```
#This is (more or less) what you should see when you start R:

R version 2.9.1 (2009-06-26)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.28 (5444) i386-apple-darwin8.11.1]

>

#R works fine as a simple calculator
#Comments, starting with "#" are not executed
#Prompt">" marks the line beginning
#Until you end an expression, each line below starts with "+"

> 123

> 1+2

> 1+
+ 2+
+ 3

> (1+3)*4

> 2^3

>log2(8)

#Data in R is organized in named objects.
#To assign a value to an object, use "<-".

> a<-3

#To see a value of an object, simply type its name
> a

#Operations on objects are done, but unless the new assignement happens, objects remain the s
> a*4

> b<-12

> a*b

> a
```

```
> b

>a<-a*3

>a

#Objects may be numeric or character (also other modes exist)
> c<-"word"
> c



#Some operations are allowed only on  objects of certain mode
> c+2

> mode(c)

> mode(a)

> d<-a+b

#Prepare your own objects of mode numeric, character, logical.

#Now there are several objects in the workspace. To see them all:
> ls()



#ls is a function. Functions are called with (), unless we want to see
#what are they doing (complicated code):
> ls

#Within parenthesis are arguments for a function.

#Objects might be saved for future use
#Arguments for save() function: object names and a filename

> save(a,b,c,d,file="testfile.Rdata")

#But where is this file saved? When no path is specified, in the working directory.
> getwd()

> ls()


#Removing an object
> rm(b)


> ls()


> rm(a,c,d)
> ls()



#And now we may load saved data
> load("testfile.Rdata")
> ls()
```

```
#Those simple single value objects are scalars. A more challenging data type is a vector:
#Vector may be of numeric, character and other modes - but all values have the same mode.

> vector<-c(1,3,7)
> vector

> mode(vector)



> vector<-c("first","second", "third")
> vector

> mode(vector)



#If not, R will force it:

> vector<-c(1,3,7,"word")
> vector


#Notice "" around 1, and indeed:

> mode(vector)


#Simple way to assign consecutive values is ":"
> vector<-1:11
> vector



> vector<-c("first","second","third","fourth","fifth")
> vector


#What is a first element of a vector?
> vector[1]

> vector[4]


#Subset may contain more than one element:
> vector[2:3]


# and in any order:
> vector[c(1,3,2,5)]


#Also repeated. But index must be equal/smaller than the vector length!
>  vector[c(1,3,2,2,125)]


#NA =NonAvailable, very important notion!
```

```
#Simple way to eliminate elements
> vector[-3]


#How to check vector length
> length(vector)

> length(vector[-3])



> new<-1:5
> new

> new<-5:1
> new
[1] 5 4 3 2 1

#Operations may be executed on vectors. Each element is treated separately:
> 2*new

> new+3


#we can even multiply/add etc. vector by another vector:
> new*new


#But waht happens, when their lengths differ? Vectors are recycled
> new*(1:3)


> vector

> vector*2


> vector

#Substituting elements:
> vector[2]<-3
> vector

#How to add elements
> vector<-c(vector,"last")
> vector



#Vector as concatenation of vectors:

> newest<-c(new,123,new*2)
> newest


> a<-c(4,6,7,3,13,5)

#Logical operations
> a>3
```

```
> a==3

> a>=3

> a>4


> b<-(a>3)
> mode(b)


#logical vector may be used for subsetting:
> a[a>4]


#Function which, tu just get subscripts:
> which(a>4)



> a[which(a>4)]


#More about function which.
> ?which
> help('*')
> help.start()


#Sometimes I have only a vague memory that a function exists:

> apropos('%')

> help("%/%")



#Vector elements may have a name
> names(a)

> names(a)<-c("daphnia","stickleback","horse","horse","Darwin")
> a


#Not every element has a name, let's try change it:
> names(a[6])<-c("last")
> a
  names(a)[6]<-c("last")
> a

#Vector may be subset by names:
> a["last"]


#Be aware of doubled names!
> a["horse"]


> vector<-c(vector,a)

#Not all elements have to have a name
```

```
> vector

  > names(vector)


> length(vector)


#Vector may be elongated by adding an element with nonexisting subscript:
> vector[16]<-4
> vector


#When quitting R:
> q()


#Current workspace = all objects we created. Workspace may be saved and loaded.

#Conventions for object names
#different files:
#.Rhistory,
#.R,
#.Rdata
```