SRT411Assignment Joo Young Kim

Joo Young Kim

February 15, 2019

SRT 411 Assignment 0: Intro to R and GIthub

Joo Young Kim

The document, "A (very) short introduction to R", "Todo" sections will be accomplished. The 14 different "Todo" sections will be done with the R programming language. By writing the code of each Todo, it will be demonstrated. Also, there is a final "Todo" provided by the professor to be completed. Thus, the total of 15 "Todo" will be demonstrated with the R programming language. These "Todo" are going to be in Rmarkdown to knitr as pdf file to be completed and submitted.

To Do 1

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. The year I started the university: 2016 The year I was born: 1993

```r
((2016-2014)/(2014-1993))*100
```

```
## [1] 9.52381
```

To Do 2

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

The previous ToDO code is going to be used. However, this time, the variable will be created. The variable that will be created will be named, "variable1"

```r
variable1=((2016-2014)/(2014-1993))*100
variable1
```

```
## [1] 9.52381
```

The answer of this is, as same as the previous result. However, the variable1 is created. Whenever the output of the calculation done needs to be displayed, now, variable1 can be generated in R programming to display the output of the previous To Do.

To Do 3

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

A vector is created by creating variable "a" and within the variable "a", the vector of sum of 4,5,8 and 11 are created.

```r
a=c(4,5,8,11)
```

The sum of 4,5,8 and 11 has been created by combining them into a vector

```
sum(x=a)
```

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

Then by using the function sum, the output has been computed.

To Do 4

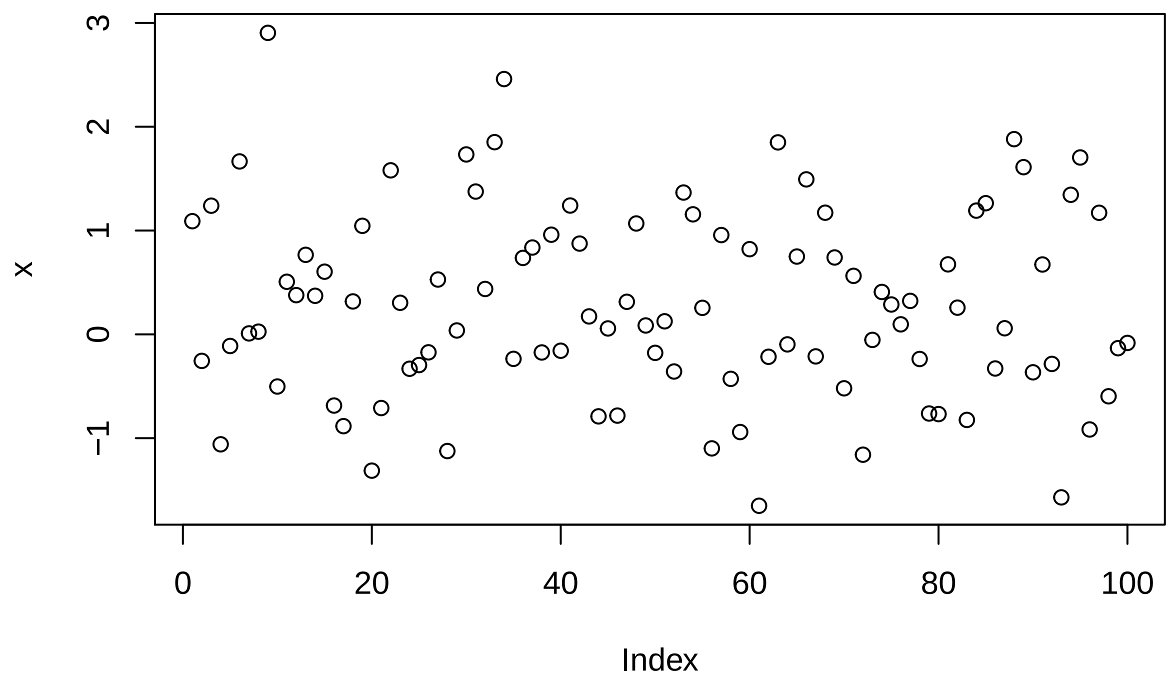
Plot 100 normal random numbers

First, by using the rnorm() function, 100 normal randowm numbers are generated.

```
x = rnorm(100)
```

Then, to plot the generated random numbers, the plot() function is used

```
plot(x)
```



To Do 5

Find help for the sqrt function.

TO find the help for the sqrt function, the help function is computed to find out the sqrt function.

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

To Do 6

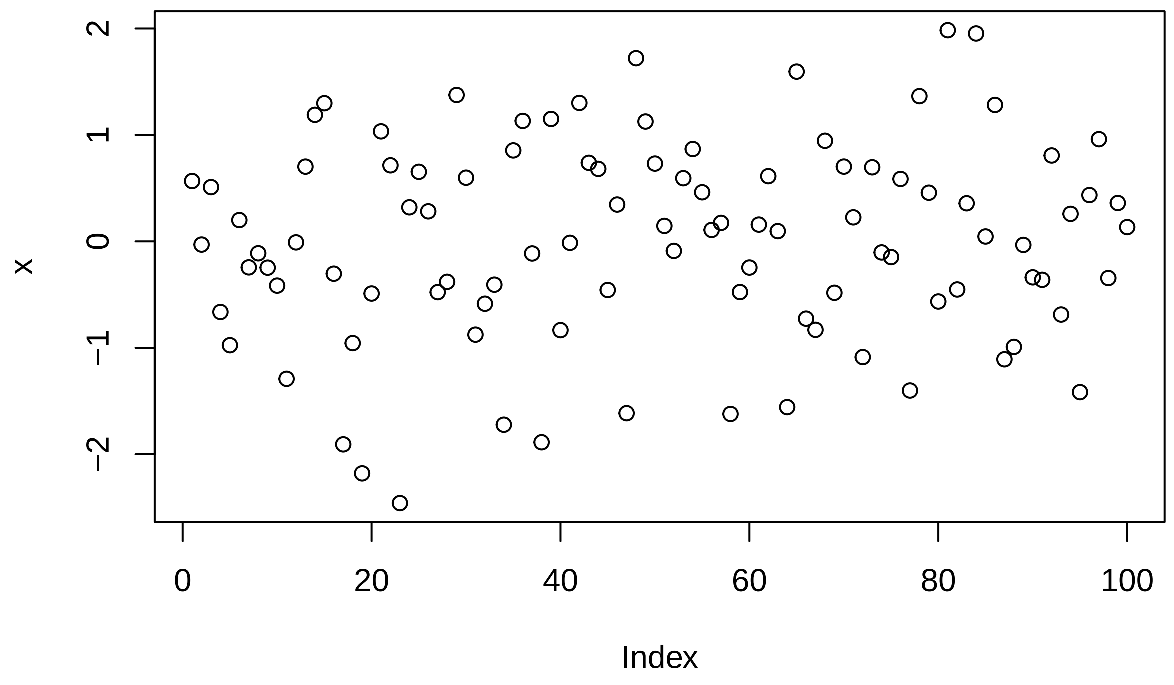
Make a file called firstscript.R containing R-code that generates 100 random numbers and plots them, and run this script several times.

The Script is called firstscript.R has been created with the R-code that generates 100 random bumbers and ploted them. The script, firstscript.R, has codes: x = rnorm(100) plot(x)

The script is saved in the working directory as working space to generate the script.

To call out the firstscript.R and generate its codes.

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



To Do 7

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. Loot at the different ways scalars, vectors and matrices are denoted in the workspace windows.

```
P= seq(from=31, to=60, by=3)
```

A vector named P is created that put the numbers 31 to 60 in a vector

```
Q=matrix(data=c(P), ncol=5, nrow=6)
```

Matrix named Q with 6 rows and 5 columns are created and also a vector named P is generated as matrix.

To Do 8

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=rnorm(100)
```

The vector x1 has been created

```
x2=rnorm(100)
```

The vector x2 has been created

```
x3=rnorm(100)
```

The vector x3 has been created

Vectors x1, x2, and x3 are created with the random normal vectors of length 100.

```
x1
```

```
##   [1]  0.88187767  0.95107146  0.36762236 -1.17215444  1.28710510
##   [6] -0.43781990  0.49542996 -0.53034797 -0.91829089 -2.79011790
##  [11]  0.44982973  1.22456564 -0.53085888 -0.36899474 -0.82452205
##  [16]  0.31389869 -0.44405548 -0.31515083 -0.27144260  2.30673468
##  [21]  0.43358970 -0.72276437  1.36829653 -0.10326344  0.30049630
##  [26] -0.25811022  0.74957962  1.38571395  1.58217452  0.60584224
##  [31]  0.67404087 -0.90544753 -0.98282372 -0.93496129 -0.13027406
##  [36]  1.14438543 -1.16090955 -0.55620589  0.67518081  0.21388519
##  [41] -1.44550150 -0.51836315  2.57229917 -0.43916281 -0.06626715
##  [46] -1.01079338  0.60158506  0.89709738 -0.13829723 -1.39551095
##  [51] -1.00325216 -0.65055370  0.24191954 -0.01042998 -0.79106884
##  [56]  0.81297253  0.31645421 -1.59088399 -0.54943864 -0.47267314
##  [61]  1.35224906 -0.50077168  1.02950272  1.44829207  1.23634159
##  [66]  1.60441173 -0.75473017  1.06325352 -0.29188880  0.92174192
##  [71]  0.31816957  0.72397724  0.48844297  0.96768434  1.23860302
##  [76]  1.53332257  0.04345122 -1.08942245 -0.11155640  0.31860997
##  [81] -0.62160656 -0.01113100 -1.09708415 -0.92773895  0.13659569
##  [86] -1.52459691 -0.77108425 -0.44108025 -0.09203614  1.92873207
##  [91] -0.23269132 -0.92680635  1.08724476 -0.54954088  1.39259785
##  [96] -0.95481353 -0.35752934  0.76227097  0.10051142 -1.09741848
```

```
x2
```

```
##   [1] -0.551592971  0.121996911  0.629292505  0.029288129 -0.583513014
##   [6] -1.820797193  0.570798553  1.404194237  0.945642167  0.396543835
##  [11] -0.097153947 -0.242990104  0.813999153 -0.101005359  0.315915186
##  [16] -0.505207786 -0.990972769  1.071761350 -0.004921276 -0.326798970
##  [21] -0.400931664 -0.385990863  1.578873295  0.143362584  0.355027967
##  [26] -0.914723002  1.261618648  2.078055880  1.624514217 -0.981941833
##  [31]  0.813009289  1.175278236  0.690313898 -0.077763021  0.076159613
##  [36]  0.143410204 -0.846815758 -0.771779048  1.480277320 -0.716223317
##  [41] -0.959831280  0.692327494  0.490176061  0.940714292 -0.302648750
##  [46]  1.884242288  0.613247763  2.017020669  0.164925810 -1.140202920
##  [51] -0.088145170 -2.216519094 -0.116639548  1.008617256  1.770068644
##  [56]  0.231720130 -0.227148700 -1.581399552  0.253240129 -0.407635279
##  [61] -0.948715959 -0.402240696 -1.216499891 -0.187869424  0.282863690
## [66]  0.192409252 -1.067854360  0.275864799 -0.311386961  0.173829034
## [71] -1.604397580 -0.916715322  0.665993762  2.411271222 -0.667742883
## [76]  0.753138831 -0.072410372  1.484077723 -0.684292494  0.790586603
## [81]  2.763280688  0.513844398  0.526188628  0.762049826  1.491353882
## [86] -0.550041600 -0.965088845  1.695793046 -1.612853890 -0.273388240
## [91] -0.342107539 -0.604319379  0.807024842  0.530517971  0.912993755
## [96]  0.121780115 -2.408781941  0.124507544  0.090641139 -0.160363961
```

```
x3
```

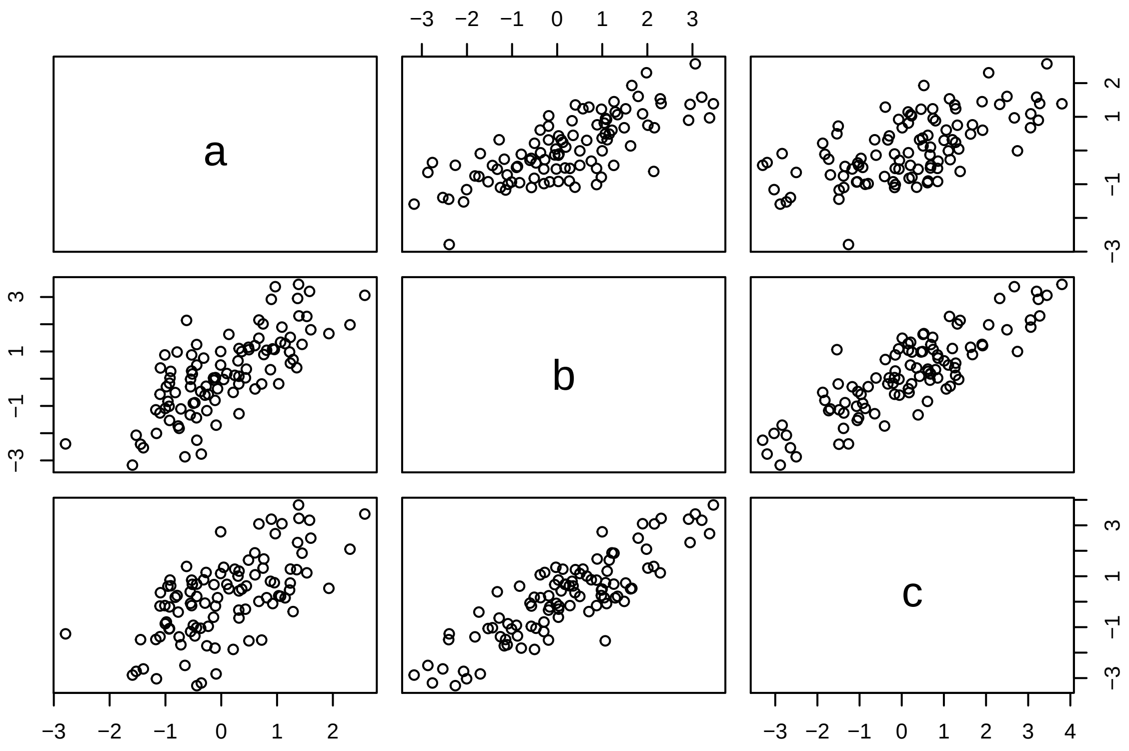
```
##   [1]  0.471269493 -0.327169606 -0.502313475 -0.341232952 -1.092310003
##   [6] -1.039533972 -2.603271931 -0.025388573  0.824100840  1.130878272
##  [11]  0.266860284 -0.523330908 -0.436485171 -0.572855334  0.683343411
##  [16] -0.138250939  0.412280196  0.104596410  1.425411987  0.082738430
##  [21] -0.326902787 -0.582986750 -0.623167335 -0.210273298  0.349804937
##  [26] -0.559285543 -0.691461552  0.336895932 -0.007442515  1.432324921
##  [31] -1.474757741  0.356009656 -0.503402596 -0.058329294  0.724966653
##  [36] -1.136813742 -1.019671941  1.716904380  0.899687670 -1.371858194
##  [41]  0.915578260  0.508147770  0.381165620 -0.295362329  0.527514030
##  [46] -1.025688160  0.703318732  0.326925229 -0.635965603 -0.104092611
##  [51]  0.226756022  0.364448252  1.152500164  1.747476208 -0.735828314
##  [56] -0.887230775  0.334126527  0.291020697 -0.877565364 -0.464101338
##  [61]  0.854908935 -0.022618138  0.420470315  0.645975307 -0.784735269
##  [66]  0.699844229  0.441669913 -1.126844628  0.548153866 -1.168976851
##  [71]  0.645259547 -1.313277256  0.478549768 -0.707309022  0.710046862
##  [76] -1.153239645  1.380841469 -0.040640165 -1.026400663  0.092575690
##  [81] -0.755919553  0.604118972  0.401426411 -0.038243168 -1.121657745
##  [86] -0.661036118  1.328864968 -0.564079448 -1.131416870 -1.130015530
##  [91] -0.391005248  0.477195422  1.167770660 -0.045920476  0.967312475
##  [96]  1.443623647 -0.425802043  0.791396941  0.490914394 -0.116669717
```

The above codes generated each of these vectors output

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

The above code indicates the data frame called t with threee columns containing respectively x1, x1+x2, and x1+x2+x3.

```
plot(t)
```



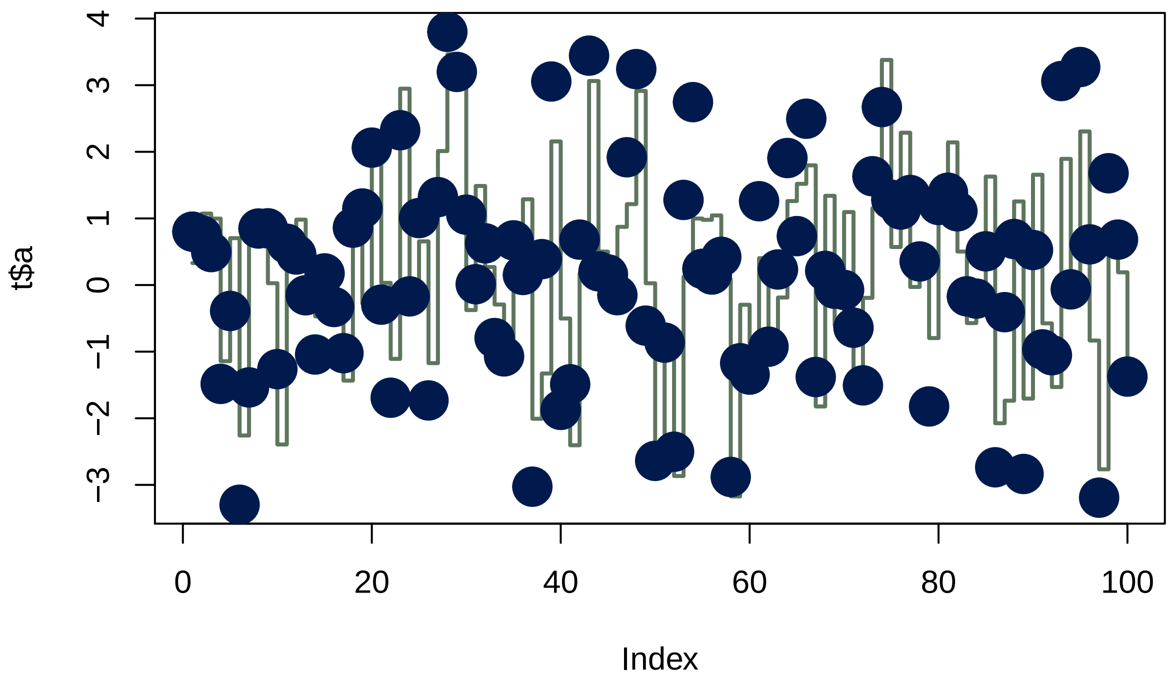
The data frame called t is plotted. It has three columns called a, b, and c that contrains respectively x1, x1+x2, x1+x2+x3.

To Do 9

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.

These lines are added with the data frame called t included in the previous To Do.

```
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))
```



The “rgb”" indicates would be the colour of the lines, plots, and points. It is used to color specification such as red, green, blue, alpha.

The “lwd” can be used to change line width

The “pch” indicates to plot with closed circles. It stands for “plot character”

The “cex” is number indicating the amount by which plotting text and symbols should be scaled relative to the default. 1=default, 1.5 is 50% larger, 0.5 is 50% smaller,etc.

To Do 10

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

```r
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 = read.table(file="tst1.txt", header=TRUE)
d2
```

```
##    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
```

```
d2$g <- d2$g *5
write.table(d2, file="tst2.txt", row.names=FALSE)
d3 = read.table(file="tst2.txt", header=TRUE)
d3
```

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

The example in FIgure 4 is used to store it in the working directory as file called tst1.txt. Then, to read it and to multiply the column called g by 5 and to store it as tst2.txt has been created.

To Do 11

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

```
TheSquareRoot = rnorm(100)
```

TheSquareRoot, variable, is created to generate a vector of 100 random numbers for next step, which is to copute the mean of the square root of a vector of 100 random numbers

```
mean(sqrt(TheSquareRoot))
```

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

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

By running the mean of the square root of a vector of 100 random numbers, there was warning messages that NaNs has been produced.

Before generating the mean, only the sqrt(TheSquareRoot) has generated to see why.

```
sqrt(TheSquareRoot)
```

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

```
##   [1]         NaN 1.296483591         NaN         NaN         NaN
##   [6]         NaN         NaN 1.205180330 0.482533305         NaN
##  [11]         NaN 1.226155107         NaN         NaN 1.285128184
##  [16] 0.625882954         NaN         NaN 1.262137940         NaN
##  [21]         NaN 1.353499324 0.787129132         NaN 0.449175782
##  [26] 0.972132912 1.083918959         NaN 1.117325283 0.490489602
##  [31]         NaN 0.411191761 0.619243525 1.023568782         NaN
##  [36]         NaN 0.115287286         NaN         NaN         NaN
##  [41]         NaN         NaN 0.978677833 0.354827828 1.000607657
##  [46]         NaN 0.570429111         NaN         NaN 0.257524814
##  [51]         NaN         NaN         NaN         NaN         NaN
##  [56]         NaN         NaN         NaN 0.006904966 0.802962530
##  [61] 1.211446725 0.818333638 0.494640449 1.047901866 0.944032350
##  [66] 0.899881833 0.636061969 0.843181638 0.894025875 0.738667346
##  [71]         NaN         NaN         NaN 0.833343113 1.066157661
##  [76] 0.581413171         NaN         NaN 0.538629233         NaN
##  [81] 0.678234990 0.398064582         NaN 0.907405288 0.744060029
##  [86] 0.303738039         NaN 1.452631947 0.905236510 0.900642190
##  [91]         NaN 1.406715656         NaN 1.175584145         NaN
##  [96]         NaN 1.224097791 0.910245469 0.394690176         NaN
```

It shows that there are square root values generated. However, there are NaNs which are not defined or calculated, this is why generating the mean of this could not be proceed.

To Do 12

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.

```
datevec=strptime( c("20140215000000", "20151025000000","20161025000000","20171025000000","2018102500000
datevec
```

```
## [1] "2014-02-15 EST" "2015-10-25 EDT" "2016-10-25 EDT" "2017-10-25 EDT"
## [5] "2018-10-25 EDT" "2019-10-25 EDT"
```

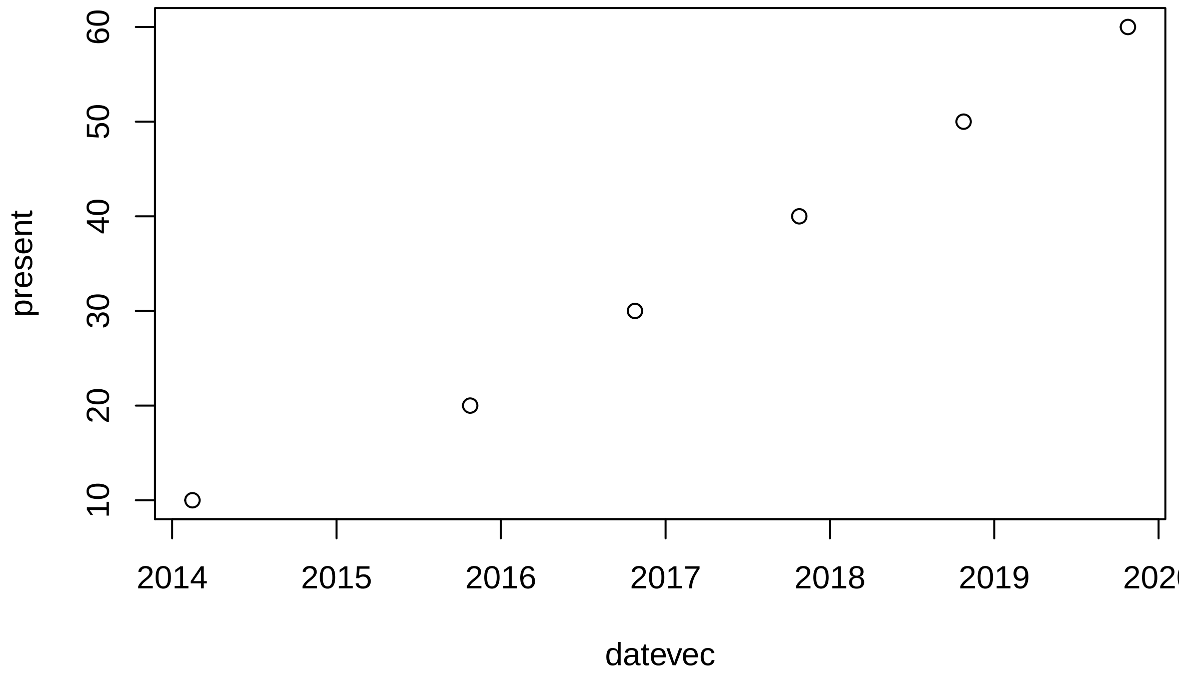
The datevec has been created

```
present=c(10,20,30,40,50,60)
present
```

```
## [1] 10 20 30 40 50 60
```

The vector of presents expected is created with present variable.

```
plot(datevec,present)
```



From creating two vectors, now the two vectors are to be plotted with x-axis of datevec and y-axis of present to represent how many presents did I got from 2014 February 15 to 2019 October 25.

To Do 13

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

```
Todo13 = seq(from=1, to=100)
variable_todo13= c()
for(i in 2:100)
  {variable_todo13[i] = Todo13[i] * 5
}
```

The vector from 1 to 100 is created with the variable, "Todo13". Then variable_todo13, variable has been created to maek a for-loop which runs through the whole vector, multiply the elements which are samller than 5 and larger than 90 with 10 and the other elements with 0.1

```
variable_todo13
```

```
##   [1]  NA  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80  85
##  [18]  90  95 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170
##  [35] 175 180 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255
##  [52] 260 265 270 275 280 285 290 295 300 305 310 315 320 325 330 335 340
##  [69] 345 350 355 360 365 370 375 380 385 390 395 400 405 410 415 420 425
##  [86] 430 435 440 445 450 455 460 465 470 475 480 485 490 495 500
```

The output of this "ToDo" is above output ##To Do 14 Write a function for the preivous ToDo, so that you can feed it any vector you like (as argumnet). 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.

```
function1 <- function(){
  Todo13 = seq(from=1,to=100)
  print(Todo13)
}
```

The fucntion could be created with only simple computation. The function has not been created according to the To Do instruction.

The Final ToDo

The final Todo in the document has a footnote. Write code that will prove that footnote true.

A footnote [^1] [^1]:footnote has been created to prove that footnote is True