R for Data Science (V): Programming

Alex Sanchez, Miriam Mota, Ricardo Gonzalo and Mireia Ferrer

Statistics and Bioinformatics Unit. Vall d'Hebron Institut de Recerca

Outline: Introduction to programming

- Introduction
- Control structures
- The `apply` family
- User defined functions

*Based on this presentation: **Programacion en R** by *Antonio Miñarro* Universitat de Barcelona.

Introduction

- We have introduced R as a *a language (a tool), to manage and analyze data*.
- It is also a *programming language*
  - It is simple and versatile
  - The user can create new functions that adapt to their needs
  - It is widely used (2nd most widely used in Data Science)
  - Users provide the community with a high variety of solutions
  - As a programming language it is not, however, very efficient

Example 1: Why may we need programming

- Assume we have the diabetes dataset and want to make a summary of every variable it contains.

```r
library(readxl)
diabetes <- read_excel("datasets/diabetes.xls")
summary(diabetes)
```

- There are categorical variables but the system cannot recognnize them.

```
##     numpacie          mort              tempsviu          edat
##  Min.   :  1.00   Length:149         Min.   : 0.00   Min.   :31.00
##  1st Qu.: 38.00   Class :character   1st Qu.: 7.30   1st Qu.:43.00
##  Median : 75.00   Mode  :character   Median :11.60   Median :50.00
##  Mean   : 75.01                      Mean   :10.52   Mean   :52.17
##  3rd Qu.:112.00                      3rd Qu.:13.90   3rd Qu.:60.00
##  Max.   :149.00                      Max.   :16.90   Max.   :86.00
##       bmi           edatdiag        tabac                sbp
##  Min.   :18.20   Min.   :26.00   Length:149         Min.   : 98.0
##  1st Qu.:26.60   1st Qu.:38.00   Class :character   1st Qu.:124.0
##  Median :31.20   Median :45.00   Mode  :character   Median :138.0
##  Mean   :31.78   Mean   :45.99                      Mean   :139.1
##  3rd Qu.:35.20   3rd Qu.:53.00                      3rd Qu.:152.0
##  Max.   :59.70   Max.   :81.00                      Max.   :222.0
##       dbp             ecg                chd
##  Min.   : 58.00   Length:149         Length:149
##  1st Qu.: 74.00   Class :character   Class :character
##  Median : 80.00   Mode  :character   Mode  :character
##  Mean   : 90.04
##  3rd Qu.: 88.00
##  Max.   :862.00
```

- A simple solution: Convert text variables into factors.

```r
library(forcats)
diabetes$mort <- as_factor(diabetes$mort)
diabetes$tabac <- as_factor(diabetes$mort)
diabetes$ecg <- as_factor(diabetes$ecg)
diabetes$chd <- as_factor(diabetes$chd)
```

```r
summary(diabetes)
```

```
##     numpacie           mort        tempsviu          edat
##  Min.   :  1.00   Vivo  :124   Min.   : 0.00   Min.   :31.00
##  1st Qu.: 38.00   Muerto: 25   1st Qu.: 7.30   1st Qu.:43.00
##  Median : 75.00                Median :11.60   Median :50.00
##  Mean   : 75.01                Mean   :10.52   Mean   :52.17
##  3rd Qu.:112.00                3rd Qu.:13.90   3rd Qu.:60.00
##  Max.   :149.00                Max.   :16.90   Max.   :86.00
##       bmi           edatdiag        tabac          sbp
##  Min.   :18.20   Min.   :26.00   Vivo  :124   Min.   : 98.0
##  1st Qu.:26.60   1st Qu.:38.00   Muerto: 25   1st Qu.:124.0
##  Median :31.20   Median :45.00                Median :138.0
##  Mean   :31.78   Mean   :45.99                Mean   :139.1
##  3rd Qu.:35.20   3rd Qu.:53.00                3rd Qu.:152.0
##  Max.   :59.70   Max.   :81.00                Max.   :222.0
##       dbp               ecg      chd
##  Min.   : 58.00   Normal  :111   No:99
##  1st Qu.: 74.00   Frontera: 27   Si:50
##  Median : 80.00   Anormal : 11
##  Mean   : 90.04
```

- But how shoulde we proceed if there were dozens or hundreds of variables that need to be changed?
- What if, besides, these variables had different names at every new file?
- The solution consists of providing some way to indicate that "any" character variable is transformed into a factor.
- This will be an example of a "program",

Changing the flow of execution

Scripts are executed "lineally"

- R, as most ordinary programming languages, is executed lineally, that is from the first to last line.
- Sometimes this needs to be changed.
  - Taking alternative flows according to certain conditions
  - Repeating some instructions while certain condition holds, or a fixed number of times,. . .
- This can be acomplished using *Flow control structures*

Loop controlled by a counter: `for` instruction

- Loops are used in programming to repeat a specific block of code made by one or more instructions.
- Syntax of for loops:

```
for (val in sequence)
{
statement
}
```

- Here, `sequence` is a vector and `val` takes on each of its value during the loop. In each iteration, `statement` is evaluated.

Example of `for` loop

- A `for` loop can be used to change the selected columns in the diabetes dataset.

```
    diabetes <- data.frame(read_excel("datasets/diabetes.xls"))
are_char <- c(2,7,10,11)
    for (i in are_char) {
      diabetes[,i]<-as_factor(diabetes[,i])
    cat(colnames(diabetes)[i], class(diabetes[,i]), "\n")
    }
```

```
## mort factor
## tabac factor
## ecg factor
## chd factor
```

```r
summary(diabetes)
```

```
##     numpacie          mort        tempsviu          edat      
##  Min.   :  1.00   Vivo  :124   Min.   : 0.00   Min.   :31.00  
##  1st Qu.: 38.00   Muerto: 25   1st Qu.: 7.30   1st Qu.:43.00  
##  Median : 75.00                Median :11.60   Median :50.00  
##  Mean   : 75.01                Mean   :10.52   Mean   :52.17  
##  3rd Qu.:112.00                3rd Qu.:13.90   3rd Qu.:60.00  
##  Max.   :149.00                Max.   :16.90   Max.   :86.00  
##       bmi           edatdiag            tabac         sbp       
##  Min.   :18.20   Min.   :26.00   No fumador:57   Min.   : 98.0  
##  1st Qu.:26.60   1st Qu.:38.00   Fumador   :51   1st Qu.:124.0  
##  Median :31.20   Median :45.00   Ex fumador:41   Median :138.0  
##  Mean   :31.78   Mean   :45.99                   Mean   :139.1  
##  3rd Qu.:35.20   3rd Qu.:53.00                   3rd Qu.:152.0  
##  Max.   :59.70   Max.   :81.00                   Max.   :222.0  
##       dbp               ecg      chd   
##  Min.   : 58.00   Normal  :111   No:99  
##  1st Qu.: 74.00   Frontera: 27   Si:50  
##  Median : 80.00   Anormal : 11          
##  Mean   : 90.04
```

Exercise

- Create a `for` loop that reads all .csv filenames in your datasets directory (or the directory you decide) and prints the name of the file and the column names in the screen.

Conditional statements: `if` / `if - else`.

- Conditional statements allow different coding blocks to be executed depending on whether a certain condition is TRUE or FALSE.

- syntax of `if` statement is:

```
if (test_expression) {
    statement
  }
```

- If the `test_expression` is TRUE, the `statement` gets executed. But if it's FALSE, nothing happens.
- Here, `test_expression` can be a logical or numeric vector, but only the first element is taken into consideration.
- In the case of numeric vector, zero is taken as FALSE, rest as TRUE.

Conditional statements: if - else.

- syntax of if-else statement is:

```
if (test_expression) {
    statement_1
}else{
    statement_2
  }
```

- If the test_expression is TRUE, then statement_1 gets executed.
- If it's FALSE then statement_2 gets executed.