Tutorial 1. Types in R

In this tutorial we will explore what different data types can be defined in R. In particular we will focus on Numeric, NA, Factors, strings, and Boolean variables. First we will take a few first steps in the R command line. You can copy and paste the code in this file, which will be denoted as follows:

This is regular text

```
this is a line of code
```

```
# [1] This is a returned value
```

The very first steps

In the first step we will define some variables in memory and call them in the prompt:

```
# first we define a variable a - 3. Note that we use <- to assign values to
# variables. Using the = sign is also possible, but this can lead to
# confusion in other operations
a <- 3
# We can find the value of a by typing a into the prompt
a
```

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

```
# if we look for another variable that has not been defined we get an error:
b
```

```
## Error: object 'b' not found
```

From this exercise we have learned a few things:

- Use # for comments. The code after the # sign is not *evaluated*
- Variables can be defined with letters. The name for a variable can be a letter or word. It can include numbers and full stops, but these should not be at the start of the variable name. Colons are not permitted either.

```
variable <- 22
# this does not work : 1variable <- 22
variable.1 <- 22  # works
```

These rules are called the *syntax* of the language. They define how we can interact and comunicate with the machine through the R language. We will find many other along the workshop.

Class “Numeric”

Variables have classes. If they are numbers they are of class “numeric”. To obtain the class of a variable we can use the function class()

```
variable1 <- 22
class(variable1)
```

```
## [1] "numeric"
```

We are now familiar with two key components of programming languages: Variables and functions. A variable stores some information, and a function is an operation applied to a variable.The syntax depends on the language. In R to call a function on a variable we always use the round parentheses.

Functions can also be called on values that are not saved in variables:

```
class(22)
```

```
## [1] "numeric"
```

Types “NA” and “NaN”

Sometimes we have missing data. This can be defined with NA or NaN (not a number) in R.

```
variable2 <- NA
class(variable2)
```

```
## [1] "logical"
```

```
variable3 <- NaN
class(variable3)
```

```
## [1] "numeric"
```

This can be useful if we are using a method with different handling for missing data.

Class “Character”

R can handle text data. In some programming languages these data are called “strings” or “characters”. These can be defined in R by using quotes:

```
variable4 <- "this is variable 4"
class(variable4)
```

```
## [1] "character"
```

Exercise

Make some numeric variables and try some mathematical operations. Try storing the result in other variables. For example:

```
my_variable1 <- 5
my_variable2 <- 10
my_sum <- my_variable1 + my_variable2
my_sum
```

Try breaking some of the rules of variable names and see what errors you get.