Tutorial comments

**Installation:**

~~- don't use a NOT question. Just put there, check if you installed these, and they can either check them all or leave open when they haven't yet.~~

~~haha tidyverse = 'A code style that reads like poetry' ;p~~

**Introduction to R:**

~~- sentence is a bit weird: ‘It is important to always work with a script, so you can save and look back what you did code.’ suggestion: ‘It is important to always work with a script, so you can save and look back at previously written code’~~

~~- maybe change a bit: ‘Therefore, your environment should consist of four windows...’ suggestion: ‘When you opened a script, your environment should consist of four windows...’~~

~~- ‘The value~~ *~~a~~* ~~is stored in the global environment’, technically I think it works similar to Python.~~ *~~a~~* ~~is a variable pointing to the vector: c(“Hello”, “world”). I think “Hello” and “world” are values here. Not sure how hard it is needed to explain that, but I think calling~~ *~~a~~* ~~a value might be wrong.~~

~~- quiz: ‘Does your RStudio environment look the same as Figure 2?’, might be good (if possible) to separate the answers if multiple are counted as incorrect. It is hard to separate which answers fits which quiz option.~~

~~I think the screenshots are very useful!~~

**Functions and objects:**

~~- ‘numbers <- c(1,2,3,4,5)’ use proper style, e.g. add spaces: numbers <- c(1, 2, 3, 4, 5) (see also~~ [~~adv-r.had.co.nz/Style.html~~](http://adv-r.had.co.nz/Style.html)~~)~~

~~- ‘All the things you do in R are functions’, suggestion: ‘You can make R do stuff using functions’~~

~~- Exercise ‘print the object: mean\_numbers’: I get an ‘object 'mean\_numbers' not found’ error. I think variables don’t go from one to a next code chunk.~~

~~- Exercise ‘print the object: mean\_numbers’: add what they should find if they done it right (e.g. [1] 3).~~

**Functions**

~~- ‘Let’s look for example at drawing~~ **~~a sample~~** ~~from a normal distribution’~~

~~- Figure 3, a bit of stats nitpicking: numbers themselves are not random, they can be drawn from a random variable, but once they are numbers, they are realizations and not random anymore.~~

~~- Before the quiz, it might be useful to show~~ *~~how~~* ~~they get to the help page (e.g. ?rnorm or search in bar Rstudio).~~

~~- quiz: ‘Why isn't Hello replaced after running the following function: `sub('hello', 'goodbye', 'The agent said: Hello stranger')?’ beware that both answer a and c can be considered true, they are both needed to prevent sub() from detecting ‘Hello’.~~

**Objects**

~~- general comment: I am not sure how it works in R, but be careful of what you call variables and objects. I think an object is the data itself (always generated by using a function) and a variable points to this object (variable is the name). An object gets removed from memory when~~

~~- Maybe add to the end that data types is the next topic where they will learn more about this?~~

**Data types**

~~- ‘You can check this with~~ **~~the function~~** ~~mode()’~~

~~- Exercise a: mmaybe already put the code for assigning the objects in the code chunk?~~

~~- Factors: ‘In the example, we see a vector of nine elements, that consists of three levels. Instead of saving all nine elements separately, it is more useful to store them as three levels of the unique values.’ I find it still a bit unclear why factors are useful sometimes. Is it memory-wise cheaper to save a numeric vector with its levels instead of character vector?~~

~~- Logical and Boolean operators: very nice part! Maybe add that T and F also refer to TRUE and FALSE? Although that may be unnecessary.~~

~~- Figure caption incorrect!~~

~~- Exercise 2: proper style consider using vector\_exercise instead of Vector.exercise.~~

**Data structures**

~~- Vector last example: proper style consider using one\_missing instead of one.missing.~~

~~- Dataframe: maybe add an example of how to create a data frame? You are using it in the next topic already.~~

**Programming structures**

~~- ‘Programming structures in R allow you to control the flow of various R expressions. They respond to certain features of the data and execute different R expressions accordingly.’ A bit abstract and unclear what you mean here.~~

~~- Maybe add something on how these are the most basic building blocks for functions.~~

~~- ‘So in this case, z is evaluated to be above or equal to 5.5, which it isn’t. Therefore, you don’t have to specify in the next statement that z needs to be between 4 and 5.5 to be able to do the resit.’ It is important to specify that you are also referring to code below. I was still looking at the example code above the text.~~

~~- average\_grades: hard to already use a data.frame in this way? (with indexing).~~

~~- ‘If you want to use an if-else statement on a vector or variable, you have two options.’ You have more options, since you can for example use apply, which they need not know, but maybe state that they have multiple options for example...~~

~~- proper style: ‘ifelse(average\_grades$avg>=8, "Passed cum laude", ifelse(average\_grades$avg>= 5.5, "Passed", "Failed"))’ -> add spaces e.g. ‘average\_grades$avg >= 8’~~

~~- Exercise ‘Look at the table~~ **~~standing~~** ~~below’~~

~~- Exercise, might be tricky for them to work with nesting (not perse a bad thing, but something I wanted to note)~~

~~- For: ‘for (x in 1:10)’ did you introduce the sequence creation with ‘:’ already before? Otherwise it might be better to use c(1, 2, 3, 4, 5, 6)?~~

**Read and write data**

~~- ‘— not sure what to advise here, changing the regional settings or not? —‘, they shouldn’t need to change their system settings, but they can use the arguments ‘dec’ and ‘sep’ in read.csv() to circumvent this problem.~~

~~- all(NEWNAME ==~~**~~=~~** ~~Theoph), remove one ‘=’~~

~~- Maybe it is good to inform the students in some way about relative and absolute paths? If you always use absolute paths, you never need to change your working directory. If you always use relative paths, it is easier to let others replicate your code.~~

~~- The last button ‘continue’, does currently not lead to anything. Might be nice to say something like: ‘well done! You finished your first course in R! Please feel free to revisit parts that were difficult, or continue with the next tutorial XXXX’~~