Assignment 0

Daniel Lichtblau & Mingxuan Luo

February 12, 2018

The purpose of this assignment is to complete the 'ToDo' assignments in the article 'A (very) short introduction to R' (i.e. link provided below). The codes and results were compiled in RStudio and then knitted into this HTML file using 'RMarkdown' and the 'knitr' package.

A (very) short introduction to R: https://cran.r-project.org/doc/contrib/Torfs+Brauer-Short-R-Intro.pdf

SOURCES: 1]: https://www.dataquest.io/blog/how-to-share-data-science-portfolio/ 2]: https://rmarkdown.rstudio.com/ 3]: https://nicercode.github.io/guides/reports/ 4]: http://kbroman.org/knitr_knutshell/pages/markdown.html 5]: http://kbroman.org/knitr_knutshell/pages/Rmarkdown.html 6]: https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf 7]: https://github.com/cyberis/TidyDataProject 8]: https://www.r-bloggers.com/r-markdown-and-knitr-tutorial-part-1/ 9]: https://www.r-bloggers.com/r-markdown-and-knitr-tutorial-part-2/ 10]: https://www.r-bloggers.com/r-markdown-and-knitr-tutorial-part-3/ 11]: https://www.r-bloggers.com/r-markdown-and-knitr-tutorial-part-4/

3.1: Compute the difference between 2017 and the year you started 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. Use brackets if you need them.

```
((2018 - 2016) / (2018 - 1990)) * 100
```

```
## [1] 7.142857
```

3.2: 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.

```
schoolYears <- (2018 - 2016)
lifeYears <- (2018 - 1990)
percentage <- schoolYears / lifeYears
answer <- percentage * 100
answer
```

```
## [1] 7.142857
```

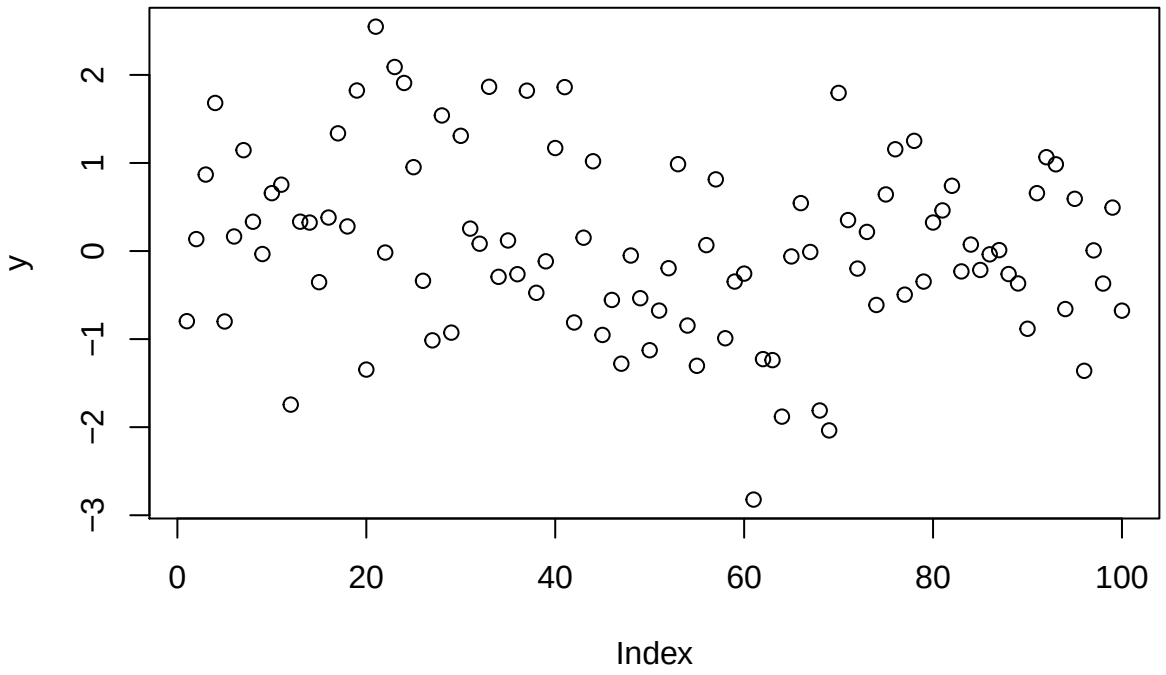
3.4: Compute the sum of 4, 5, 8 and 11 by first combining them into a vector and then using the function sum.

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

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

3.5: Plot 100 normal random numbers.

```
y = rnorm(100)
plot(y)
```

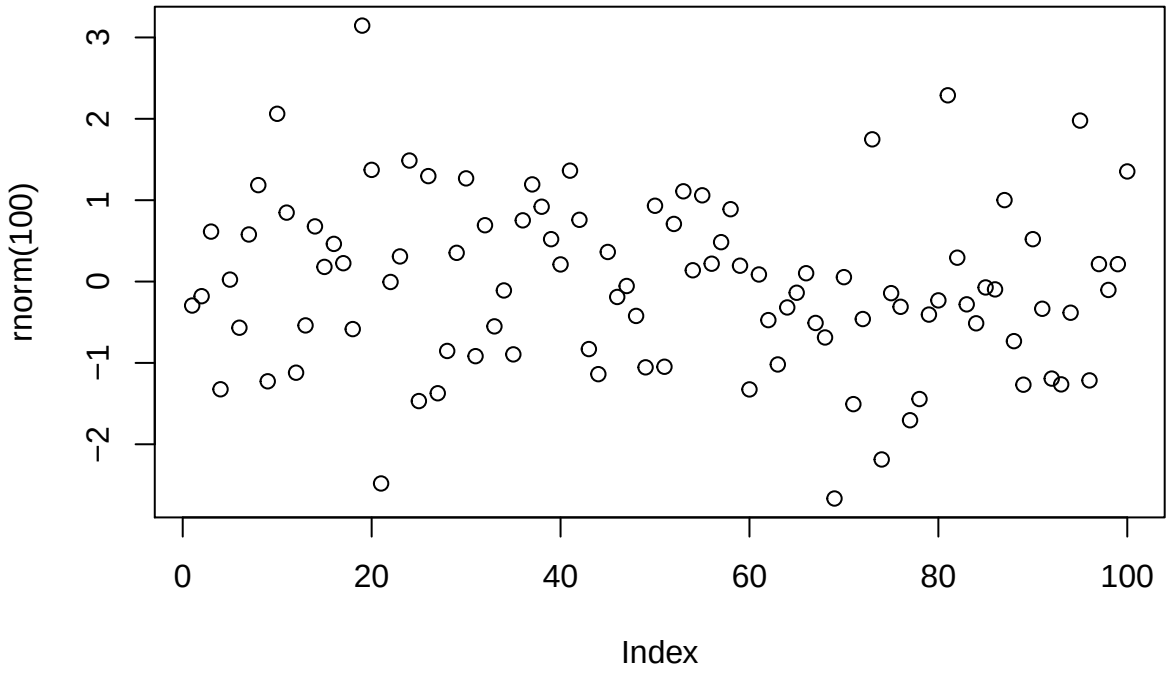


4.0: Find help for the 'sqrt' function.

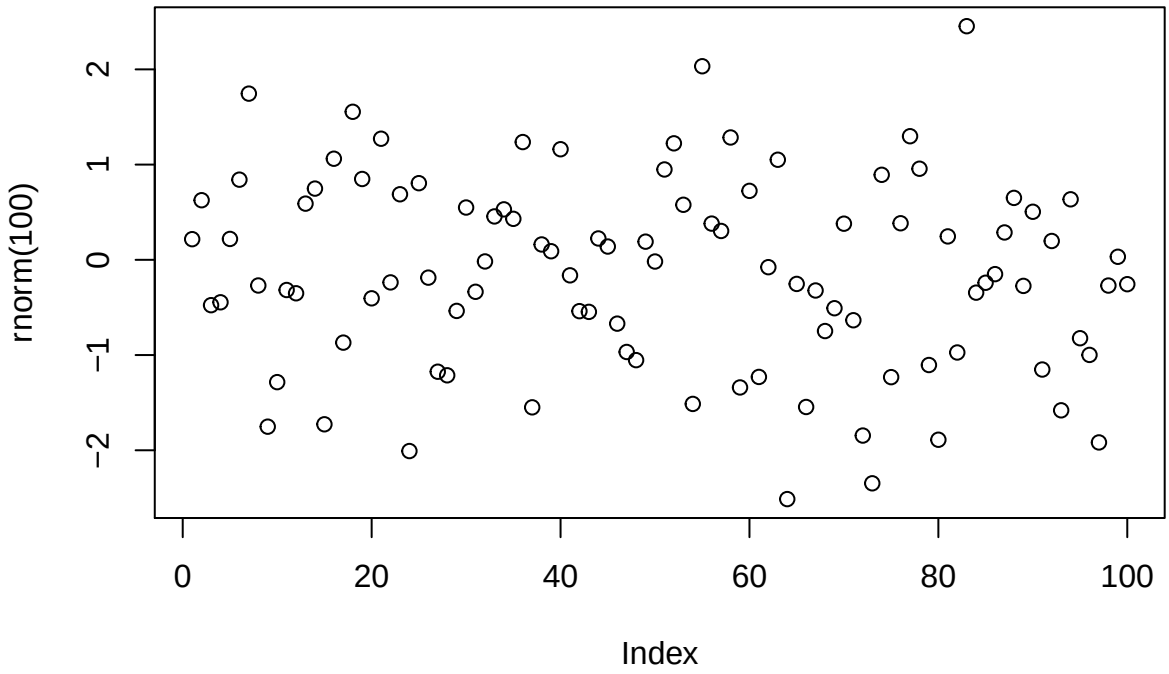
```
help(sqrt)
```

5.0: Make a file called 'firstscript.R' containing R-code that generates 100 random numbers and plots them, and run this script several times.

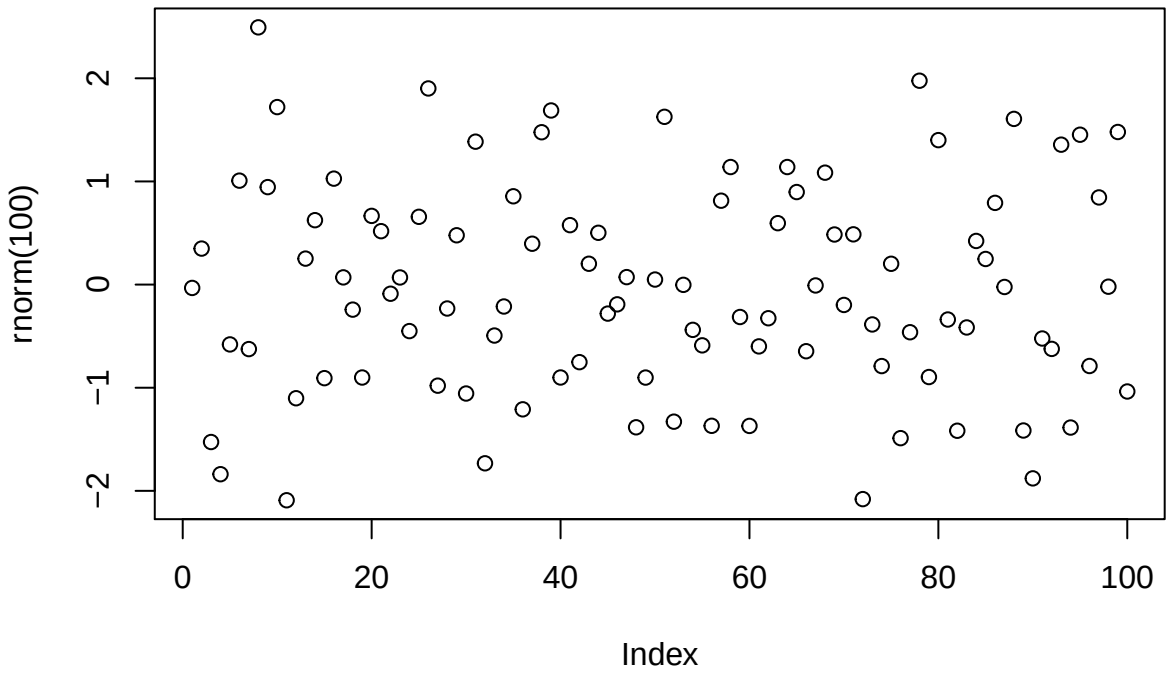
```
source("/home/dlichtblau/Desktop/SRT411/assign0/TODOs_screenshots/5.0/firstscript.R")
```



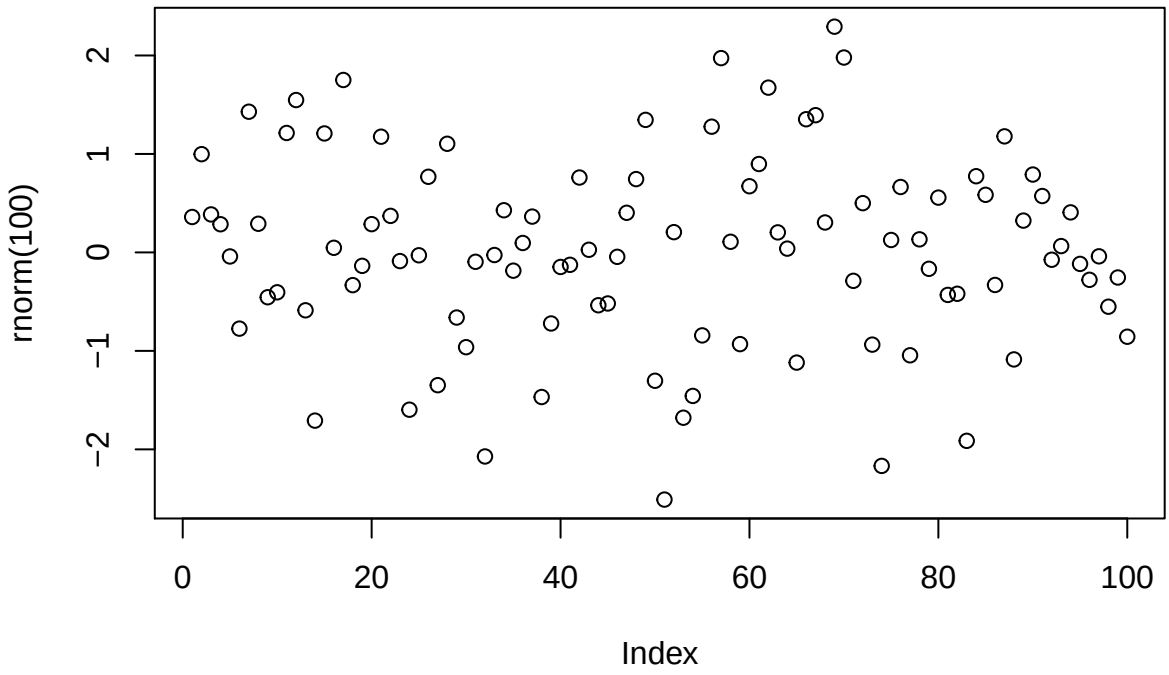
```
source("/home/dlichtblau/Desktop/SRT411/assign0/TODOs_screenshots/5.0/firstscript.R")
```



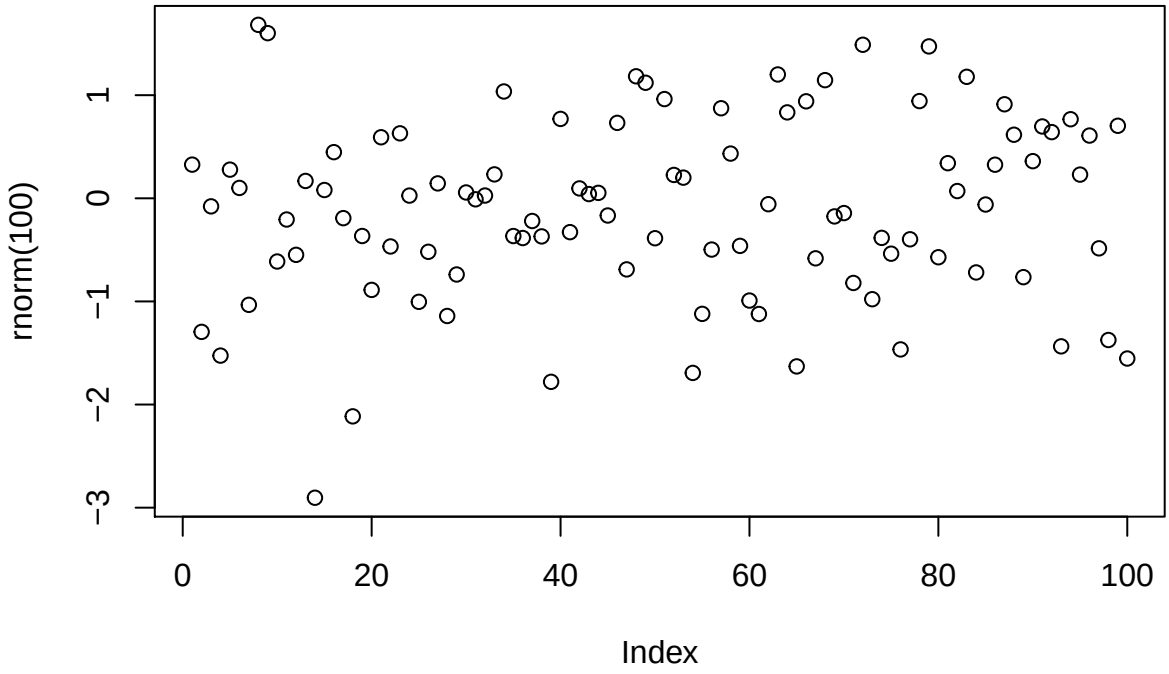

```
source("/home/dlichtblau/Desktop/SRT411/assign0/TODOs_screenshots/5.0/firstscript.R")
```




```
source("/home/dlichtblau/Desktop/SRT411/assign0/TODOs_screenshots/5.0/firstscript.R")
```




```
source("/home/dlichtblau/Desktop/SRT411/assign0/TODOs_screenshots/5.0/firstscript.R")
```



6.2: 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 different ways scalars, vectors and matrices are denoted in the workspace window.

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

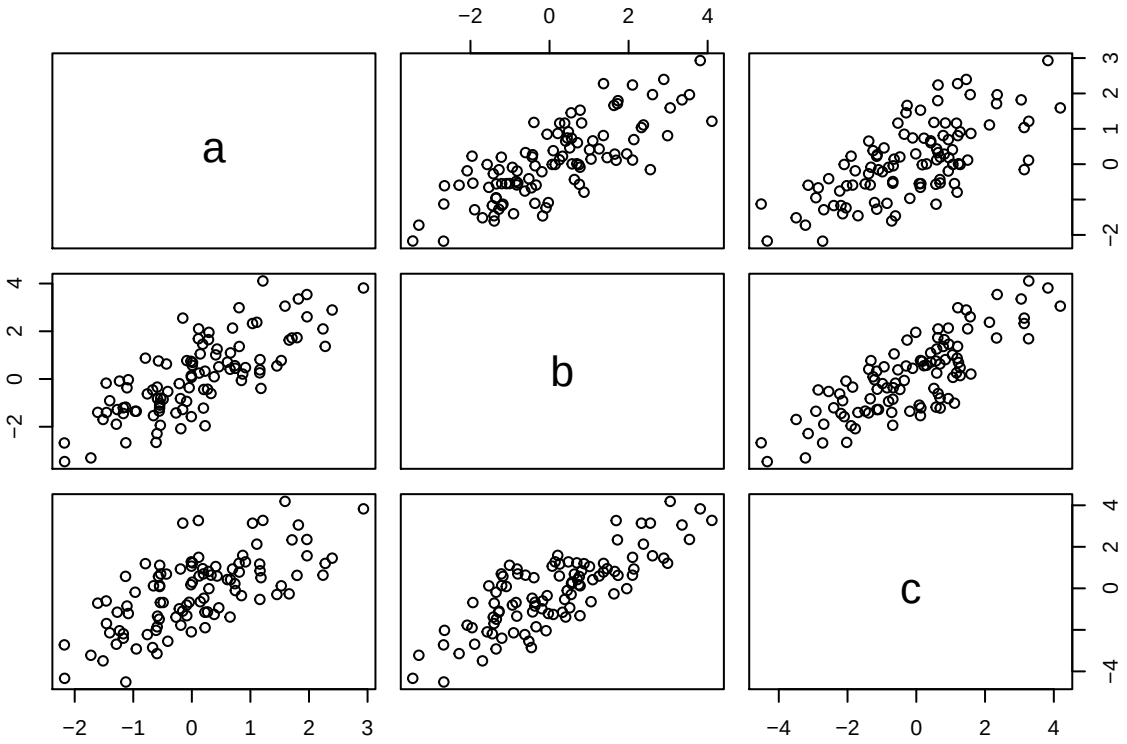
```
##  [1] 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
## [24] 54 55 56 57 58 59 60
```

```
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: Make a script file 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 <- c(rnorm(100))
x2 <- c(rnorm(100))
x3 <- c(rnorm(100))
t = data.frame(a = c(x1), b = c(x1+x2), c = c(x1+x2+x3))
plot(t)
```

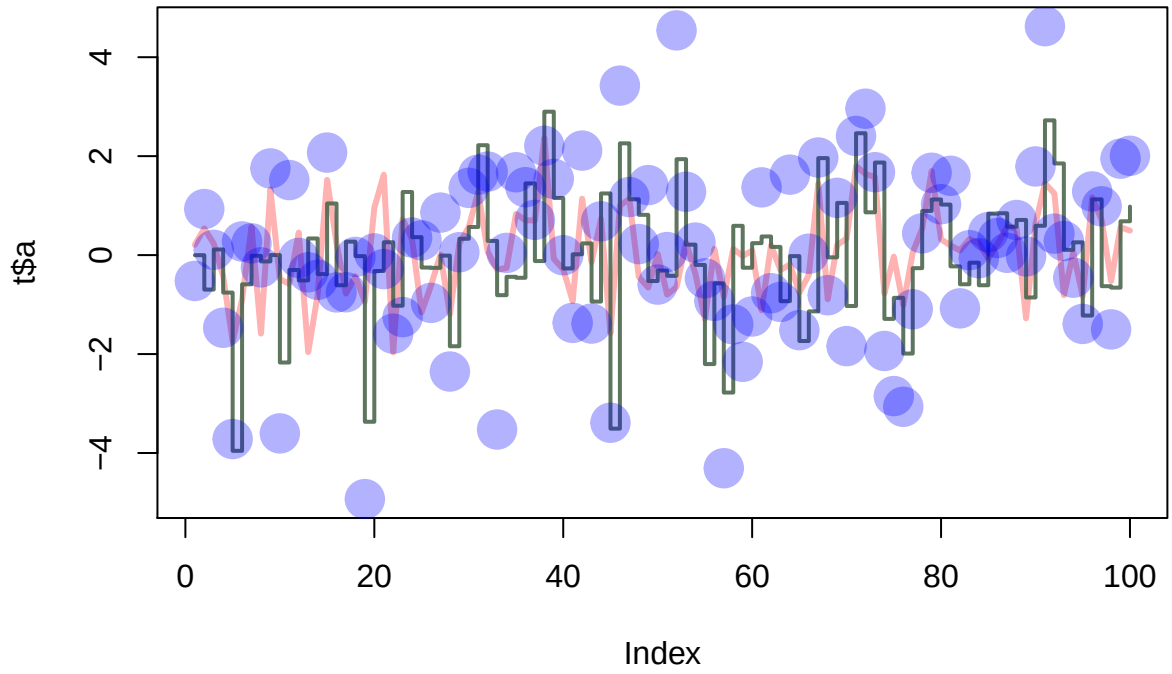



```
#sd(t)
```

7.0: Add these lines to the script file of the previous section. Try to find out, either by experimenting or by using the help, what the meaning is of rgb, the last argument of rgb, lwd, pch, cex.

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



8.0: Make a file 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'.

```
setwd("/home/dlichtblau/Desktop/SRT411/assign0/TODOs_screenshots/5.0/")
#setwd("/home/")
d = 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))
d
```

```
##    a  g  x
## 1  1  2  3
## 2  2  4  6
## 3  4  8 12
## 4  8 16 24
## 5 16 32 48
## 6 32 64 96
```

```
write.table(d, file="tst1.txt", row.names=FALSE)
d2 = read.table(file="tst1.txt",header=TRUE)
g5 <- d2$g*5
g5
```

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

```
gNew = data.frame(a = c(1, 2, 4, 8, 16, 32), g = c(2, 4, 8, 16, 32, 64), gCol5 = g5, x = c(3, 6, 12, 24
gNew
```

```
##    a  g gCol5  x
## 1  1  2    10  3
## 2  2  4    20  6
## 3  4  8    40 12
## 4  8 16    80 24
## 5 16 32   160 48
## 6 32 64   320 96
```

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

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

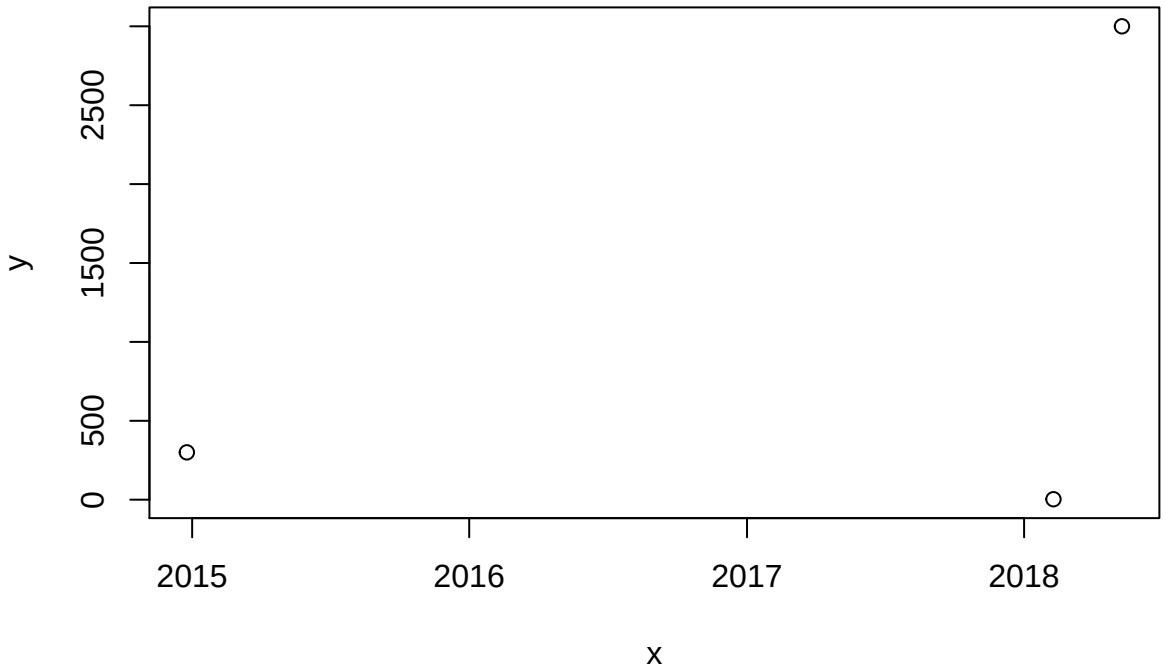
```
    mean(sqrt(c(rnorm(100))))
```

```
## Warning in sqrt(c(rnorm(100))): NaNs produced
```

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

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

```
date2 = strptime(c("20180208165300", "20141225000001", "20180510000001"), format="%Y%m%d%H%M%S")
x = date2
y = c(3, 300, 3000)
plot(x,y)
```



11.2: 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.

```
vect = 1:100
for (loopVect in vect)
{
  if (vect[loopVect] < 5 | vect[loopVect] > 90)
  {
    vect [loopVect] = vect[loopVect] * 10
  }
  else
  {
    vect[loopVect] = vect[loopVect] * 0.1
  }
}
vect
```

```
##   [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: 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 specification of the counter.

```
vect = function(start, end)
{
  x = seq(from = start, to = end)
  for (counter in x)
  {
    if (counter < 5 | counter > 90)
      (x[counter] = counter * 10)
    else (x[counter] = counter * 0.1)
  }
  x
}
vect(start = 2, end = 150)
```

```
##   [1]    2.0   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
## [101] 1010.0 1020.0 1030.0 1040.0 1050.0 1060.0 1070.0 1080.0 1090.0 1100.0
## [111] 1110.0 1120.0 1130.0 1140.0 1150.0 1160.0 1170.0 1180.0 1190.0 1200.0
## [121] 1210.0 1220.0 1230.0 1240.0 1250.0 1260.0 1270.0 1280.0 1290.0 1300.0
## [131] 1310.0 1320.0 1330.0 1340.0 1350.0 1360.0 1370.0 1380.0 1390.0 1400.0
## [141] 1410.0 1420.0 1430.0 1440.0 1450.0 1460.0 1470.0 1480.0 1490.0 1500.0
```

11.3*: Actually, people often use more for-loops than necessary. The ToDo above can be done more easily and quickly without a for-loop but with regular vector computations.

```
vect1 <- seq(from = 1, to = 5, by = 1) * 10
vect2 <- seq(from = 6, to = 89, by = 1) * 0.1
vect3 <- seq(from = 90, to = 100, by = 1) * 10
vect123 <- c(vect1, vect2, vect3)
vect123
```

```
##   [1]   10.0   20.0   30.0   40.0   50.0    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  900.0
##  [91]  910.0  920.0  930.0  940.0  950.0  960.0  970.0  980.0  990.0 1000.0
```