0	2.0	2.08	3	5.0

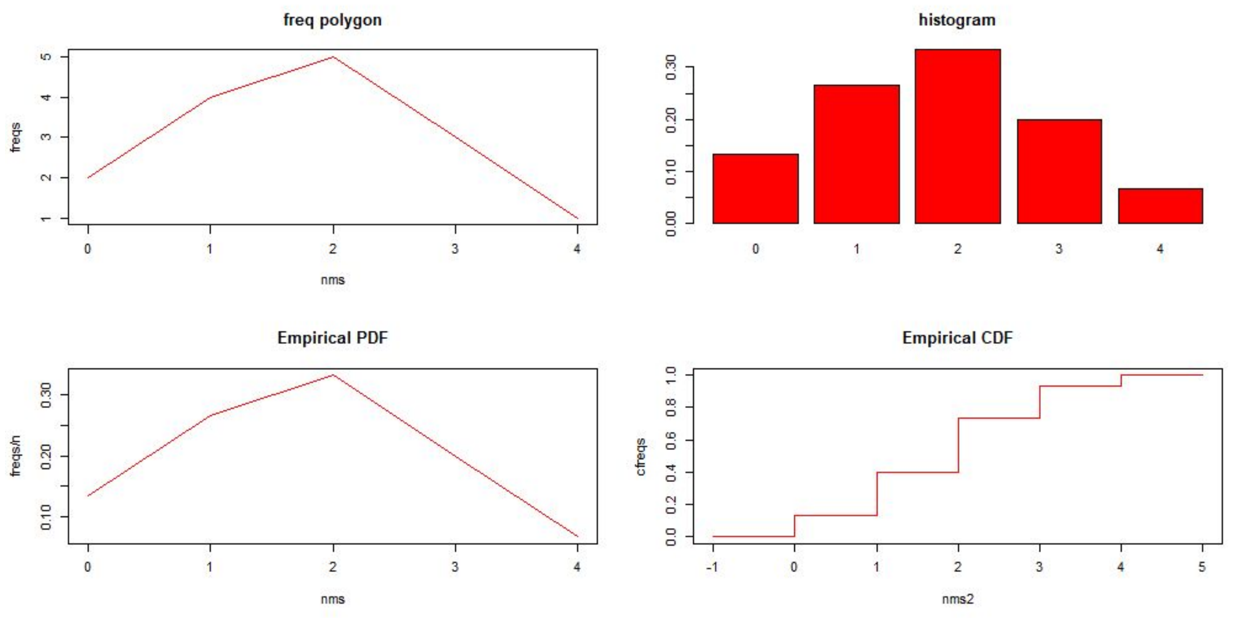
The distribution law is:

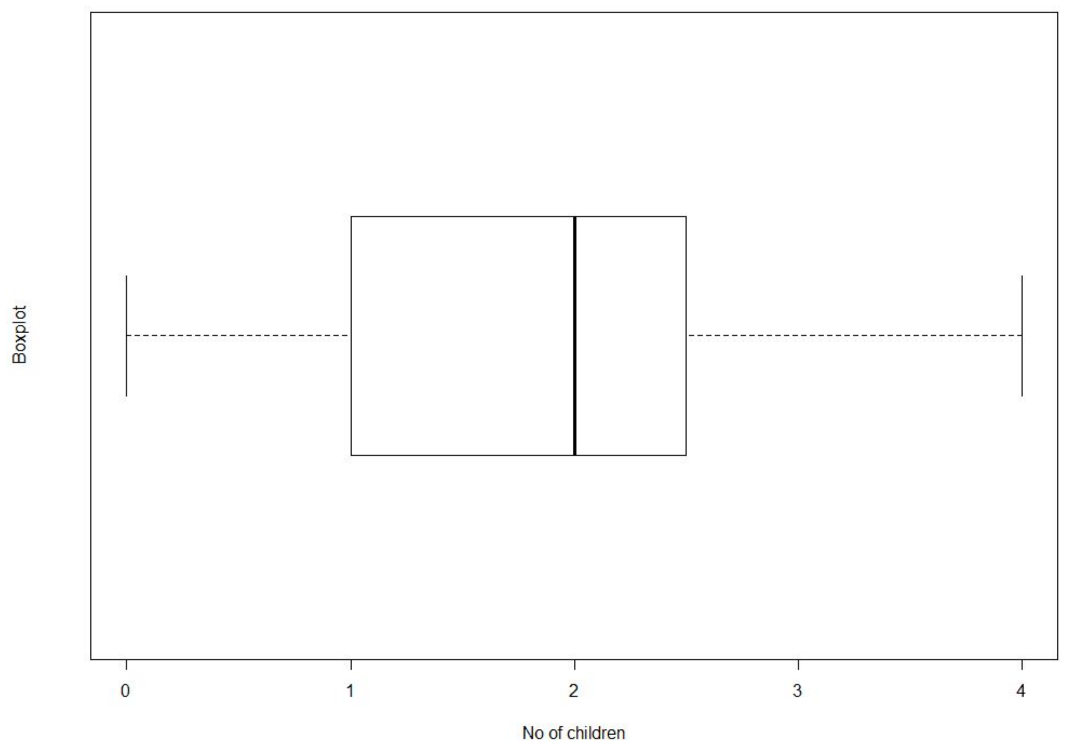
NoR	1	2	3	4	5
FreqRate	0.28	0.44	0.21	0.06	0.01

```
[1] "The most common number of rooms in flats is 2"
[1] "Having more than 2 rooms is more common."
```

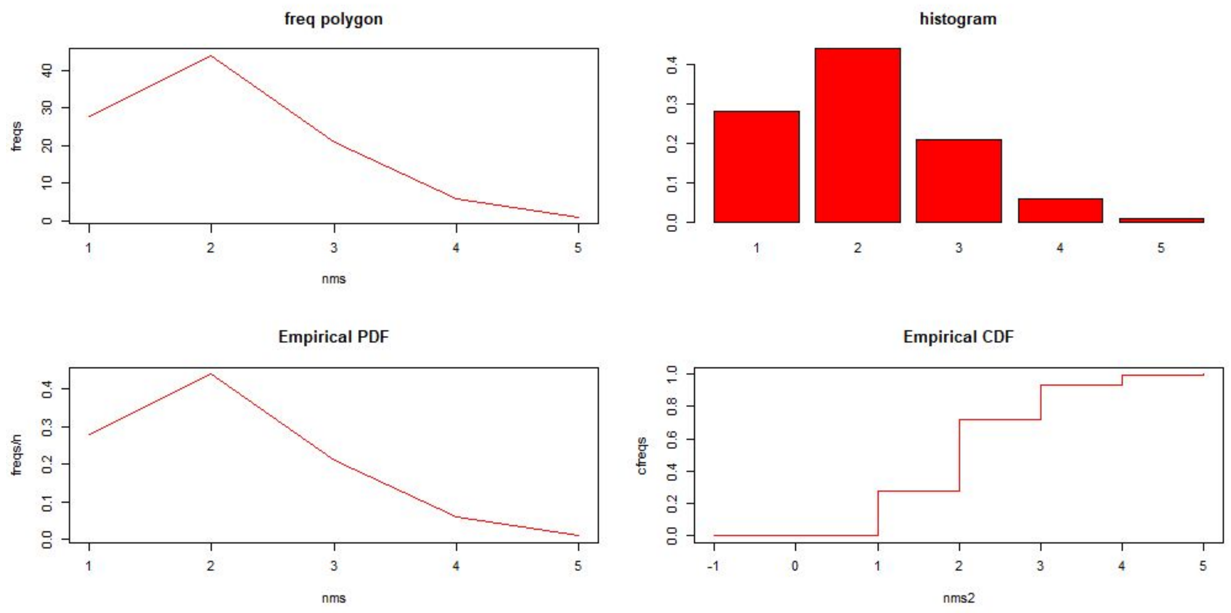
Figures:

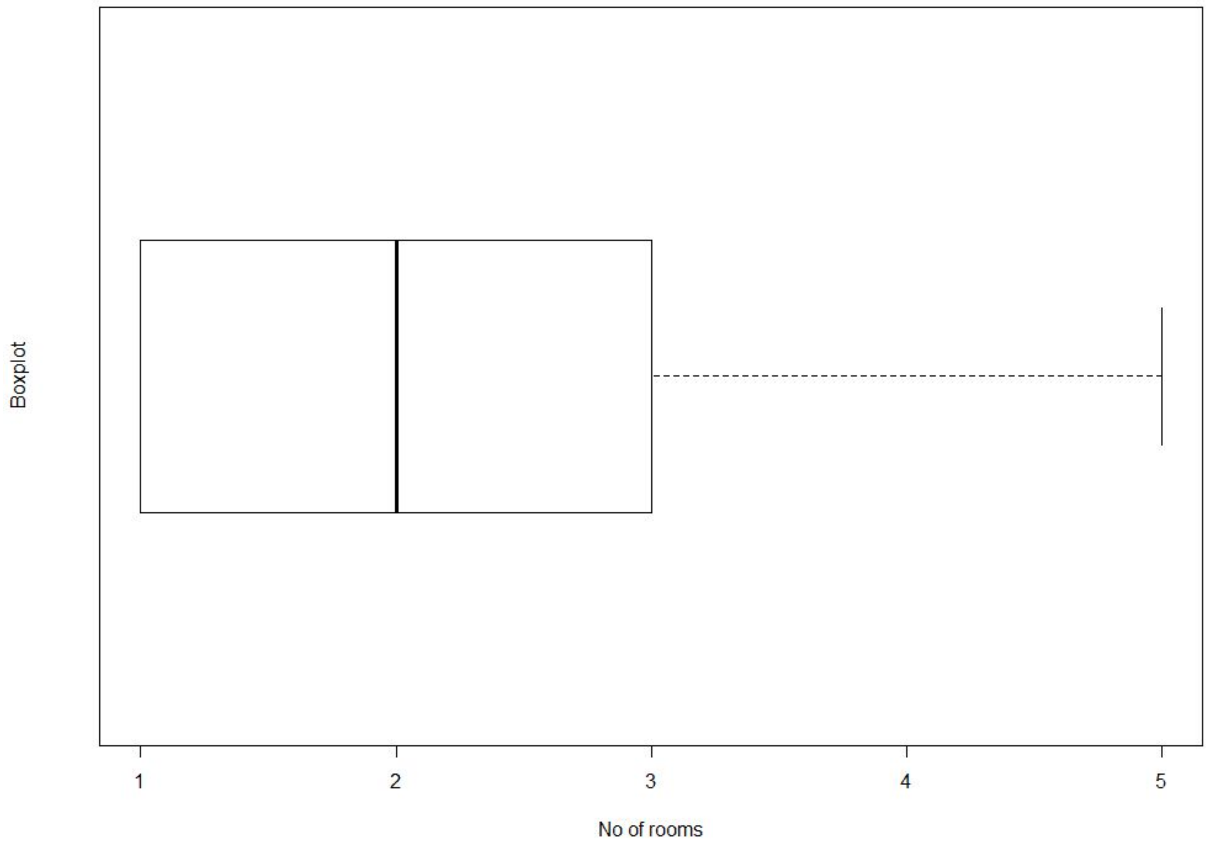
Number of Children:





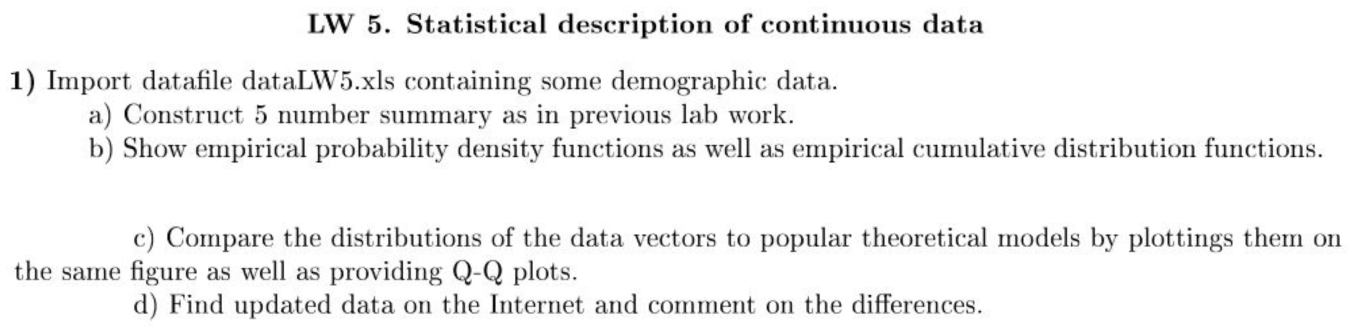
Number of Rooms:





[1] "The **boxplot** graphically summarizes the center and the spread of the data"

LAB 5



LW 5. Statistical description of continuous data

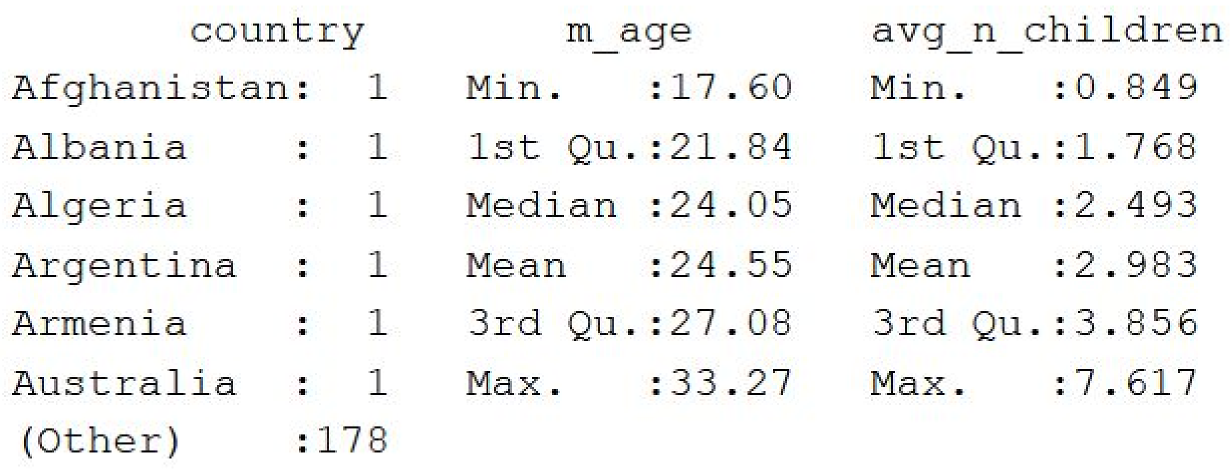
1) Import datafile dataLW5.xls containing some demographic data.
a) Construct 5 number summary as in previous lab work.
b) Show empirical probability density functions as well as empirical cumulative distribution functions.

c) Compare the distributions of the data vectors to popular theoretical models by plottings them on the same figure as well as providing Q-Q plots.
d) Find updated data on the Internet and comment on the differences.

For the updated data , I will use the data found on the CIA - World Factbook website.
And there was only data for the number of children.
We can see that Greece's number of children increased .Also that the updated data follows the Normal distribution with a twist on the left. And it does it better than the previous data did. Also we see that the biggest frequency is around 1.8 children and after that the next biggest frequency is way less than in the previous data. So, the line of the empirical PDF is smoother in the updated data with just one sharp moment; in comparison to 2 sharp moments of the previous data.

Output:

#FIVE NUMBER SUMMARY
> summary(D)



```
        country           m_age          avg_n_children 
 Afghanistan:  1   Min.   :17.60   Min.   :0.849  
 Albania    :  1   1st Qu.:21.84   1st Qu.:1.768  
 Algeria    :  1   Median :24.05   Median :2.493  
 Argentina  :  1   Mean   :24.55   Mean   :2.983  
 Armenia    :  1   3rd Qu.:27.08   3rd Qu.:3.856  
 Australia  :  1   Max.   :33.27   Max.   :7.617  
 (Other)    :178                                  
```

```
country            m_age    avg_n_children
62  Greece         26.94385           1.364

   country         m_age    avg_n_children
81   Japan         28.57219         1.307
```

CIA data:

```
no        country       avg_n_children
208       208  Greece            1.43
```

Program Code:

```
if(!require(gdata)){
  install.packages('gdata')
  library(gdata)
}



D <- read.csv(file='dataLW5.txt', sep='\t', dec=',', header=TRUE)
D<- D[,-3]

#FIVE NUMBER SUMMARY
summary(D)

#a) situation lithuania
id<- which(D$country=='Greece')
D_Lithuania <-D[id,]
D_Lithuania
id<- which(D$country=='Japan')
D_Japan <-D[id,]
D_Japan

#b) EMPIRICAL PDF and CDF
n<-nrow(D)
nbin<-1+3.3*log10(n)
```

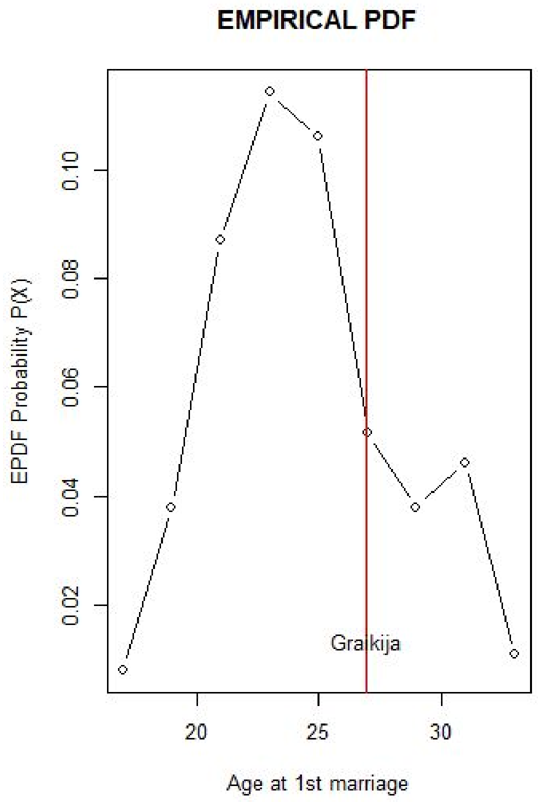
```
nbin<- 9
H<-hist(D$m_age,breaks="Sturges")
EPDF <- H$counts/(n*diff(H$breaks[1:2]))
ECDF <- cumsum(EPDF)*2
nms<-H$mids
layout(matrix(c(1,1,2,2),ncol=2))
plot(nms,EPDF,type='b',main='EMPIRICAL PDF',
        xlab='Age at 1st marriage',
        ylab='EPDF Probability P(X)')
abline(v=D_Lithuania$m_age,col='red')
lines(c(D_Lithuania$m_age,D_Lithuania$m_age),c(0,0.1),col='red')
text(D_Lithuania$m_age,0.013,'Graikija')
plot(nms,ECDF,type='b',main='EMPIRICAL CDF',
        xlab='Age at 1st marriage',
        ylab='ECDF Probability F(X),P(X < x)')
MeanA<-mean(D$m_age)
SA<-sd(D$m_age)
mA<-min(D$m_age)
MA<-max(D$m_age)
xs<-seq(mA,MA,(MA-mA)/100)
ps<-dnorm(xs,MeanA,SA)
plot(nms,EPDF,type='b',main='Theoretical PDF/Model(normal distribution)',
        xlab = 'Age at 1st marriage',
        ylab= 'Probability P(x)')
lines(xs,ps,col='blue')

#visual comparison of quantiles
qqnorm(scale(D$m_age))
lines((c(mA,MA)-MeanA)/SA,(c(mA,MA)-MeanA)/SA,type='l',col='green')


#now for number of children



#b) EMPIRICAL PDF and CDF
n<-nrow(D)
nbin<-1+3.3*log10(n)
nbin<- 9
H<-hist(D$avg_n_children,breaks="Sturges")
EPDF <- H$counts/(n*diff(H$breaks[1:2]))
ECDF <- cumsum(EPDF)*2
nms<-H$mids
layout(matrix(c(1,1,2,2),ncol=2))
plot(nms,EPDF,type='b',main='EMPIRICAL PDF',
        xlab='Number of Children',
        ylab='EPDF Probability P(X)')
abline(v=D_Lithuania$avg_n_children,col='red')
lines(c(D_Lithuania$avg_n_children,D_Lithuania$avg_n_children),c(0,0.1),col='red')
```

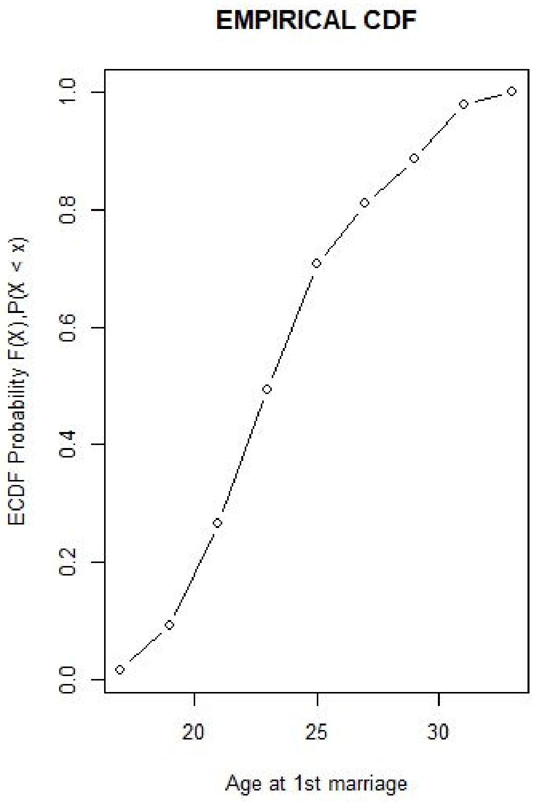
```
text(D_Lithuania$avg_n_children,0.05,'Graikija')
plot(nms,ECDF,type='b',main='EMPIRICAL CDF',
        xlab='Number of Children',
        ylab='ECDF Probability F(X),P(X < x)')
MeanA<-mean(D$avg_n_children)
SA<-sd(D$avg_n_children)
mA<-min(D$avg_n_children)
MA<-max(D$avg_n_children)
xs<-seq(mA,MA,(MA-mA)/100)
ps<-dnorm(xs,MeanA,SA)
plot(nms,EPDF,type='b',main='Theoretical PDF/Model(normal distribution)',
        xlab = 'Number of Children',
        ylab= 'Probability P(x)')
lines(xs,ps,col='blue')

#visual comparison of quantiles
qqnorm(scale(D$avg_n_children))
lines((c(mA,MA)-MeanA)/SA,(c(mA,MA)-MeanA)/SA,type='l',col='green')

#now for updated data found on CIA-World Factbook website
#updated data for number of children
D <- read.csv(file='dataCIA.txt', sep="", dec='.', header=TRUE)

id<- which(D$country=='Greece')
D_Lithuania <-D[id,]
D_Lithuania

n<-nrow(D)
nbin<-1+3.3*log10(n)
nbin<- 9
H<-hist(D$avg_n_children,breaks="Sturges")
EPDF <- H$counts/(n*diff(H$breaks[1:2]))
ECDF <- cumsum(EPDF)*2
nms<-H$mids
layout(matrix(c(1,1,2,2),ncol=2))
plot(nms,EPDF,type='b',main='EMPIRICAL PDF',
        xlab='Number of Children',
        ylab='EPDF Probability P(X)')
abline(v=D_Lithuania$avg_n_children,col='red')
lines(c(D_Lithuania$avg_n_children,D_Lithuania$avg_n_children),c(0,0.1),col='red')
text(D_Lithuania$avg_n_children,0.13,'Graikija')
plot(nms,ECDF,type='b',main='EMPIRICAL CDF',
        xlab='Number of Children',
        ylab='ECDF Probability F(X),P(X < x)')
MeanA<-mean(D$avg_n_children)
SA<-sd(D$avg_n_children)
mA<-min(D$avg_n_children)
MA<-max(D$avg_n_children)
```

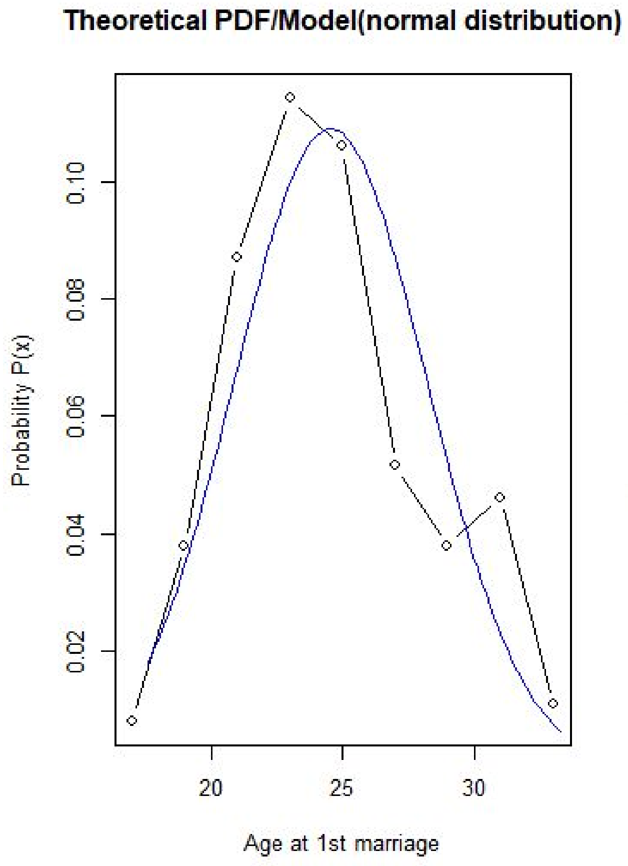
```
xs<-seq(mA,MA,(MA-mA)/100)
ps<-dnorm(xs,MeanA,SA)
plot(nms,EPDF,type='b',main='Theoretical PDF/Model(normal distribution)',
        xlab = 'Number of children',
        ylab= 'Probability P(x)')
lines(xs,ps,col='blue')

#visual comparison of quantiles
qqnorm(scale(D$avg_n_children))
lines((c(mA,MA)-MeanA)/SA,(c(mA,MA)-MeanA)/SA,type='l',col='green')
```

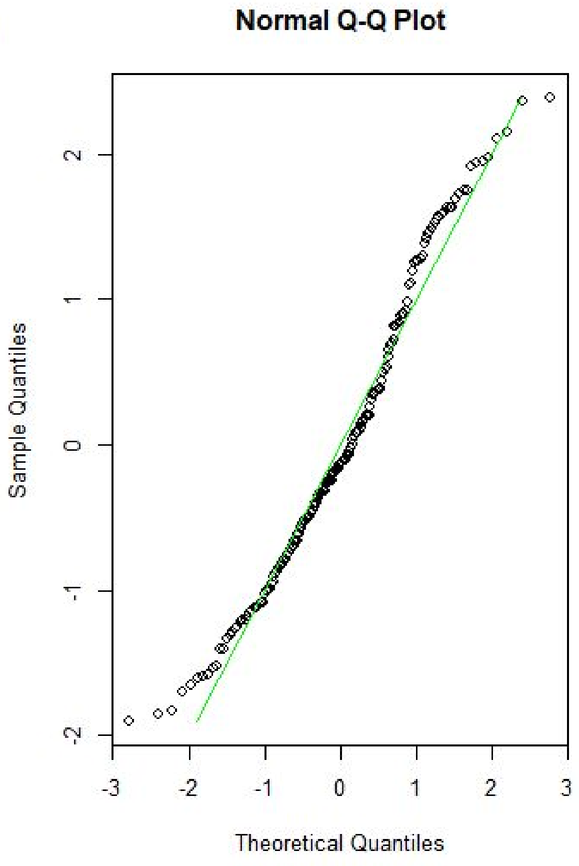
Figures:

<u>Age at 1st Marriage:</u>

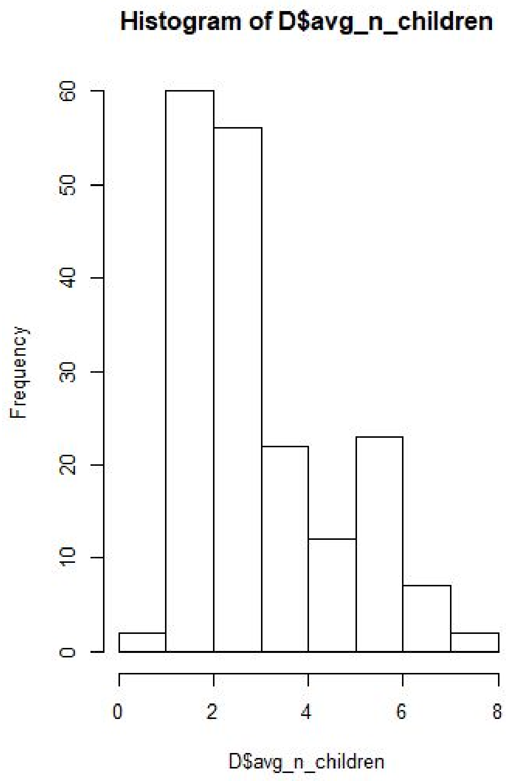


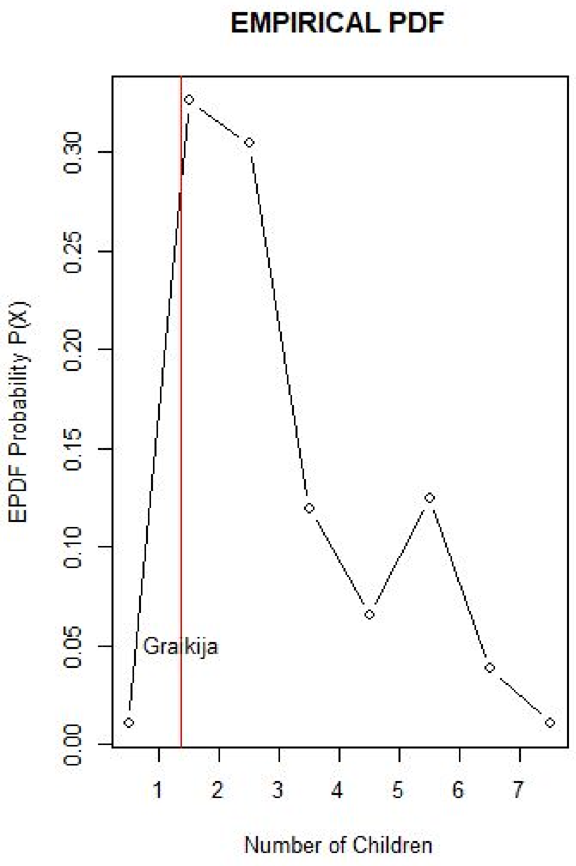


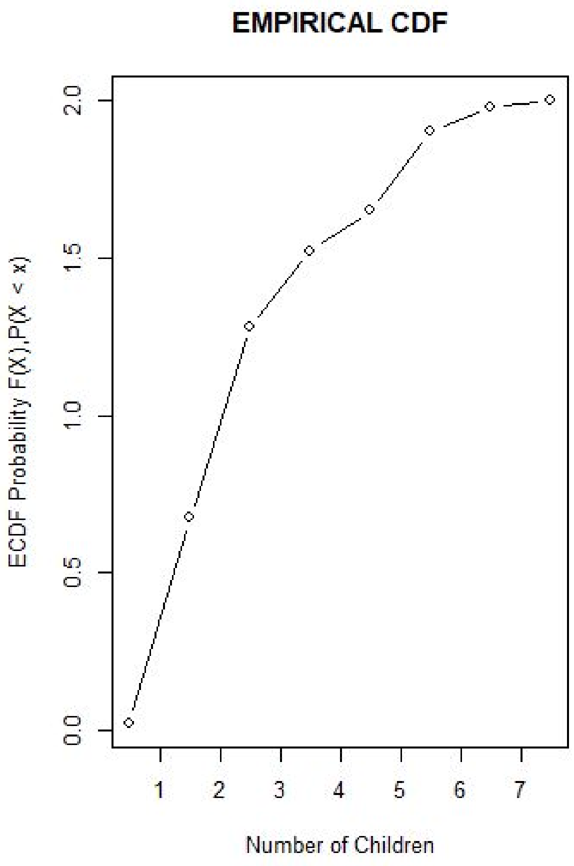


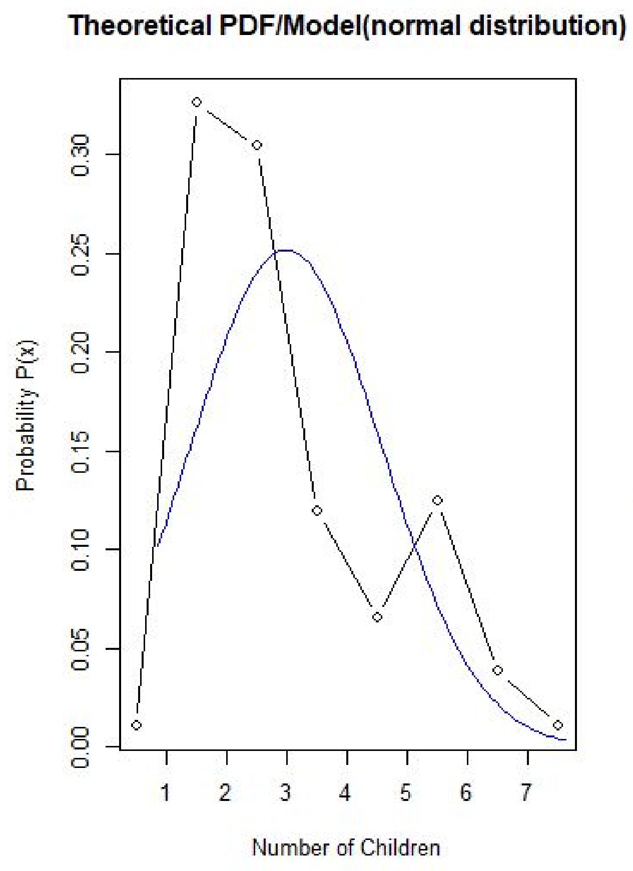


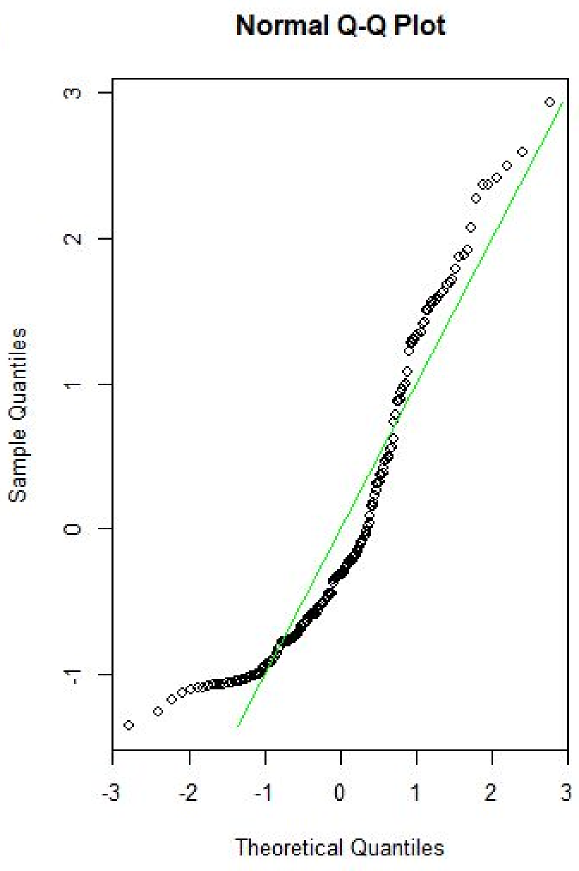
Number of Children:



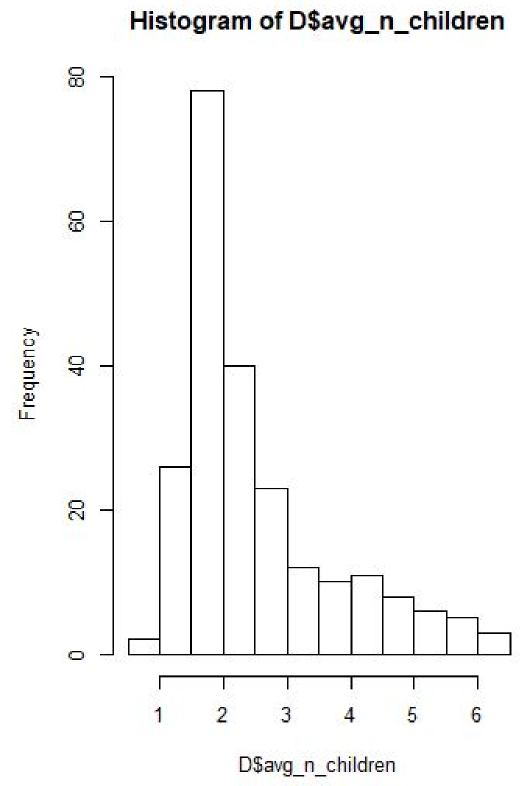


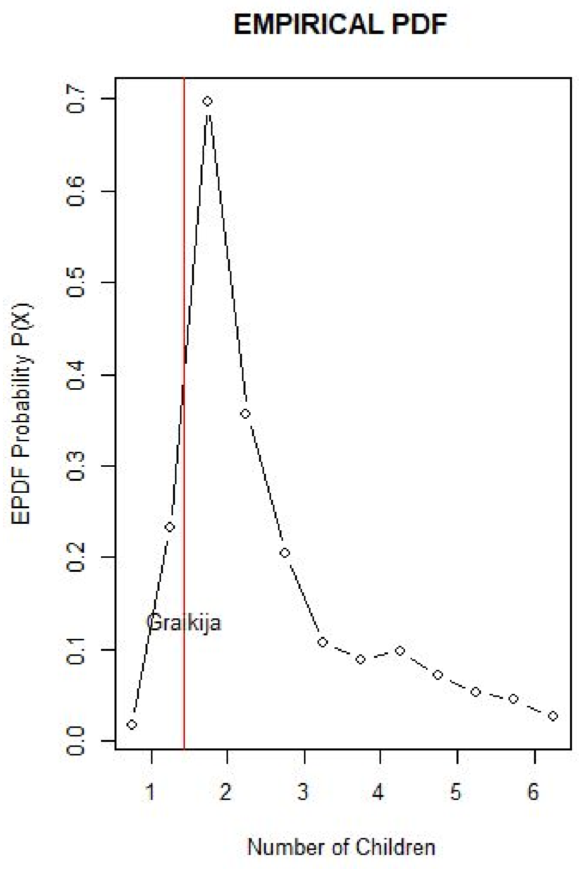


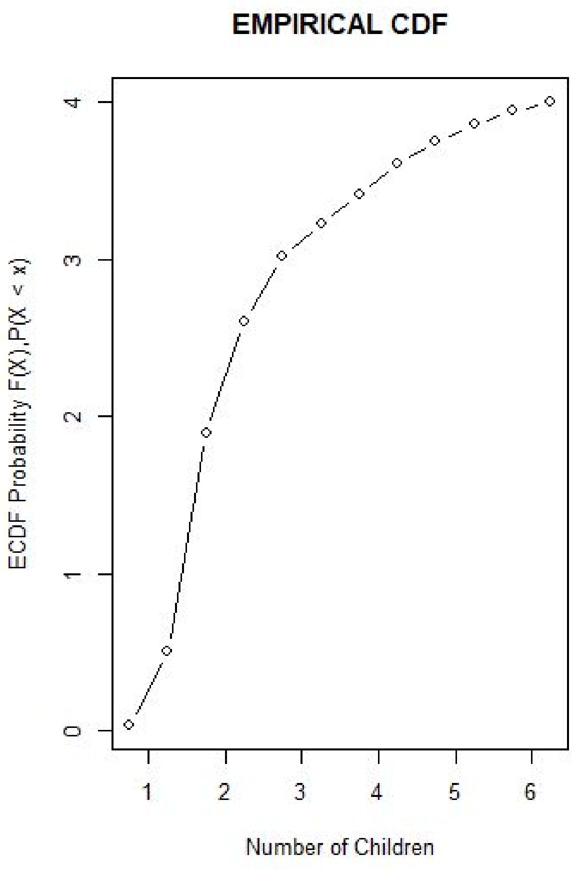

Theoretical PDF/Model(normal distribution)
Probability P(x)
0.00
0.05
0.10
0.15
0.20
0.25
0.30
1
2
3
4
5
6
7
Number of Children

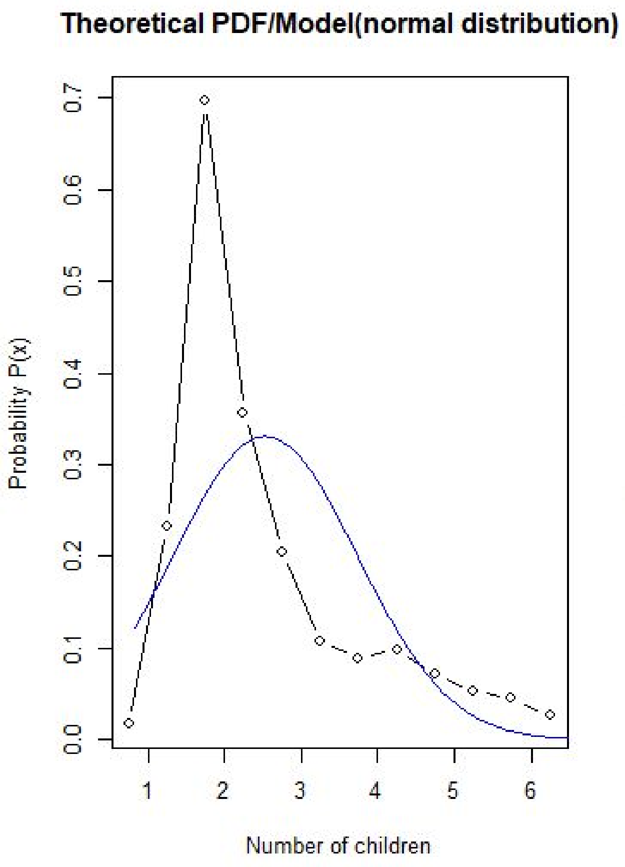

Normal Q-Q Plot
Sample Quantiles
-1
0
1
2
3
-3
-2
-1
0
1
2
3
Theoretical Quantiles

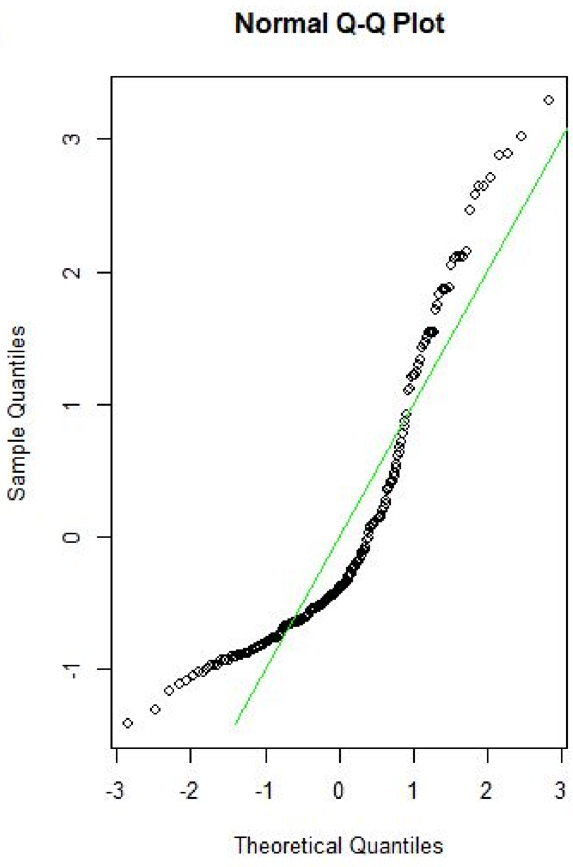
CIA Data - Number of Children:








Theoretical PDF/Model(normal distribution)
Probability P(x)
Number of children
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
1
2
3
4
5
6


Normal Q-Q Plot
Sample Quantiles
Theoretical Quantiles
-1
0
1
2
3
-3
-2
-1
0
1
2
3

LAB 6

LW 6. Hypotheses testing

1) Work with the same data file dataLW5.xls.
- a) Perform goodness-of-fit test for normality.
- b) Check hypotheses about the average values for each data vector.
- c) Find 95% confidence intervals for the average values.
- d) Try to check hypotheses and determine confidence limits for other parameters.

Output:

Age:

```
Pearson's Chi-squared test

data:  aadm_normalized and dnorm(aadm_normalized)
X-squared = 33488, df = 33124, p-value = 0.07903

One-sample Kolmogorov-Smirnov test

data:  aadm_normalized
D = 0.075968, p-value = 0.2387
alternative hypothesis: two-sided

     Shapiro-Wilk normality test

data:  aadm_normalized
W = 0.97238, p-value = 0.001041
```

Number of children:

```
     Pearson's Chi-squared test

data:  aadm_normalized and dnorm(nooc_normalized)
X-squared = 32016, df = 31850, p-value = 0.2549

One-sample Kolmogorov-Smirnov test

data:  nooc_normalized
D = 0.1588, p-value = 0.0001866
alternative hypothesis: two-sided
```

```
Shapiro-Wilk normality test

data:  nooc_normalized
W = 0.88971, p-value = 2.047e-10

> message(paste("[",t1,"; ",t2,"]", sep=""))
[-1.97301191513627; 1.97301191513626]
> message(Tn)



13.1485481947046
> #Tn is outside ->reject H0
> #with 0.95 confidence average age at first marriage is 24.2

> message(paste("[",t1,"; ",t2,"]", sep=""))
[-1.97301191513627; 1.97301191513626]
> message(Tn)
-0.148438899255863
> #Tn is inside ->do not reject H0
> #with p=0.95

with confidence of 95 % the true average Number of children is in
[2.75237999292365; 3.2129678331633]

with confidence of 80 % the true average Number of children is in
[2.83254666823414; 3.13280115785282]
```

Program Code:

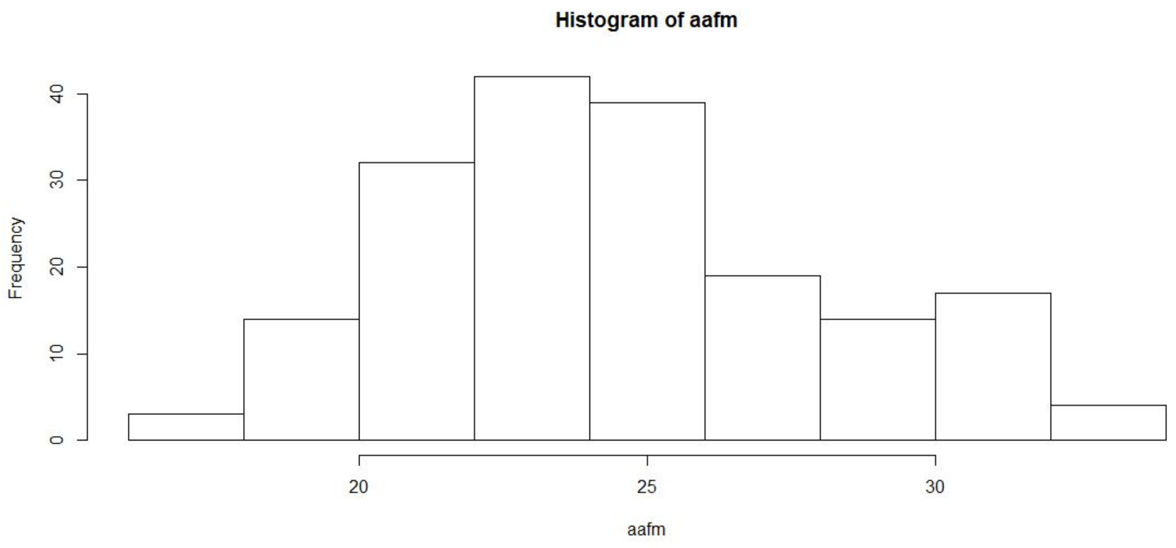
```
D <- read.csv(file='dataLW5.txt', sep='\t', dec=',', header=TRUE)
df <- D[,-3]

aafm<-df$m_age
nooc<-df$avg_n_children

# a)
hist(aafm)
M_aafm<-mean(aafm)
S_aafm<-sd(aafm)
aadm_normalized<-as.numeric(scale(aafm))

#H0 :aafm_normalized ~N(0,1)
#H1 :aafm_normalized ~/~N(0,1)

#type 1 Stat error(alpha):
#rejection of a true null hypothesis
#type 2 Stat error(beta):
#failure to reject a false null hypothesis


#Universal
chisq.test(aadm_normalized,dnorm(aadm_normalized))
#0.07903>0.05
#not rejecting normality

#in general as gof for normality
ks.test(aadm_normalized,"pnorm")
#0.2387>0.05
#p>alpha - more confidence to reject H0

#dedicated far more to normal cases
shapiro.test(aadm_normalized)
#0.001041 <0.05/0.01 /0.10
#H0 must be rejected

#goodness.fit(),AdequacyModel
#gofTest(),gofTest

#Data is normal(alpha 0.05).
#Data is not normal(alpha 0.001).

hist(nooc)
nooc_normalized<-as.numeric(scale(nooc))
```

```
#H0 :nooc_normalized ~N(0,1)
#H1 :nooc_normalized ~/~N(0,1)

#type 1 Stat error(alpha):
#rejection of a true null hypothesis
#type 2 Stat error(beta):
#failure to reject a false null hypothesis


#Universal
chisq.test(aadm_normalized,dnorm(nooc_normalized))
#0.2549>0.05
#reject

#in general as gof for normality
ks.test(nooc_normalized,"pnorm")
#0.00018>0.05
#not rejefct

#dedicated far more to normal cases
shapiro.test(nooc_normalized)
#2.047e-10 <0.05
#not rejected


#b)

#H0:aafm=21
#H1:aafm=/= 21
M<-mean(aafm)
S<-sd(aafm) #for sample
n<-length(aafm)
Tn<-(M-21)*sqrt(n)/S
t1<-qt(0.025,n-1)
t2<-qt(0.975,n-1)
message(paste("[",t1,"; ",t2,"]", sep=""))
message(Tn)
#Tn is outside ->reject H0
#with 0.95 conf avg aafm is 24.2

#H0:nooc=3
#H1:nooc=/= 3
M<-mean(nooc)
S<-sd(nooc) #for sample
n<-length(nooc)
```

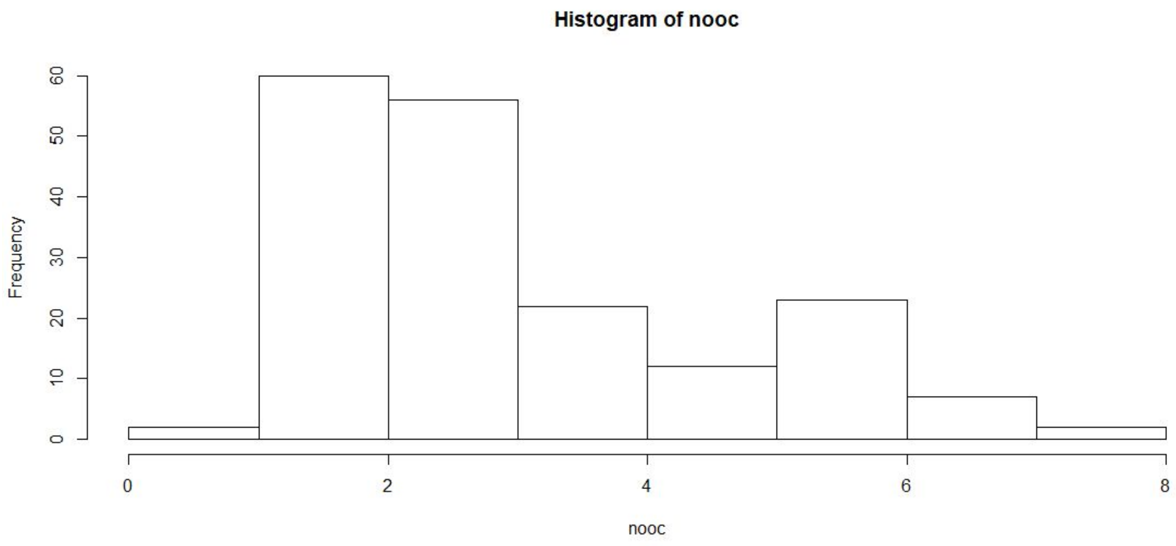
```
Tn<-(M-3)*sqrt(n)/S
t1<-qt(0.025,n-1)
t2<-qt(0.975,n-1)
message(paste("[",t1,"; ",t2,"]", sep=""))
message(Tn)
#Tn is inside ->dont reject H0
#with p=0.95

#c)
alpha<-0.05
M<-mean(nooc)
S<-sd(nooc)
n<-length(nooc)
Tn<-(M-3)*sqrt(n)/S
t1<-qt(alpha/2,n-1)
t2<-qt(1-alpha/2,n-1)
e1<-t1*S/sqrt(n)
e2<-t2*S/sqrt(n)
CI<-(paste("[",M+e1,"; ",M+e2,"]", sep=""))
LB<-paste('with confidence of',100*(1-alpha),' % the true average is in ',CI,sep ='')
message(LB)
#with confidence of 95% th true average
#no of children is in
#[2.7523;3.2129]

alpha<-0.2
M<-mean(nooc)
S<-sd(nooc)
n<-length(nooc)
Tn<-(M-3)*sqrt(n)/S
t1<-qt(alpha/2,n-1)
t2<-qt(1-alpha/2,n-1)
e1<-t1*S/sqrt(n)
e2<-t2*S/sqrt(n)
CI<-(paste("[",M+e1,"; ",M+e2,"]", sep=""))
LB<-paste('with confidence of',100*(1-alpha),' % the true average is in ',CI,sep ='')
message(LB)
#with confidence of 80% th true average
#no of children is in


#d)
# H0: D_aafm =15
# H1: D_aafm =/=15

alpha<-0.05
```

```
M<-mean(nooc)
S<-sd(nooc)
n<-length(nooc)
X2n<-(n-1)*S^2/15
X21<-qchisq(alpha/2,n-1)
X22<-qchisq(1-alpha/2,n-1)
#X2n is inside ->we dont reject H0 with chosen alpha

# H0: D_aafm >=15
# H1: D_aafm <15

alpha<-0.05
M<-mean(nooc)
S<-sd(nooc)
n<-length(nooc)
X2n<-(n-1)*S^2/15
X2<-qchisq(alpha,n-1)
# reject H0

#CI for dispersion
alpha<-0.05
M<-mean(nooc)
S<-sd(nooc)
n<-length(nooc)
X22mp<-qchisq(1-alpha/2,n-1)
X21mp<-qchisq(alpha/2,n-1)
LCL <- n*S^2/X22mp
UCL <-n*S^2/X21mp
```

Figures:



Histogram of age



Histogram of Number of Children