Hints for Exercise Sheet 1

Call the `R` help to learn more about an `R` function:

1. `?function.name` to get help directly in the `R` window
2. OR (recommended)
   type `options(help_type="html")` into `R`, and use `?function.name` to access the `HTML`-help

Comments in `R`:

- # in front of the respective line

Read data into `R`:

- Read in a dataset: `dataset<-read.table("path.txt", header=TRUE)`
- Make a dataset stored in an `R` package accessible:
  `library(splancs)`
  `data(southlancs)`
- Source an `R` function or an `.RData` object: `source("germany.txt")`, `source("asthma.RData.txt")`

How to explore a dataset:

- Use `names(dataset)` to see which variables are included in the dataset
- Use `head(dataset)` to see the first rows of a dataset
- Address a variable in the dataset by using `dataset$variable.name`

Install and load an `R` package:

- Package is available from CRAN (`www.r-project.org`), but not yet installed: `install.packages("pkg")`
- Load an installed `R` package: `library(pkg)`

Types of objects in R:

- Vector:
  - Create a vector: `vec<-c(0,1,2)`
  - Extract k-th element of a vector: `vec[k]`
  - Initialize a vector with n elements: `rep(0,n)`
- Matrix:
  - Create a matrix: `m<-matrix(0,nrow=I,ncol=J)`
  - Extract i,j-th element of a matrix: `m[i,j]`
  - Extract i-th row of a matrix: `m[i,]`
  - Extract j-th column of a matrix: `m[,j]`
  - Subset a matrix: e.g. `subset<-m[m[,1]==1,]`
  - Add a column/row to a matrix: `cbind(), rbind()`
  - Dimension of a matrix: `dim(m)`
- List:
  - A list contains several objects which can be of a different type
  - Extract the k-th object of list l: `l[[k]]`
  - Extract the i,j-th element of the k-th object of a list l: `l[[k]][i,j]`

Use implemented `R` functions:

- Look at the `R` help of this function: `?function.name`
- Which arguments have to be provided for the function?
  `function.name(argument1, argument2,...)`
- Call the function, e.g. `glm()`: `model.name<-glm(argument1,argument2)`
- Results:
  - Which components does the result have: `names(model.name)`
  - Look at a component of the result: `model.name$component1`
  - Look at the model summary: `summary(model.name)`

Write your own function:

- Use the following syntax:

```
euclid<-function(argument1,argument2,...){
-- calculations --
return(result)
}
```

- For `for`-loops use the following syntax:
  `for (i in 1:n){ }`
- For `if`-loops use the following syntax:
  `if (a==1){ }`

Plots in R:

- Use function: `plot(x,y)`
- Arguments which can be passed regarding caption and axes of a plot: `xlab="..."`, `ylab="..."`, `xlim=c(c1,c2)`, `ylim=c(c1,c2)`, `main="..."`
- Arguments which can be passed regarding type of lines and points: `type="..."`, `pch="..."`
- Add points to an existing plot: `points(x,y,pch="+")`