Last edited: 2016-04-11 19:10:00

R as a calculator

R can be used as a simple calculator. Standard syntax is valid.

```
1 + 1
```

```
## [1] 2
```

```
5^2
```

```
## [1] 25
```

```
sqrt(25)
```

```
## [1] 5
```

Object definition

To define objects in R we use `<-` to assign a value (or anything really) to a named object.

```
one <- 2
one + one
```

```
## [1] 4
```

```
one + 1
```

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

```
mytext <- c("I", "am", "hungry")
mytext
```

```
## [1] "I"      "am"     "hungry"
```

Getting help

To access R's built-in help pages use `help(<functionName>)` or simply `?`

```
?c
```

Working directory

To set the working directory (i.e. the folder on the disk where R is pointing to) use `setwd()`

```
getwd()
```

```
## [1] "/home/ede/tappelhans/uni/marburg/lehre/git/pages/marburg-open-courseware.github.io/bsc/envchang
```

```
setwd("/home/ede/tappelhans/uni/marburg/lehre/2016/ss/PS_global_change")
```

Object types

First rule of R: *Everything is possible*

Second rule of R: *Everything is a vector*

vectors

```
vec <- 1:10
vec
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
mytext <- c("I", "am", "hungry")
mytext
```

```
## [1] "I"      "am"     "hungry"
```

```
newvec <- c(vec, mytext, NA)
newvec
```

```
##  [1] "1"      "2"      "3"      "4"      "5"      "6"      "7"
##  [8] "8"      "9"      "10"     "I"      "am"     "hungry" NA
```

matrices

```
mat <- matrix(1:9, ncol = 3)
mat
```

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

data frames

```
df <- data.frame(var1 = 1:5,
                 var2 = c("a", "b", "c", "d", "e"),
                 var3 = rep(NA, 5))
df
```

```
##   var1 var2 var3
## 1    1    a   NA
## 2    2    b   NA
## 3    3    c   NA
## 4    4    d   NA
## 5    5    e   NA
```

lists

```
lst <- list(vector = vec,
            text = mytext,
            NA,
            mat,
            df = df)
lst
```

```
## $vector
##  [1]  1  2  3  4  5  6  7  8  9 10
##
## $text
## [1] "I"      "am"     "hungry"
##
## [[3]]
## [1] NA
##
## [[4]]
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
##
## $df
##   var1 var2 var3
## 1    1    a   NA
## 2    2    b   NA
## 3    3    c   NA
## 4    4    d   NA
## 5    5    e   NA
```

Object structure

we can check the internal structure of any object using `str()`

```
str(vec)
```

```
##  int [1:10] 1 2 3 4 5 6 7 8 9 10
```

```
str(mat)
```

```
##  int [1:3, 1:3] 1 2 3 4 5 6 7 8 9
```

```
str(df)
```

```
## 'data.frame':    5 obs. of  3 variables:
##  $ var1: int  1 2 3 4 5
##  $ var2: Factor w/ 5 levels "a","b","c","d",..: 1 2 3 4 5
##  $ var3: logi  NA NA NA NA NA
```

```
str(lst)
```

```
## List of 5
##  $ vector: int [1:10] 1 2 3 4 5 6 7 8 9 10
##  $ text  : chr [1:3] "I" "am" "hungry"
##  $       : logi NA
##  $       : int [1:3, 1:3] 1 2 3 4 5 6 7 8 9
##  $ df    :'data.frame':  5 obs. of  3 variables:
##   ..$ var1: int [1:5] 1 2 3 4 5
##   ..$ var2: Factor w/ 5 levels "a","b","c","d",..: 1 2 3 4 5
##   ..$ var3: logi [1:5] NA NA NA NA NA
```

Accessing object content

there are different ways of accessing the content of objects

using $

For objects that have named variables we can use $

```
df$var1
```

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

```
lst$df
```

```
##   var1 var2 var3
## 1    1    a   NA
## 2    2    b   NA
## 3    3    c   NA
## 4    4    d   NA
## 5    5    e   NA
```

```
lst$df$var1
```

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

[and [[

we can also use [with index numbers

```
mat[1, 3]
```

```
## [1] 7
```

```
df[1, 3]
```

```
## [1] NA
```

for lists we need [[two square brackets

```
lst[[1]]
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

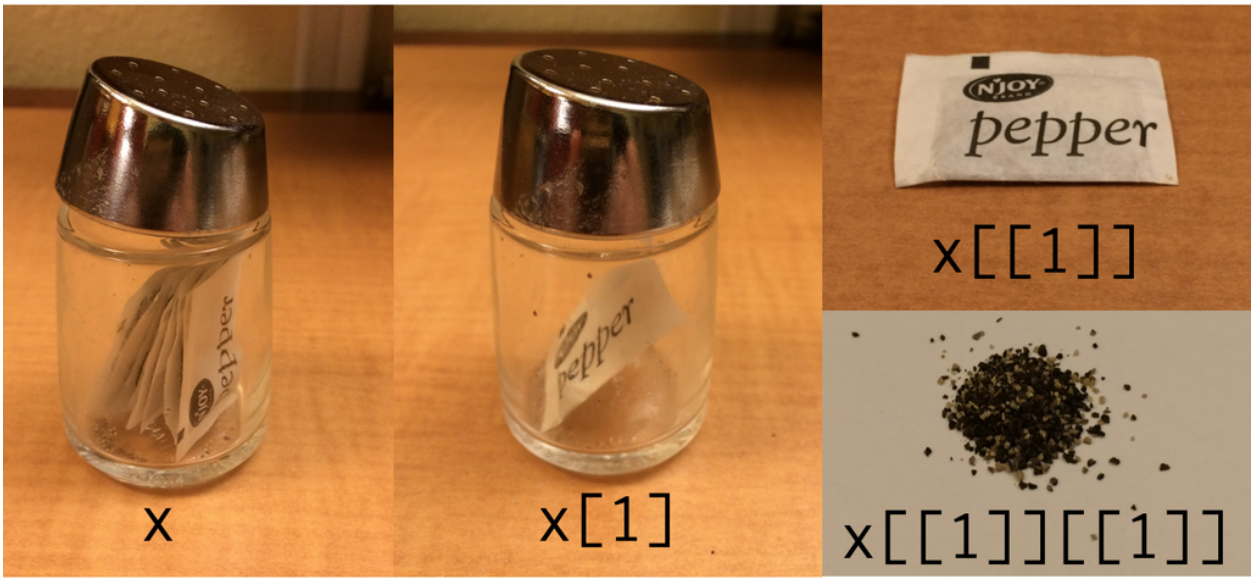
```
lst[[5]][[2]]
```

```
## [1] a b c d e
## Levels: a b c d e
```

```
lst[[5]][[2]][[3]]
```

```
## [1] c
## Levels: a b c d e
```

Or as Hadley Wickham explains more visually:



Source: @hadleywickham on Twitter http://twitter.com/hadleywickham/status/643381054758363136

combinations also work

```
lst[[5]]$var2
```

```
## [1] a b c d e
## Levels: a b c d e
```

Saving data

we can easily write data to the disk

```
write.csv(df, "my_dataframe.csv", row.names = FALSE)
```

Loading data

and just as easy we can read data from the disk

```
dat <- read.csv("my_dataframe.csv")
dat
```

```
##   var1 var2 var3
## 1    1    a   NA
## 2    2    b   NA
## 3    3    c   NA
## 4    4    d   NA
## 5    5    e   NA
```

as a final note, we can also remove files from the disk

```
file.remove("my_dataframe.csv")
```

```
## [1] TRUE
```