Untitled

GitHub Documents

This is an R Markdown format used for publishing markdown documents to GitHub. When you click the **Knit** button all R code chunks are run and a markdown file (.md) suitable for publishing to GitHub is generated.

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

3.1 Calculator ToDo

Compute the difference between 2014 and the start year at this university and divide this by the difference between 2014 and the year you were born. Multiply this with 100 to get the percentage of your life you have spent at this university.

```
(2016-2014)/(2014-1978)*100
```

```
## [1] 5.555556
```

3.2 Workspace ToDo

Repeat the previous ToDo, but with several steps in between. You can give the variables any name you want, but the name has to start with a letter

```
seneca<-2016
year<-1994
born <-1978
multip <-100
(seneca - year)/(year-born)*multip
```

```
## [1] 137.5
```

3.4 Function ToDo

Compute the sum of 4, 5, 8 and 11 by ???rst combining them into a vector and then using the function sum.

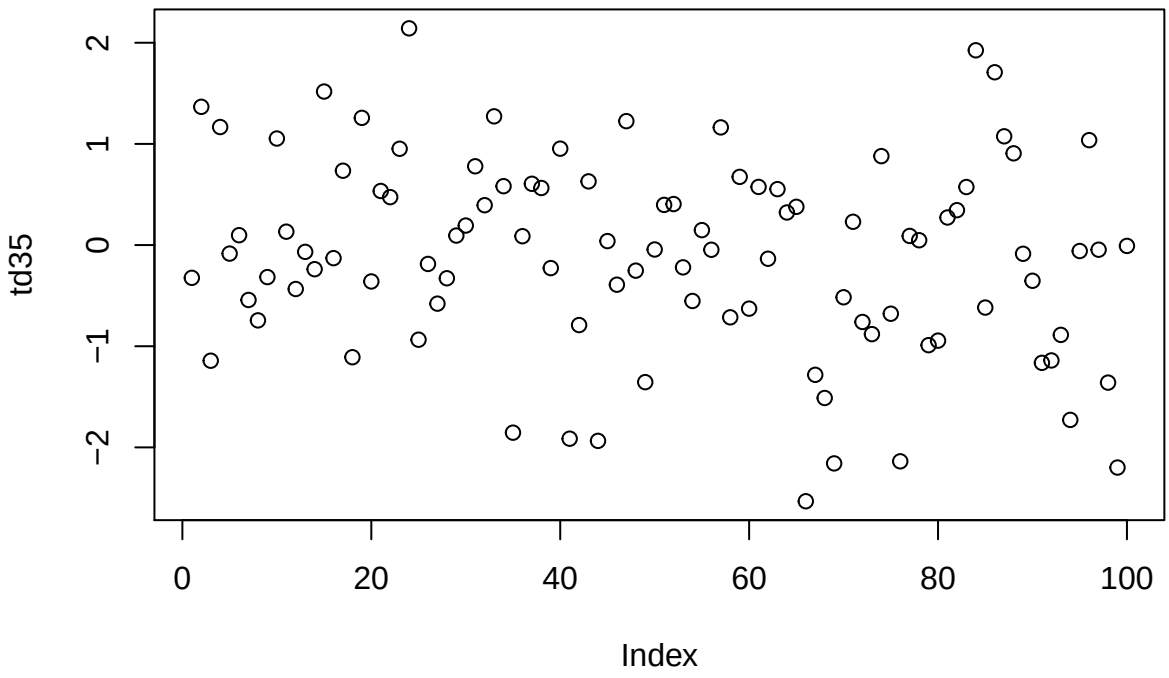
```
td34<-c(4,5,8,11)
sum(td34)
```

```
## [1] 28
```

3.5 Plots ToDo

Plot 100 normal random numbers.

```
td35<-rnorm(100,0,1)
plot(td35)
```



4 Help and documentation ToDo

Find help for the sqrt function.

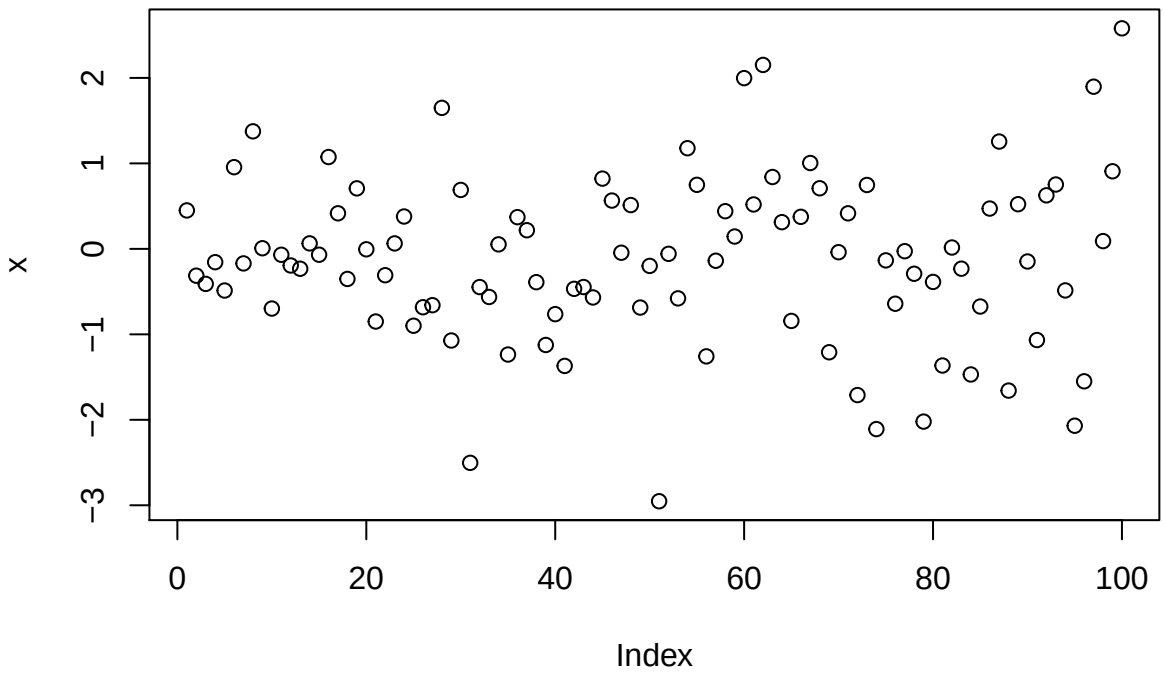
```
help(sqrt)
```

```
## starting httpd help server ... done
```

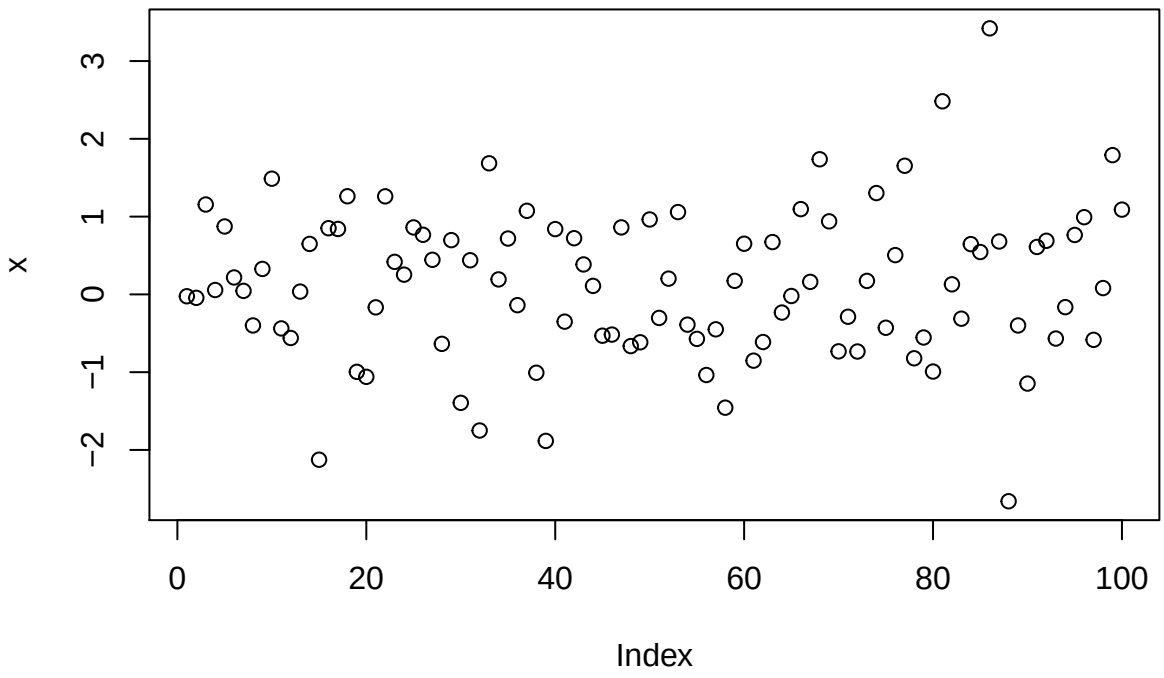
5 Scripts ToDo

Make a ???le called firstscript.R containing Rcode that generates 100 random numbers and plots them, and run this script several times.

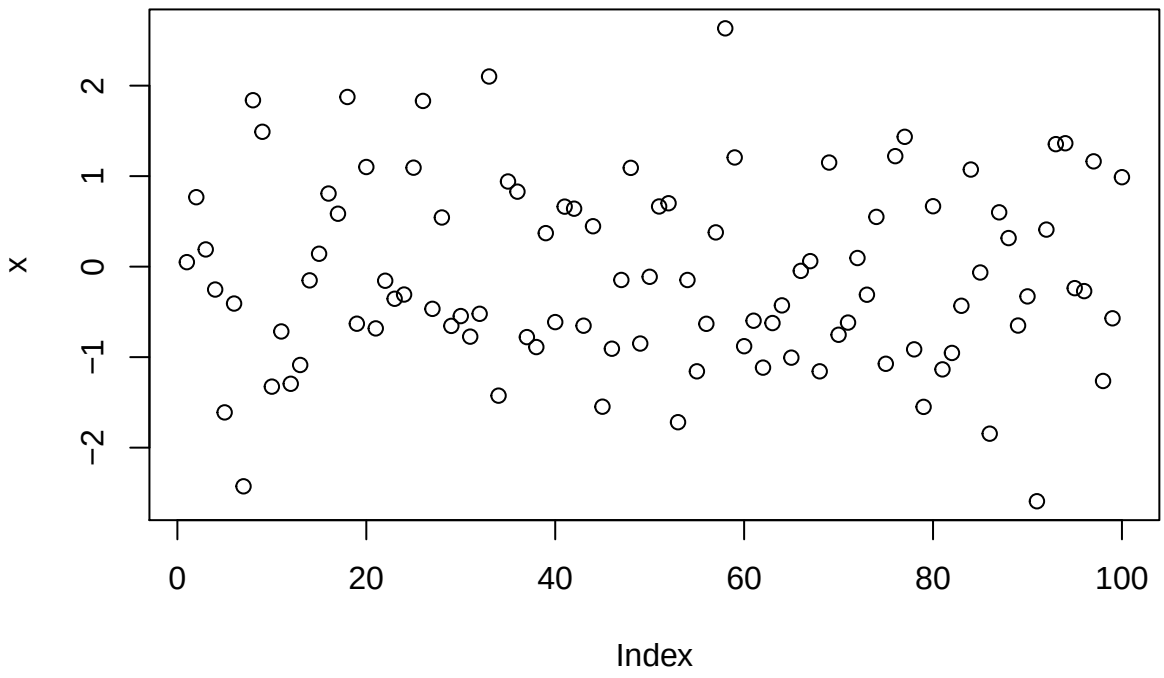
```
source("firstscript.R")
```




```
source("firstscript.R")
```




```
source("firstscript.R")
```



6.2 Matrices ToDo

Put the numbers 31 to 60 in a vector named P and in a matrix with 6 rows and 5 columns named Q. Tip: use the function seq. Look at the di???erent ways scalars, vectors and matrices are denoted in the workspace window

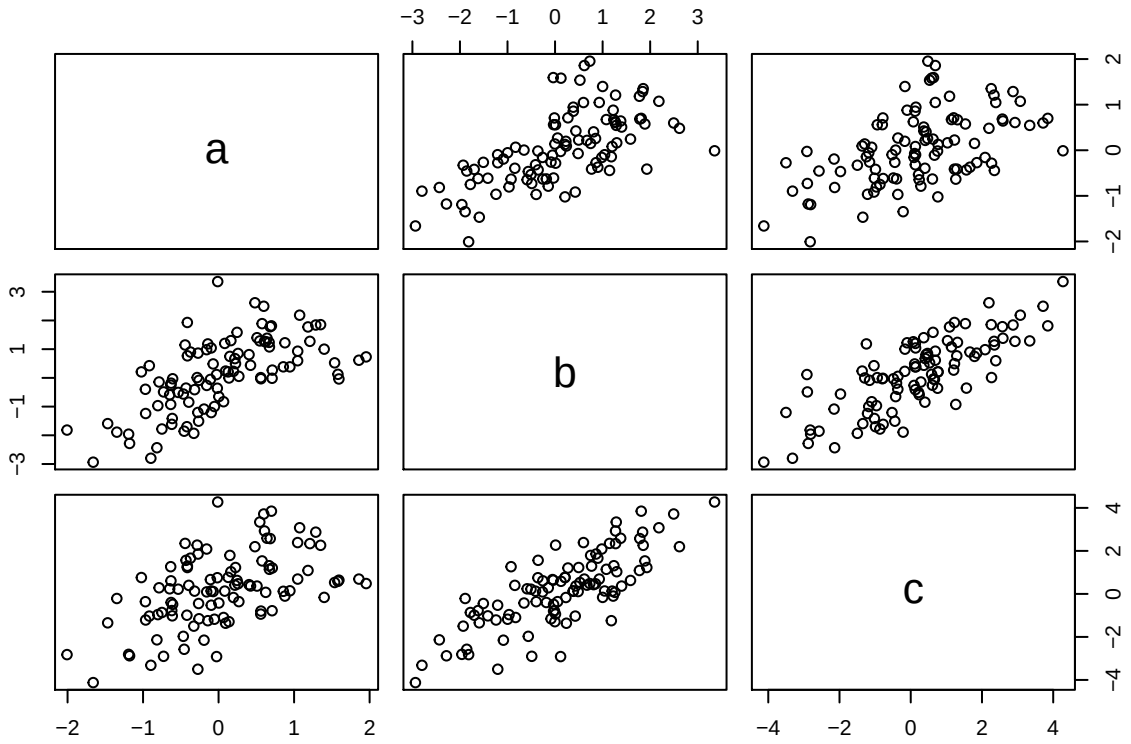
```
P=seq(from=31,to=60,by=1)
Q=matrix(data=c(31:60),ncol = 5,nrow = 6)
Q
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   31   37   43   49   55
## [2,]   32   38   44   50   56
## [3,]   33   39   45   51   57
## [4,]   34   40   46   52   58
## [5,]   35   41   47   53   59
## [6,]   36   42   48   54   60
```

6.3 Data Frames ToDo

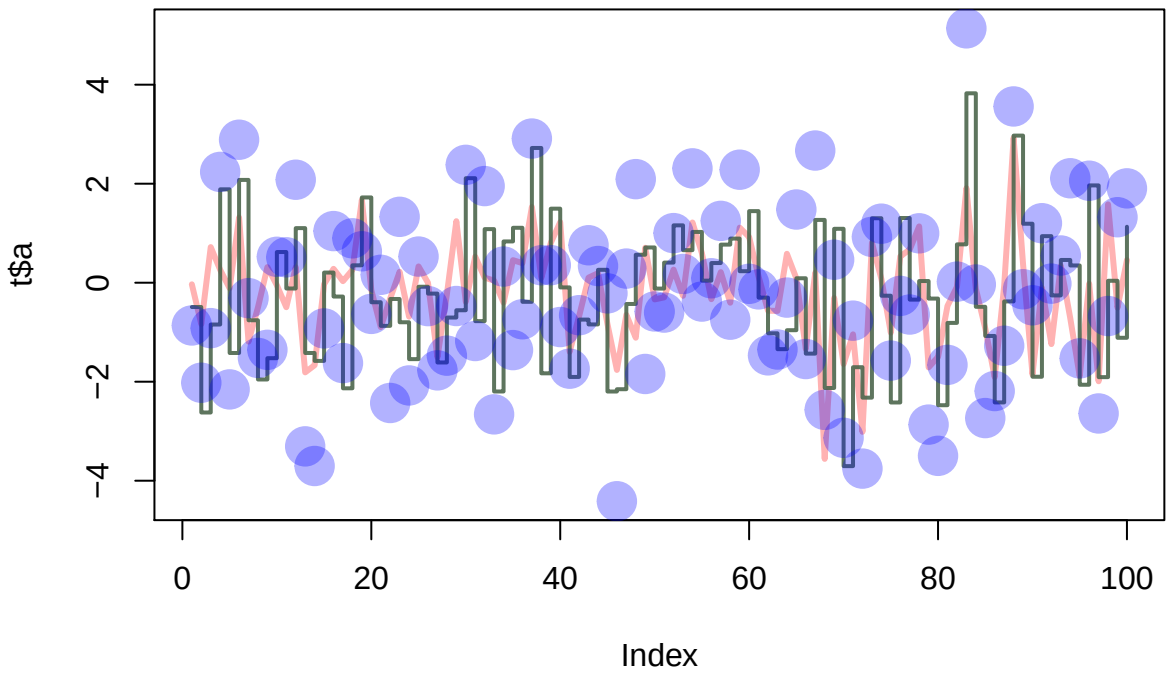
Make a script ???le which constructs three random normal vectors of length 100. Call these vectors x1, x2 and x3. Make a data frame called t with three columns (called a, b and c) containing respectively x1, x1+x2 and x1+x2+x3. Call the following functions for this data frame: plot(t) and sd(t). Can you understand the results? Rerun this script a few times.

```
x1 = rnorm(100)
x2 = rnorm(100)
x3 = rnorm(100)
t = data.frame(a=x1,b=x1+x2,c=x1+x2+x3)
plot(t)
```



7 Graphics ToDo Add these lines to the script ???le of the previous section. Try to ???nd out, either by experimenting or by using the help, what the meaning is of rgb, the last argument of rgb, lwd, pch, cex.

```
x1 = rnorm(100)
x2 = rnorm(100)
x3 = rnorm(100)
t = data.frame(a=x1,b=x1+x2,c=x1+x2+x3)
plot(t$a, type="l", ylim=range(t), lwd=3, col=rgb(1,0,0,0.3))
lines(t$b, type="s", lwd=2, col=rgb(0.3,0.4,0.3,0.9))
points(t$c, pch=20, cex=4, col=rgb(0,0,1,0.3))
```



RGB: Red Green Blue value of color (first 3 column are color intensity , and last is the color alfa work like transparency) Pch: S-compatible vector symbols , asci character, vector symbole, depend on value cex: character (or symbol) expansion: a numerical vector. This works as a multiple of par("cex"). lwd: line width for drawing symbols see

8 Reading and writing data files ToDo

Make a ???le called tst1.txt in Notepad from the example in Figure 4 and store it in your working directory. Write a script to read it, to multiply the column called g by 5 and to store it as tst2.txt

```
d1<-read.table("tst1.txt",header=TRUE)
d2<-data.frame(a=c(1,2,4,8,16,32),g=c(2,4,8,16,32,64),x=c(3,6,12,24,48,96))
(d3<-d1*matrix(c(1,1,1,1,1,1,5,5,5,5,5,5,1,1,1,1,1,1)))
```

```
##    a   g  x
## 1  1  10  3
## 2  2  20  6
## 3  4  40 12
## 4  8  30 24
## 5 16 160 48
## 6 32 320 96
```

```
write.table(d3,file="tst2.txt",row.names=FALSE)
```

8.1 Reading and writing data files ToDo

only storing the 2nd column in tst3.txt

```
d = read.table(file = "tst1.txt", header=TRUE)
d1 = data.frame(a<-c(1,2,4,8,16,32),g<-c(2,4,8,16,32,64),x<-c(3,6,12,24,48,96))
d2 <-d1$g*5
write.table(d2, file = "tst3.txt",row.names=FALSE)
d2
```

```
## [1]  10  20  40  80 160 320
```

9 Not Available data ToDo

Compute the mean of the square root of a vector of 100 random numbers. What happens?

```
td9 <- rnorm(100)
#or td9<-c(rnorm(100))
mean(sqrt(td9))
```

```
## Warning in sqrt(td9): NaNs produced
```

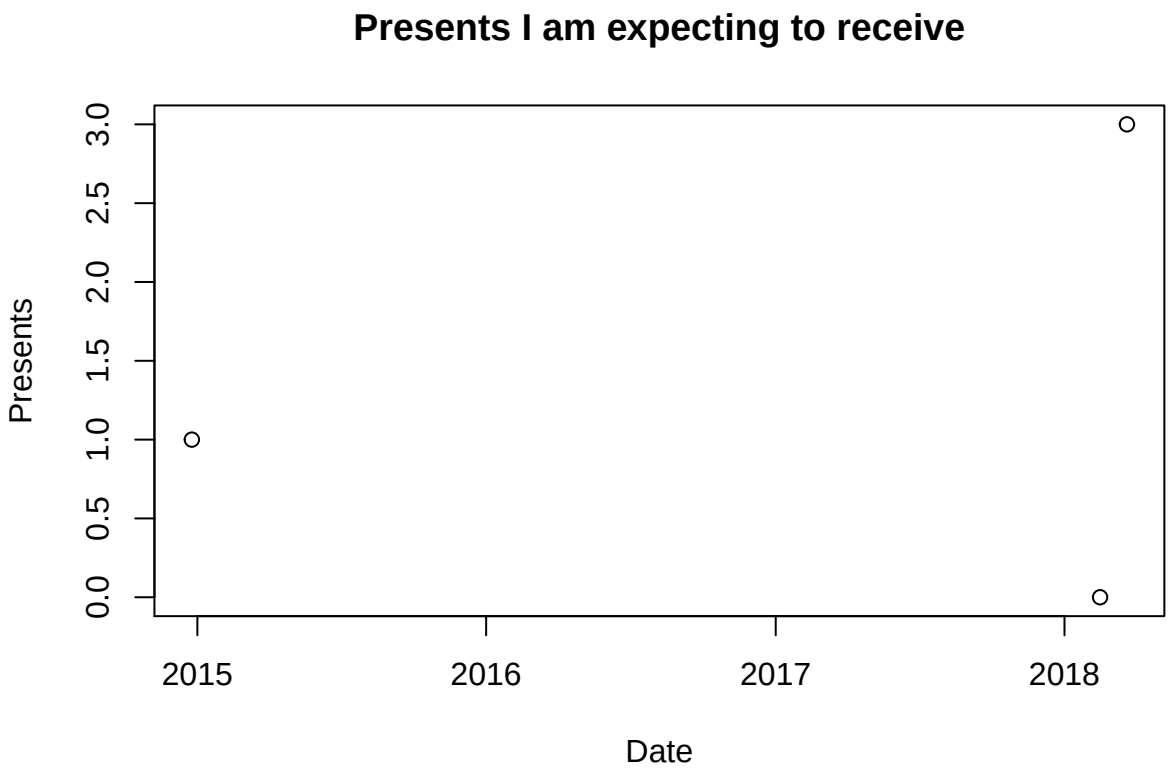
```
## [1] NaN
```

"NaNs produced", some negative value is appear when generate 100 random number.To solve the problem, use script

10.2 Dates ToDo

Make a graph with on the x-axis: today, Sinterklaas 2014 and your next birthday and on the y-axis the number of presents you expect on each of these days. Tip: make two vectors ???rst

```
data1<-strptime(c("20180215","20141225","20180321"),format="%Y%m%d")
data2=c(0,1,3)
plot(data1,data2,xlab="Date",ylab="Presents",main="Presents I am expecting to receive")
```



11.2 For-loop ToDo

Make a vector from 1 to 100. Make a for-loop which runs through the whole vector. Multiply the elements which are smaller than 5 and larger than 90 with 10 and the other elements with 0.1

```
h = seq(from=1,to=100)
s = c()
for(i in 1:100)
 {
  if (i<5 | i >90)
   {
    s[i]=h[i] * 10
   }
  else{
   s[i]=h[i]*0.1
   }
}
s
```

```
##   [1]   10.0   20.0   30.0   40.0    0.5    0.6    0.7    0.8    0.9    1.0
##  [11]    1.1    1.2    1.3    1.4    1.5    1.6    1.7    1.8    1.9    2.0
##  [21]    2.1    2.2    2.3    2.4    2.5    2.6    2.7    2.8    2.9    3.0
##  [31]    3.1    3.2    3.3    3.4    3.5    3.6    3.7    3.8    3.9    4.0
##  [41]    4.1    4.2    4.3    4.4    4.5    4.6    4.7    4.8    4.9    5.0
##  [51]    5.1    5.2    5.3    5.4    5.5    5.6    5.7    5.8    5.9    6.0
##  [61]    6.1    6.2    6.3    6.4    6.5    6.6    6.7    6.8    6.9    7.0
```

```
##  [71]    7.1    7.2    7.3    7.4    7.5    7.6    7.7    7.8    7.9    8.0
##  [81]    8.1    8.2    8.3    8.4    8.5    8.6    8.7    8.8    8.9    9.0
##  [91]  910.0  920.0  930.0  940.0  950.0  960.0  970.0  980.0  990.0 1000.0
```

11.3 Writing your own function ToDo

Write a function for the previous ToDo, so that you can feed it any vector you like (as argument). Use a for-loop in the function to do the computation with each element. Use the standard R function length in the speci???cation of the counter. a)

```
k=1:10
fun = function(arg1)
{
  l = length(arg1)
  for(i in 1:l)
  {
    if (arg1[i] < 5 | arg1[i] > 90)
    {
      arg1[i] = arg1[i] * 10
    } else
    {
      arg1[i] = arg1[i] * 0.1
    }
  }
  return (arg1)
}
fun(arg1=k)
```

```
##  [1] 10.0 20.0 30.0 40.0  0.5  0.6  0.7  0.8  0.9  1.0
```