

Lab 1

R, RStudio and Posit Cloud

RStudio is a framework that contains an R console with a similar look-and-feel than Matlab. It is easier to work in R using RStudio.

Now a big company has merged R and Python in RStudio and called it Posit. If you have not installed your RStudio in your computer, you can use Posit Cloud to work on it.

But before everything, run the following command in your console:

```
tinytex::install_tinytex()
```

R Lab

Now keep practicing. Complete the R chunks below to answer the questions

Exercise 1

Code below initialises variable x . Add your code below to calculate

$$w = \sqrt{x} + 5 + \sin(a)/\pi, \text{ for } x = 6 \text{ and } a = 40$$

Hint: Note that number π is a constant in R with name **pi**. Also R has function **sin** that calculates the sine of the number inside brackets, and function **sqrt** that calculates the square root of a number.

```
x <- 6
```

Answer:

Exercise 2

```
#Create a numeric vector called 'a' with values 1, 2, 3, 4, 5.  
 #(write you code below here)  
# using [], move the position of values in 'a', so the final result is 1, 5, 2, 4, 3  
 #(write you code below here)
```

Question: What does the symbol '#' does in R language.

Your answer:

Exercise 3

```
v1 <- c(4,6,8,24)  
2*v1
```

```
## [1] 8 12 16 48
```

```
# add your vector 'a' and 'v1' (what happens?) and why  
# (your line of code)
```

```
# calculate 5 times 'a'
# (your line of code)

# calculate the cosine of 'a'
# (your line of code)
```

Exercise 4

Random number generation

We can generate a sample of (pseudo) random numbers from different probability distributions.

```
# It generates 10 values from a standard normal
rnorm(10)

## [1] 0.07123789 0.03924031 1.21875342 0.56720237 0.30847718 -0.18485744
## [7] -0.99556540 0.22906521 -1.13573477 0.98808147

# type ?rnorm in the console and answer:
# 1. What is the mean value of the values generated by the command above

# Create a sequence of 20 random values with mean 2 and variance 4
# (your code here)

# It generates 5 values from a t-student distribution with 10 degrees of freedom
rt(5,df=10)

## [1] 0.3782198 -1.6647186 1.3763484 0.7300167 0.6583129

# Google how to generate values from a chi-squared distribution in R
# Generate 10 values of a chi-squared distribution with 10 degrees of freedom
# (your code here)
```

Exercise 5

Indexing

R vector or matrix indexes start at 1. This is different from Python, C, C++ or Java where vector and matrix indexes start at 0. One can choose certain elements of a vector by calling them with their index. Similarly, one can avoid certain element of a vector by dismissing it by its index (with a minus sign at front).

```
x <- c(0,-3,4,-1,45,90,-5)

x[c(4,6)]

## [1] -1 90

x[1:3]

## [1] 0 -3 4

y <- c(1,4)
x[y]

## [1] 0 -1

x[-1]

## [1] -3 4 -1 45 90 -5
```

```
x[-c(4,6)]
```

```
## [1] 0 -3 4 45 -5
```

```
x[-(1:3)]
```

```
## [1] -1 45 90 -5
```

```
# Understand the code above
```

Question: What does x[10] return? why?

Exercise 6

Boolean

In addition to arithmetic operations like '+', '-', '*', computers can do boolean operations, logical operations.

```
x <- c(0,-3,4,-1,45,90,-5)
```

```
x > 0
```

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

```
# Understand what the previous line does
```

```
# Which values of x are truly smaller than -0.5?
```

```
# (write your code here)
```

```
# We can use boolean operators (& = AND, |= OR)
```

```
x <= -2 | x > 5
```

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

```
x > 40 & x < 100
```

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

```
# Understand the previous lines
```

```
# Which values of x are truly greater than 0 but lower or equal to 45?
```

```
# (write your code here)
```

```
x[x>0]
```

```
## [1] 4 45 90
```

```
x[x <= -2 | x > 5]
```

```
## [1] -3 45 90 -5
```

```
x[x > 40 & x < 100]
```

```
## [1] 45 90
```

```
# Understand the previous lines
```

```
# Print values of x that greater than 0 but lower or equal to 45
```

```
# (write your code here)
```

Exercise 7

Matrices

Matrices are common formats for our datasets. Many functions use them as input to do calculations. Understand the code below and fill the space “Your code” to answer questions.

```
# I create a matrix of 2x5
m <- matrix(c(45,23,66,77,33,44,56,12,78,23),nrow =2,ncol=5,byrow=T)
m
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]  45  23  66  77  33
## [2,]  44  56  12  78  23
```

```
# Obtaining elements from a matrix
m[2,3]
```

```
## [1] 12
```

```
m[-2,1]
```

```
## [1] 45
```

```
m[1,-c(3,5)]
```

```
## [1] 45 23 77
```

```
m[1,]
```

```
## [1] 45 23 66 77 33
```

```
m[,4]
```

```
## [1] 77 78
```

```
# Get all columns but number 2
# (Your code)
```

Exercise 8

Data Frames

Data frame is an R object similar to a matrix but that it can have columns that are numbers and or characters (letters and symbols). The only restriction is that every column should have the same number of rows. See below how we access columns and values in data frames.

```
# Create a data frame
student_data <- data.frame(
  Name = c("Jon", "Haizea", "Ibon", "Jaione"),
  Age = c(20, 21, 19, 22),
  Grade = c("A", "B", "A", "C"),
  stringsAsFactors = FALSE
)
```

```
# Dimension of the data.frame (how many rows and columns)
dim(student_data)
```

```
## [1] 4 3
```

```
# Display data.frame
print(student_data)
```

```
##      Name Age Grade
## 1    Jon  20     A
## 2 Haizea  21     B
```

```
## 3   Ibon  19    A
## 4 Jaione 22    C

#Display a particular value in the data.frame
student_data[3, 2]

## [1] 19

#Display a column
student_data[, 3]

## [1] "A" "B" "A" "C"

# Variable names
names(student_data)

## [1] "Name" "Age"  "Grade"

# Accessing specific elements in the data frame
# Print the name of the second student
student_data$Name[2]

## [1] "Haizea"

# Print the grade of the fourth student
print(student_data$Grade[4])

## [1] "C"

# Add a new column for Gender
student_data$Gender <- c("M", "F", "F", "M")
print(student_data)

##      Name Age Grade Gender
## 1   Jon  20    A      M
## 2 Haizea 21    B      F
## 3   Ibon 19    A      F
## 4 Jaione 22    C      M

# Filter the data frame to display students with age less than 21
young_students <- student_data[student_data$Age < 21, ]
print(young_students)

##      Name Age Grade Gender
## 1   Jon  20    A      M
## 3 Ibon  19    A      F

# Calculate the mean age of students
mean_age <- mean(student_data$Age)
print(mean_age)

## [1] 20.5
```

Exercise 9

read.table

We can open .csv and .txt files using **read.table**. First thing we have to know is the format of the file, that is, whether it has a header, what is the symbol that separates columns and what is the decimal symbol.

1. Open File1.csv using Notepad or Wordpad and answer.

- Name and extension of the file?
- Does it have a header?
- What is the delimiter of columns?
- What is the decimal symbol?
- Does it have missing values?

```
# Knowing the answer to those question, the file will be open this way
students1 <- read.table("File1.csv", header = TRUE, sep = ";", dec = ",")
print(students1)
```

```
##      Name Age Grade Height_cm
## 1    Jon  20     A    175.5
## 2 Haizea  21     B    163.2
## 3   Ibon  19     A    168.9
## 4 Jaione  22     C    182.1
```

2. Open File2.csv using Notepad and answer the same questions.

```
# Knowing the answer to those question, the file will be open this way
students2 <- read.table("File2.csv", header = TRUE, sep = ",", dec = ".", na.strings = "")
print(students2)
```

```
##      Name Age Grade Height_cm
## 1    Jon  20     A    175.5
## 2 Haizea  21 <NA>    163.2
## 3   Ibon  NA     A    168.9
## 4  <NA>  22     C         NA
```

Question:

- What does NA mean in your dataset?

3. Open File3.csv (with quoted strings)

```
students3 <- read.table("File3.csv", header = TRUE, sep = ",", dec = ".", quote = "\"")
```

```
## Warning in read.table("File3.csv", header = TRUE, sep = ",", dec = ".", :
## incomplete final line found by readTableHeader on 'File3.csv'
```

```
print(students3)
```

```
##      Name Age Grade Height_cm
## 1    Jon  20     A    175.5
## 2 Haizea  21     B    163.2
## 3   Ibon  19     A    168.9
## 4 Jaione  22     C    182.1
```

4. Open File4 (without header)

```
students4 <- read.table("File4", header = FALSE, sep = ",", dec = ".")
print(students4)
```

```
##      V1 V2 V3   V4
## 1    Jon 20  A 175.5
## 2 Haizea 21   163.2
## 3   Ibon NA  A 168.9
## 4 Jaione 22  C 182.1
```

5. Open File5 with what you know now in the R chunk below

Exercise 9

- Create a Lab1.pdf, Lab1.html and Lab1.docx

Homework: Posit Cloud

- Log-in Posit Cloud with your Deusto account (you must have joint using the link in ALUD first)
- Do the exercises in Primers/The basics/Programming basics. You will need to use your headphones to hear the videos.