Exercise Series 1

1. Vectors

What is the output of the following commands? Try to predict the solutions before you type in the commands. We define:

```
x <- c(5, 2, 1, 4); xx <- c(1, 10, 15, 18); y <- rep(1, 5)
z <- c(TRUE, FALSE, TRUE, TRUE); w <- c("Marie", "Betty", "Peter")
```

a)
```
sum(x)
range(x)
length(y)
sum(y)
```

b)
```
c(x, y, 13)
```

c)
```
xx - x
c(x, 12) * y
1:6 + 1
1:9 + 1:2
```

d)
```
x <= 2
x <= 2 & z
```

e)
```
substring(w, 2, 4)
paste(substring(w, 1, 2), substring(w, 5, 5), sep = "...")
```

f)
```
cbind(x, xx)
cbind(2, 6:1, rep(c(3, 1, 4), 2), seq(1.1, 1.6, by = 0.1))
```

2. Sequences of numbers

Create the following sequences. Use the commands `rep()` and `seq()`.

a)
```
## [1] 1 2 3 4 5 6 7 8 9
```

b)
```
##  [1] "m" "w" "m" "w" "m" "w" "m" "w" "m" "w"
```

c)
```
##  [1] 1 2 3 4 1 2 3 4 1 2 3 4
```

d)
```
##  [1] 4 4 4 3 3 3 2 2 2 1 1 1
```

Hint: Use the argument `each` of the function `rep()`.

e)
```
##  [1] 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
```

f)
```
##  [1]  1  1  3  3  5  5  7  7  9  9 11 11
```

3. Matrices and data frames

a) Generate the following matrices.

```
##      [,1] [,2] [,3] [,4]
## [1,]    1  101  201  301
## [2,]    2  102  202  302
## [3,]    3  103  203  303
## [4,]    4  104  204  304
## [5,]    5  105  205  305
```

```
##      [,1] [,2] [,3]
## [1,]    5    0    0
## [2,]    0    5    0
## [3,]    0    0    5
```

b) Define following data frame:

```
##    Namen Age Blind
## 1    Bob  27  TRUE
## 2  Alice  34 FALSE
## 3    Kim  21 FALSE
## 4  Julia  25  TRUE
## 5 Robert  29  TRUE
```

c) Explore the properties of your generated objects. Which class of R-objects do they belong to? How are they structured?

Hint: Use the functions `class()`, `dim()`, `str()`, `summary()`.

4. Lapwings

For various meadows at Zurich Airport, we counted the daily number of lapwings on several occasions. For every bird, we noted the kind of activity (resting, feeding, flying) as well as the ground conditions (damp, dry, wet). The data was stored in `vogel.dat`.

It contains the columns date (`Datum`), time (`Zeit`), id of meadow (`Feld.Nr`), count (`Anzahl`), activity (`Taetigkeit`) and ground condition (`Boden`). The factor variable `Taetigkeit` has the three levels: `ru` (resting), `fr` (feeding), `fl` (flying), and the factor variable `Boden` has the three levels `n` (wet), `t` (dry), `f` (damp).

a) To read in the data into R, type the following command:

```
url <- "http://stat.ethz.ch/Teaching/Datasets/NDK/vogel.dat"
d.vogel <- read.table(url, sep = ";", header = TRUE)
```

Take a look at the data using the functions `str()` and `summary()`.

b) Create a new data frame that only contains the meadow id and the counts. How many birds were counted on average?

c) Create a data frame only with the data of meadow 1413.

d) Create a vector that contains the number of birds of meadow 1413.

e) On how many occasions (days) did one observe feeding birds? How many birds were counted in total while feeding? What are the corresponding observation numbers?

f) Change the number of lapwings of the eighth observation (eighth row) to 6. Delete the third and seventh observation from the data frame.

Hint: `mean()`, `sum()`, `which()`.

5. Getting to know data: iris blossoms

The data set `iris` contains measurements of the length and the width (in cm) of petals and sepals of three iris species:

- 1: Setosa,
- 2: Versicolor,
- 3: Virginica.

(Source: R. A. Fisher, The Use of Multiple Measurements in Taxonomic Problems, Annals of Eugenics, Vol. 7, Part II, 1936, pp. 179-188.)

a) This data set `iris` is already part of the standard R installation. Consider the object `iris`. How is it structured? How many observations (lines) does it contain? How many variables (columns)?

Hint: `nrow()`, `ncol()`, `dim()`, `str()`

b) To get an overview of the range of values, look at the `summary()` of the data set. Which information on the data set does it provide?

c) For the variable `Sepal.Length` check the results above by using the R-functions `min()`, `max()`, `mean()`, `median()`, `quantile()`. If necessary, make use of the help functions `?quantile` etc.

6. Missing values

Statistics needs data. Unfortunately, data often cannot be collected fully. Therefore many data sets contain "gaps", non-existing measurements, so-called `NA`s (not available). In this exercise you will get to know how R deals with `NA`s. We work with the data set `iris`. Make a copy of the iris data set by using following command:

```
d.iris <- iris
```

a) Assume that we were unable to take the second observation of `Petal.Length` and `Petal.Width`. In addition, the data for `Sepal.Length`, `Sepal.Width` and `Petal.Width` are missing for the fifth observation. Replace these five fields by `NA`.

Hint: Replace the values by `NA`s using e.g.

```
d.iris[2, 3:4] <- NA
```

b) Consider the first eight observations of the modified data set, to observe how the `NA`s are displayed by R. The commands `class()`, `nrow()`, `ncol()`, `dim()`, `str()` also work for the data set with missing values. What does change in the `summary()`?

c) Try to confirm the given values for the variable `Sepal.Length` using `min()`, `max()`, `mean()`, `median()`, `quantile()`. Is there a difference?

d) There are functions that cannot handle `NA`s (Result '`NA`' or 'Error: missing observations'). There is a trick to make them calculate the correct results: simple functions such as `min()`, `max()`, `mean()`, `median()`, `quantile()`, `range()` etc. can take an argument `na.rm`. When you set its value to `TRUE`, the `NA`s will not be considered in the calculation. Try to confirm the values provided by `summary()` again, using this new argument.

e) Why should missing values always be coded by `NA`, and not, for instance, filled with a zero? Explain for the case of the `mean()` function.

f) Experiment with missing values in the statistical functions `var()`, `sd()`, `cor()`. Can you explain the behaviour of R?

g) Select only those observations which have missing values in either `Sepal.Length` or in `Petal.Length`.

Hint: Use the function `is.na()`.

h) The function `na.omit()` eliminates all observations from the data frame for which **any(!)** variable contains `NA`s. Save the result of `na.omit(d.iris)`. How many observations remain? How many remain using `na.omit(d.iris[ ,1:3])`?

Note:

Higher-level functions such as `t.test()` or `wilcox.test()` have an argument `na.action`, with which the reaction to `NA`s can be determined. `na.action = na.omit` first deletes all lines (observations) with `NA`s before anything is calculated.