Introduction to Programming

Lecture 9-10: Introduction to R (cont'd)

Clément Mazet-Sonilhac
clement.mazet@sciencespo.fr

Sciences Po Paris

More functions

Working with several files (cont'd)

- Write a function called `analyze()` that :
  1. takes a `filename` as an argument
  2. displays the three graphs produced in the previous lesson (average, min and max inflammation over time).
- Hint : `analyze("../data/inflammation-01.csv")` should produce the three graphs already shown, while `analyze("../data/inflammation-02.csv")` should produce corresponding graphs for the second data set. Be sure to document your function with comments.

More functions

Working with several files (cont'd)

- How to save results ?
  i. add `pdf("inflammation-01.pdf")` before calling the function `analyze()`
  ii. add `dev.off()` after.

More functions

Functions + Loops = New functions !

- How to list all .csv files in a folder ? (in Bash : ls -a *.csv)

More functions

Functions + Loops = New functions !

- How to list all .csv files in a folder ? (in Bash : ls -a *.csv)
- Several solutions with `list.files()` :
    1. `list.files(path = "C:/.../data", pattern = "csv")`

More functions

Functions + Loops = New functions !

- How to list all .csv files in a folder ? (in Bash : ls -a *.csv)
- Several solutions with `list.files()` :
  1. `list.files(path = "C:/.../data", pattern = "csv")`
  2. `list.files(path = "C:/.../data", pattern = "inflammation")`

More functions

Functions + Loops = New functions !

- How to list all .csv files in a folder ? (in Bash : ls -a *.csv)
- Several solutions with `list.files()` :
  1. `list.files(path = "C:/.../data", pattern = "csv")`
  2. `list.files(path = "C:/.../data", pattern = "inflammation")`
  3. Bash-style ? `list.files(path = "C:/.../data", pattern = "*inf*.csv")`

More functions

Functions + Loops = New functions !

- How to list all .csv files in a folder ? (in Bash : ls -a *.csv)
- Several solutions with `list.files()` :
  1. `list.files(path = "C:/.../data", pattern = "csv")`
  2. `list.files(path = "C:/.../data", pattern = "inflammation")`
  3. Bash-style ? `list.files(path = "C:/.../data", pattern = "*inf*.csv")`

Solution 1 : use `pattern = glob2rx("*inf*.csv")`
Solution 2 : use `pattern = "inflammation-[0-9]2.csv"`

More functions

Functions + Loops = New functions !

- We want to run our function `analyze()` on all files :

More functions

Functions + Loops = New functions !

- We want to run our function `analyze()` on all files :
- **Exercice 1** : write a function `analyseall()` using `analyze()` in a loop.
  - Hint 1 : use the function `list.files()` to create a list of files and store it in `filenames`
  - Hint 2 : use a loop (`for f in filenames`)

Some help !

More functions

Functions + Loops = New functions !

- We want to run our function `analyze()` on all files :
- **Exercice 1** : write a function `analyseall()` using `analyze()` in a loop.
  - Hint 1 : use the function `list.files()` to create a list of files and store it in `filenames`
  - Hint 2 : use a loop (`for f in filenames`)
- Try to save every graph in a `pdf` file !

More functions

Conditions

- Our previous lessons have shown us how to :
  1. manipulate data
  2. define our own functions
  3. repeat things.
- However, the programs we have written so far always do the same things, regardless of what data they're given. We want programs to make choices based on the values they are manipulating.

More functions

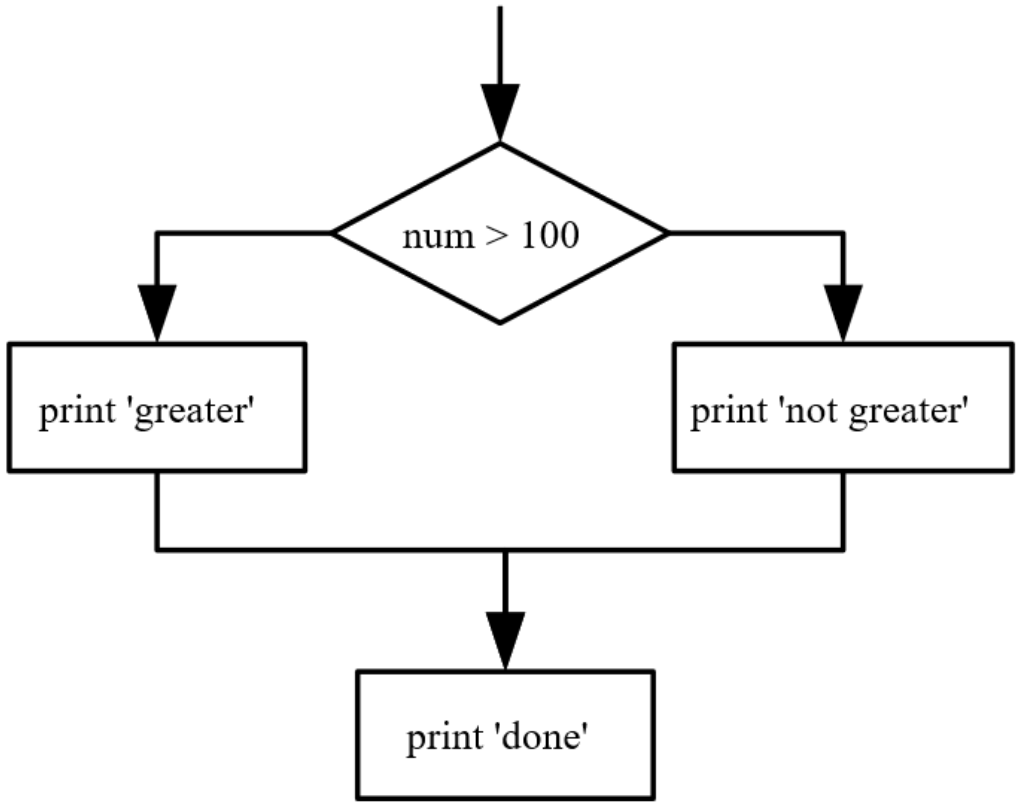
Conditions

- **Exercise 2** : What does this code do ?

```
num <- 37

if (num > 100) {
print("greater")
}

else {
print("not greater")
}

print("done")
```

More functions

Conditions



In the example above, the test `num > 100` returns the value FALSE, which is why the code inside the if block was skipped and the code inside the else statement was run instead.

More functions

Conditions

- **Exercise 3** : Write a function `sign()` that takes as argument a number and returns the value "negative", "null" or "positive"
- Hint : use `else, if, else if`

More functions

Conditions

- We can combine tests with **&** (both true) and | (one true or the other) :
- **Exercise 3bis** : Write a function `sign2()` that takes as argument two numbers and returns the value "both negative", "other", "both zero" or "both positive"

More functions

Conditions

- **Exercise 4** : Find the file containing the patient with the highest average inflammation score. Print the file name, the patient number (row number) and the value of the maximum average inflammation score.

```
# Exercice 4 (Hard!): Introduction to Programming
filenames <- list.files(path = "C:/../data", pattern = "inflammation-[0-9]{2}.csv", full.names = TRUE)

filename_max <- "" # filename where the maximum average inflammation patient is found
patient_max <- 0 # index (row number) for this patient in this file
average_inf_max <- 0 # value of the average inflammation score for this patient

for (f in filenames) {
  d <- read.csv(file = f, header = FALSE)
  d.means <- apply(d, 1, mean)
  for (patient_index in 1:length(d.means)){
    patient_average_inf <- d.means[patient_index]
    # Add your code here ...
  }
}
print(filename_max)
print(patient_max)
print(average_inf_max)
```

More functions

Conditions

- **Exercise 5** : Re-write the `analyze()` function with an option to save of not, using a second argument `output` that takes the default value NULL and using `if`
- Tips : Have a look to this function `!is.null()`

More functions

Key points :

1. Save a plot in a pdf file using pdf("name.pdf") and stop writing to the pdf file with dev.off().
2. Use if (condition) to start a conditional statement, else if (condition) to provide additional tests, and else to provide a default.
3. The bodies of conditional statements must be surrounded by curly braces

More functions

Key points :

1. Save a plot in a pdf file using pdf("name.pdf") and stop writing to the pdf file with dev.off().
2. Use if (condition) to start a conditional statement, else if (condition) to provide additional tests, and else to provide a default.
3. The bodies of conditional statements must be surrounded by curly braces
4. Use == to test for equality.
5. X & Y is only true if both X and Y are true.
6. X | Y is true if either X or Y, or both, are true.

Datatype

What and why ?

Everything in R is an object. R has 6 (although we will not discuss the raw class for this workshop) atomic vector types :

- character
- numeric (real or decimal)
- integer
- logical
- complex

Appendix

More functions

Functions + Loops = New functions !

```
analyzeall <- function(pattern) {

filenames <- list.files(path = "C :/Users/.../data", pattern = XXX,
full.names = TRUE)

for (f in filenames) {

XXX }

}
```

back