BDM600 - Lab 1 - Group 8

Ran Arino; Zubeka Dane Dang; Solmaz Heidar Nassab

2024-01-15

Credentials

All members answered the whole questions individually, the file shows the merged result.

3.1 Calculator

Compute the difference between 2014 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.

ToDo 1

```
(2021 - 2014)/(2014 - 1987) * 100
```

```
## [1] 25.92593
```

ToDo 2

```
university_start_year <- 2021
birth_year <- 1987
percentage_at_university <- (university_start_year - 2014) / (2014 - birth_year) * 100
percentage_at_university
```

```
## [1] 25.92593
```

3.4 Functions

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

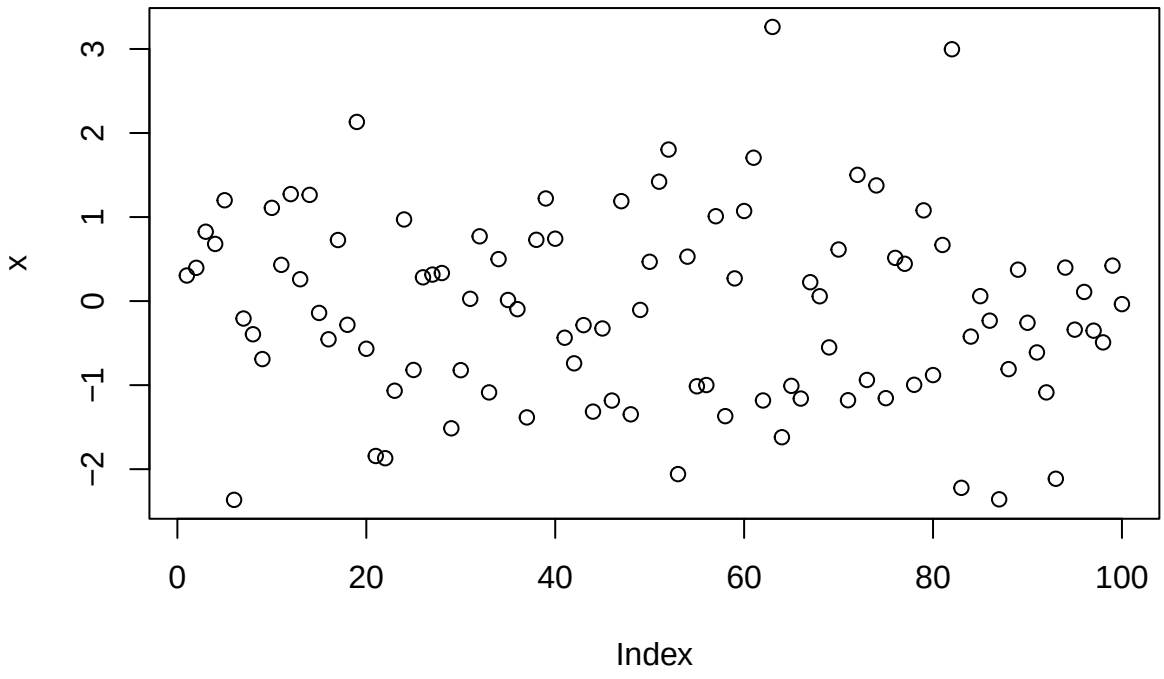
```
number <- c(4, 5, 8, 11)
sum(number)
```

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

3.5 Plots

Plot 100 normal random numbers.

```
x <- rnorm(100)
plot(x)
```



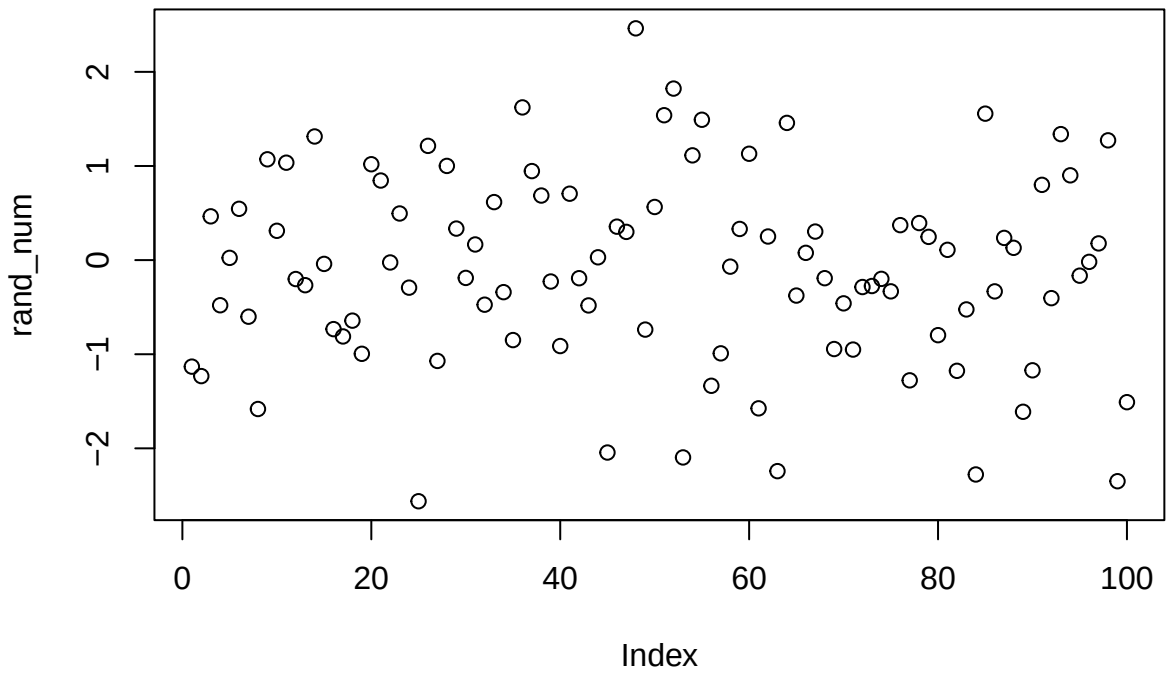
4 Help and documentation Find help for the sqrt function.

```
help("sqrt")
```

5 Scripts

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

```
# make sure the script is in the same directory.
path <- "firstscript.R"
source(path)
```



6.2 Matrices

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 dfferent ways scalars, vectors and matrices are denoted in the workspace window.

```
# Create vector P using the seq function
P <- seq(31, 60)
print(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 54 55
## [26] 56 57 58 59 60
```

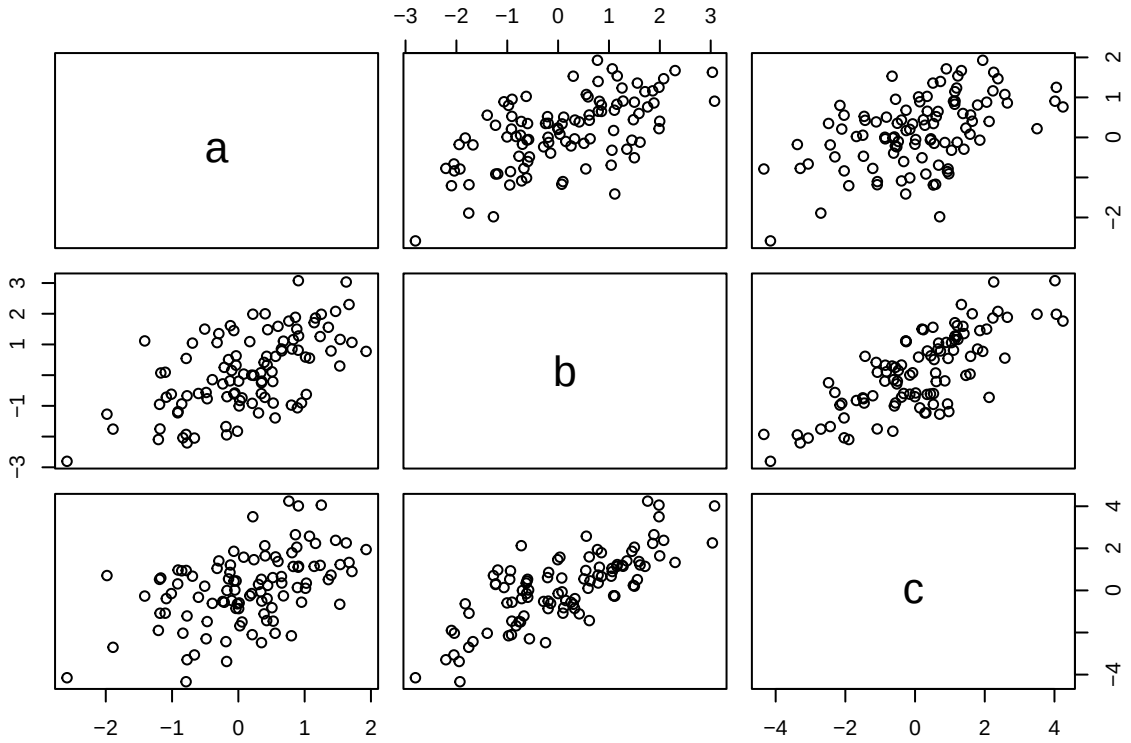
```
# Create matrix Q with 6 rows and 5 columns
Q <- matrix(P, nrow = 6, ncol = 5, byrow = TRUE)
Q
```

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

6.3 Data frames

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.

```
# make sure the script is in the same directory.
path <- "secondscript.R"
source(path)
```




```
## [1] 0.8785757
## [1] 1.259977
## [1] 1.684873
```

7 Graphics

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.

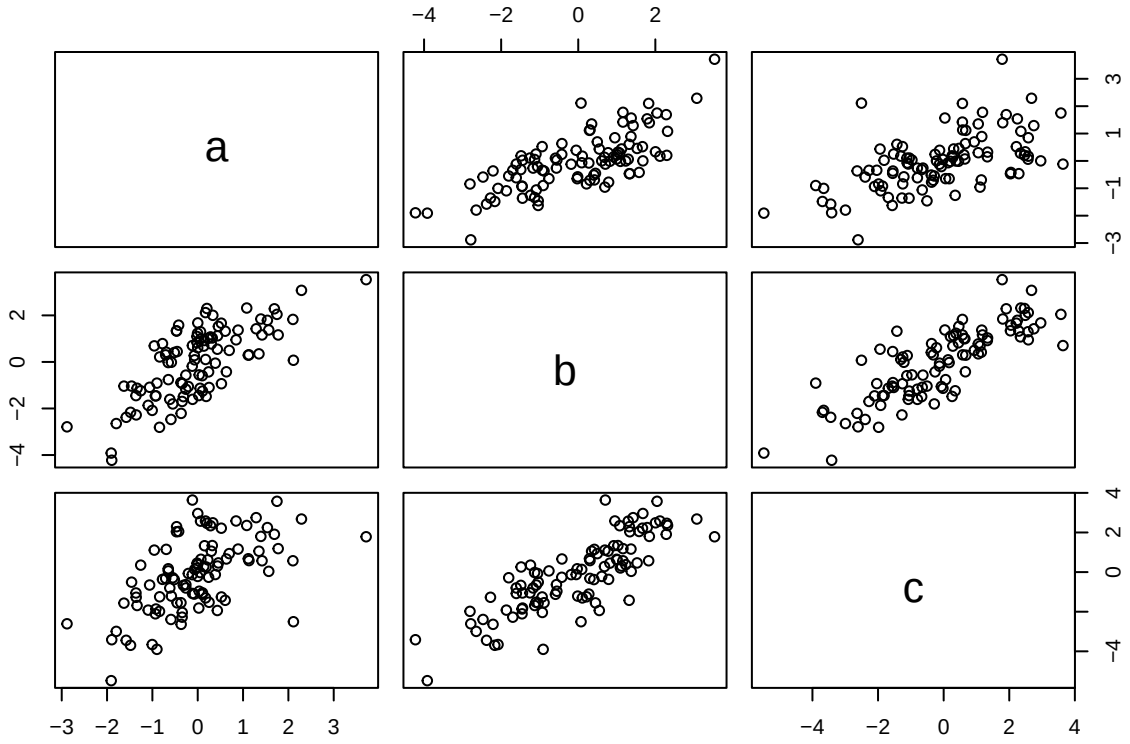
```
help("rgb") # RGB Color Specification
help("lwd") # Line Width
```

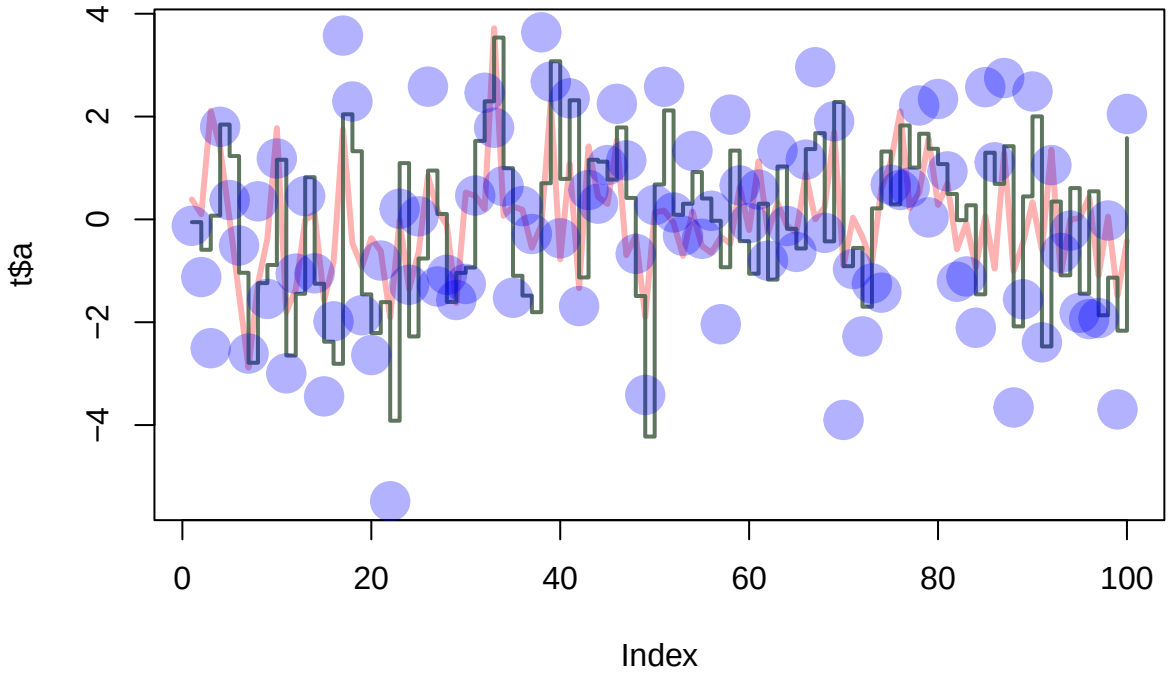
```
## No documentation for 'lwd' in specified packages and libraries:
## you could try '??lwd'
```

```r
help("pch") # Plotting point character
help("cex")# Size of the objects
```

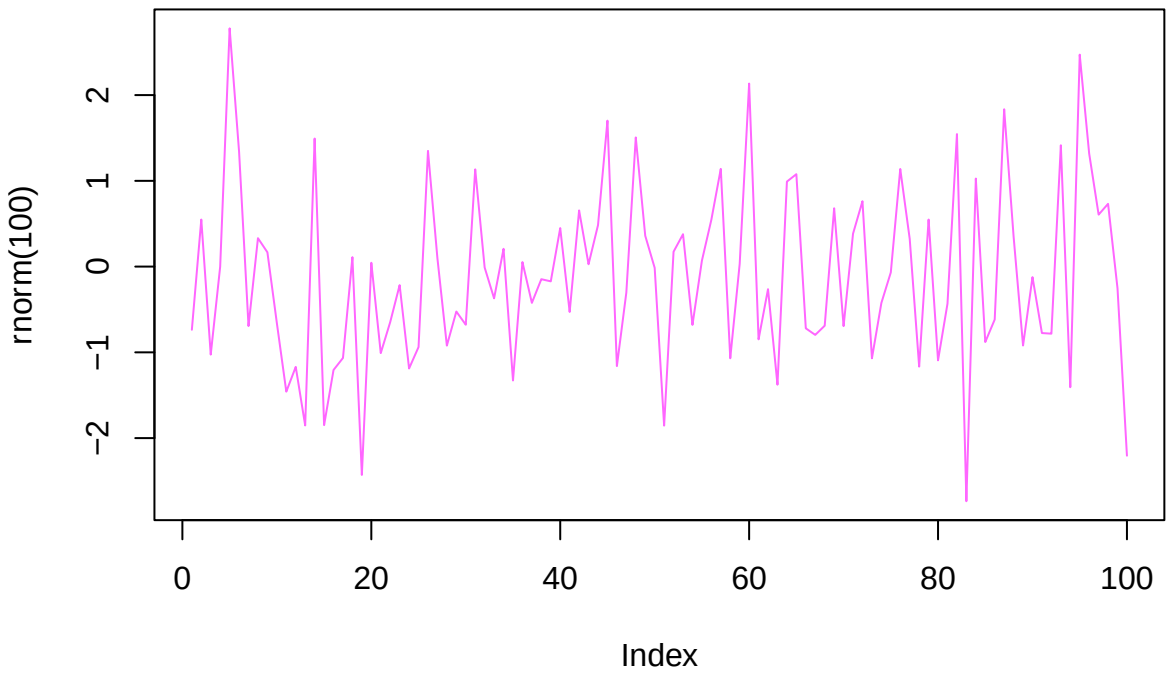
```
## No documentation for 'cex' in specified packages and libraries:
## you could try '??cex'
```

```r
# make sure the file is in the same directory.
path <- "thirdscript.R"
source(path)
```

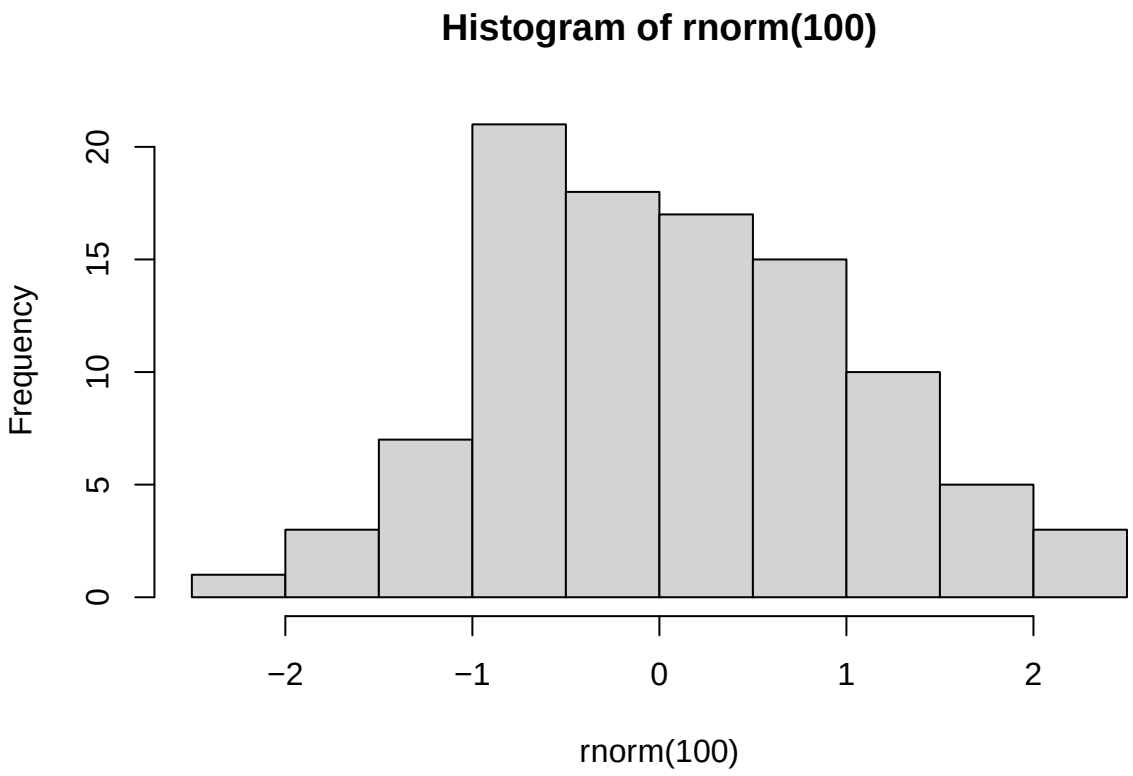





```
# A simple histogram plot of 100 random number
plot(rnorm(100), type="l", col=rgb(1,0,1,0.6))
```




```
hist(rnorm(100))
```



8 Reading and writing data files

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.

```
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)
  )
write.table(d, file = "tst1.txt", row.names = FALSE)

d2 <- data.frame(
  a = c(1, 2, 4, 8, 16, 32),
  g = 5*c(2, 4, 8, 16, 32, 64),
  x = c(3, 6, 12, 24, 48, 96)
  )
write.table(d2, file = "tst2.txt", row.names = FALSE)
```

9 Not available data

```r
# return NaN
rand_num <- rnorm(100)
mean_squart <- mean(sqrt(rand_num))
```

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

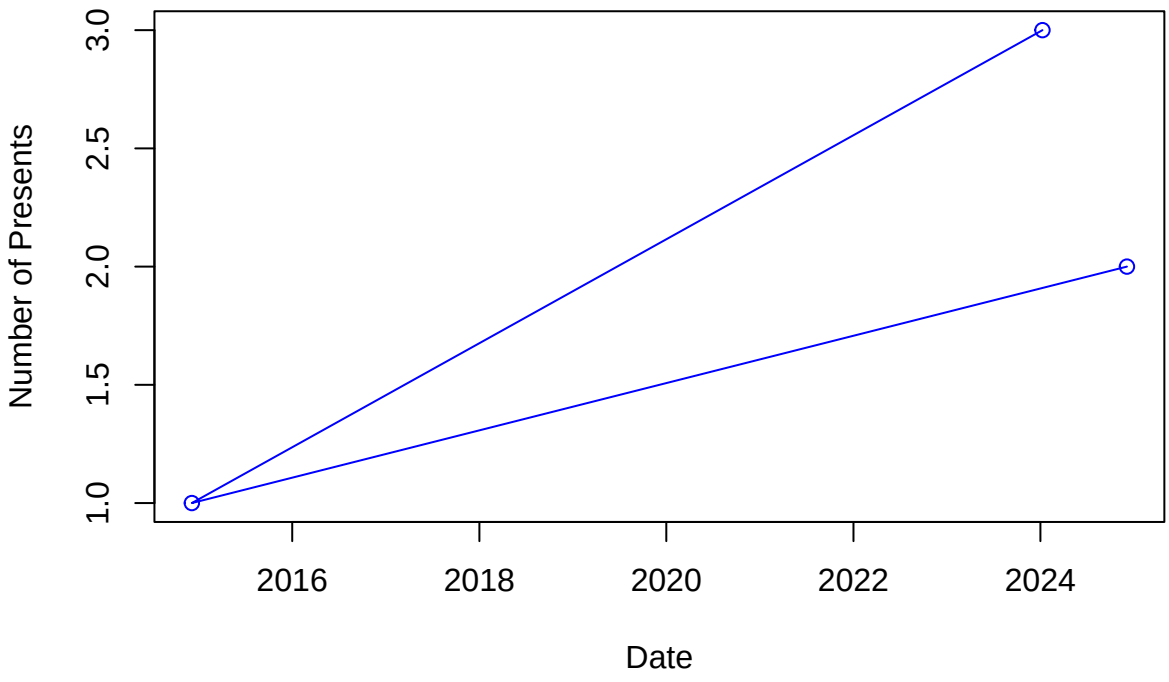
```r
print(mean_squart)
```

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

10.2 Dates

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 first.

```r
dates <- as.Date(c("2024-01-09", "2014-12-05", "2024-12-04"))
presents <- c(3, 1, 2)
plot(dates, presents, type = "o", col = "blue", xlab = "Date",
     ylab = "Number of Presents")
```



11.2 For-loop

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.

```
x <- 1:100

# Initialize an empty vector to store the results
y <- vector()

# Loop through the elements of x
for (i in x) {
  # Check the condition and multiply accordingly
  if (i < 5 || i > 90) {
    y[i] <- i * 10
  } else {
    y[i] <- i * 0.1
  }
}

# Print the result
print(y)
```

```
##   [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 functions

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.

```
func <- function(vec) {
  result <- numeric(length(vec))
  for (i in seq_along(vec)) {
    if (vec[i] < 5 || vec[i] > 90) {
      result[i] <- vec[i] * 10
    } else {
      result[i] <- vec[i] * 0.1
    }
  }
  return(result)
}
vec <- seq(1, 30)
func(vec)
```

```
##  [1] 10.0 20.0 30.0 40.0  0.5  0.6  0.7  0.8  0.9  1.0  1.1  1.2  1.3  1.4  1.5
## [16]  1.6  1.7  1.8  1.9  2.0  2.1  2.2  2.3  2.4  2.5  2.6  2.7  2.8  2.9  3.0
```