Writing For Loops, If-Else Statements, and Functions in R

• Institut Pasteur de Madagascar
• January 2020

• E²M²: Ecological and Epidemiological Modeling in Madagascar
The Power of Programming

• So far, much of what we saw demonstrates how to use R like an extremely smart calculator.
  • We write commands and it executes them.
The Power of Programming

• So far, much of what we saw demonstrates how to use R like an extremely smart calculator.

• The true power of the program comes from allowing R to query large datasets and make decisions for you.
The Power of Programming

• So far, much of what we saw demonstrates how to use R like an extremely smart calculator.

• The true power of the program comes from allowing R to query large datasets and make decisions for you.

• Three key programming tools are helpful:
  1. If-else statements
  2. For-loops
  3. Functions
• So far, much of what we learned demonstrates how to use R like an extremely smart calculator.

• The true power of the program comes from allowing R to query large datasets and make decisions for you.

• Three key programming tools are helpful:
  1. If-else and ifelse statements
  2. For-loops
     Allow you to control the flow of our programming and cause different things to happen depending on the value of tests
  3. Functions
For-Loops

```r
for (variable in vector) {
  do something
}
```

For (i in 1:10) {print i}
For-loops

• “Looping”, “cycling”, “iterating” is nothing more than automating a multi-step process by organizing sequences of actions or ‘batch’ processes and by grouping the parts that need to be repeated.

• For loops execute for a prescribed number of times, as controlled by a counter or an index, incremented at each iteration cycle.
If Statements

If condition is TRUE, then perform some action; otherwise do not perform that action.

if (condition is TRUE)
    { do something }
If condition is TRUE, then perform some action; otherwise do not perform that action.

if (condition is TRUE)
    { do something }
else { do different thing }
If-Else Statements

If condition is TRUE, then perform some action; otherwise do not perform that action.

```plaintext
if (condition is TRUE)
    { do something } else
    { do different thing }
```

It is important to note that else must be in the same line as the closing braces of the if statement.
A function is a piece of code written to carry out a specified task.

- `mean(x)`, `sum(x)`,...`rep(x,y)`

- Lots of pre-written functions organized in multitude of packages.

- If you can not find a function in R to do what you need, you can write your own function.
Functions

• Avoid writing repetitive lines of codes:
  • Reduce workload
  • Help avoid errors

• Make it easy to reuse and share task for different data sets/users
Functions

function_name <- function(arguments) {
    body
}

where the code in between the curly braces is the body of the function.
Functions

• Things to consider:
  • Function allows you to define exactly what you want to do
  • Name your user defined function.
  • Make sure that the name that you choose for the function is not an R reserved word. This means that you, for example, don’t want to pick the name of an existing function for your own UDF.
Let’s try it out in R

Open: “ForLoops_FunctionsWorkingScript.R”