```
#Prepare a vector with number of eggs in a nest, with 6 elements:
eggNest <- c(1,3,4,0,5,3)


#Check if they have names
> names(eggNest)
NULL


#Give names to vector elements:
> names(eggNest) <- c("sparrow", "crow", "finch","sparrow", "stork")

> eggNest
sparrow    crow   finch sparrow   stork    <NA>
      1       3       4       0       5       3

#Check names
> names(eggNest)
[1] "sparrow" "crow"    "finch"   "sparrow" "stork"   NA


#Not every element has a name. Give sixth element a name "eagle"
> names(eggNest)[6] <- "eagle"

> eggNest
sparrow    crow   finch sparrow   stork   eagle
      1       3       4       0       5       3


#Take eggNest element with name "finch"
 > eggNest["finch"]
finch
    4

 #Take eggNest element with name "sparrow". Be aware of doubled names!
> eggNest["sparrow"]
sparrow
      1

#Which elements of the vector "eggNest" are bigger than 2?
> eggNest>2
sparrow    crow   finch sparrow   stork   eagle
  FALSE    TRUE    TRUE   FALSE    TRUE    TRUE
> which(eggNest>2)
 crow finch stork eagle
    2     3     5     6

#What are the names of birds which had less than 3 eggs?
> names(eggNest)[eggNest<3]
[1] "sparrow" "sparrow"


#Change second sparrow entry into "housesparrow".
> names(eggNest)
[1] "sparrow" "crow"    "finch"   "sparrow" "stork"   "eagle"
> names(eggNest)[4]<-"housesparrow"

#Which vector element  name is "stork"?
which(names(eggNest)=="stork")

#Change number of eggs for eagle into 4.
```

```
> eggNest["eagle"]<-4
> eggNest



#Another data type: factor
#Prepare a vector with place of origin of 15 fish samples

lakes
 [1] "Schoeh" "Schoeh" "Schoeh" "GrPl"   "GrPl"   "GrPl"   "KlPl"   "KlPl"   "KlPl"
"Tramm"  "Tramm"  "Tramm"  "other"  "other"  "other"

>lakes<-rep(c("Schoeh","GrPl","KlPl","Tramm","other"), each =3 )

#Prepare a vector with number of fish caught at each location
 fish
 [1] 6 3 8 3 7 7 7 0 2 9 7 1 5 7 0
> fish<-c(6, 3, 8, 3, 7, 7, 7, 0, 2, 9, 7, 1, 5, 7, 0)

#Compute average number of fish caught
>mean(fish)

#Compute mean number of fish for each lake separately
#Each lake is treated as a separate level of a factor "lakes"and function is applied to
each level
#tapply() applies a function for each level separately.
#tapply(vector of data, vector with level assignement, function)
>lakesFactor<-factor(lakes)
>lakesFactor
> tapply (fish, lakesFactor, mean)

# Get by lake max, median, sum...

>tapply(fish, lakesFactor,max)
>tapply(fish, lakesFactor,median)
>tapply(fish, lakesFactor,sum)


#Check what happens when you use lakes instead of lakesFactor
#R by default converts lakes into factor

###########

#There are a plenty of example datatsets provided with standard R installation. To see
them, use function data():
> data()

#To use one of them, give dataset name as an argument:
> data(chickwts)


#Display chickwts object. What class it is?
>chickwts
> class(chickwts)


## This is a dataframe. It looks as a matrix or table, in reality it is a bunch of
## vectors of the same length. Vectors may have different mode (one character, another
logical etc)
#check dimensions/numer of columns and rows of chickwts
>dim(chickwts)
```

```
>ncol(chickwts)
>nrow(chickwts)

#Subset this data frame by first row
> chickwts[1,]

#And by first column
> chickwts[,1]


#second column
#feed values are here as factors. No "", levels.
> chickwts[,2]


#Subset the dataframe by "feed" column, using its name (there are 2 ways of doing it)
>chickwts$feed
>chickwts[,"feed"]


#Take first 4 rows from all columns
> chickwts[1:4,]

#Take rows 3,7,12,1 from all columns
> chickwts[c(3,7,12,1),]


#Select rows 3,7,12,1, from column 2
> chickwts[c(3,7,12,1) , 2]
#or:
>subset<-c(3,7,12,1); chickwts[subset, 2]



#transpose the dataframe and check dimensions now
>t(chickwts)
>dim(t(chickwts))


#Apply a function "mean" on the whole dataframe. What it did?
> mean(chickwts)

#Repeat it for the first column
> mean(chickwts[,1])

#Compute mean chciken weight for each type of feed separately
##tapply = applying function separately for each level of a factor.
> tapply(chickwts[,1],chickwts[,2],mean)
or
>tapply(chickwts$weight,chickwts$feed,mean)




#########
##Make your own dataframe, with data about bacterial growth
#It should contain columns "medium", "strain" and "growth".
#Medium are 4 repetitions of a set "minimal","full","min+vit", mode:factor
#strain: "normal" and mutant, each repeated 6 times, mode:factor
#growth: values 2,4,5,2,3,6,1,4,2,3,6,5
```

```
BacterialGrowth<-data.frame(
        medium=as.factor(rep(c("minimal","full","min+vit"),4)),
        strain=as.factor(rep(c("normal","mutant"),each=6)),
        growth=c(2,4,5,2,3,6,1,4,2,3,6,5))

#Compute average growth
# by strain
# by medium

tapply(BacterialGrowth$growth,BacterialGrowth$strain,mean)
tapply(BacterialGrowth$growth,BacterialGrowth$medium,mean)

#Write it differently

tapply(BacterialGrowth[,"growth"],BacterialGrowth[,"strain"],median)
tapply(BacterialGrowth[,3],BacterialGrowth[,2],median)


#Read help file for the function write.table() and write BacterialGrowth in a
tab-delimited txt file

write.table(BacterialGrowth, file="bactGrowth.txt")
############################

#######################
#How to load data. Have a look on a .txt file with daphnia measurements
#read helpfile for read.table() function

#Read-in data from daphnia.txt file. Keep the header!
daphnia<-read.table("daphnia.txt",header=TRUE)

#Compute average size by clone
tapply(daphnia$size,daphnia$clone,mean))

#Compute average size by medium
tapply(daphnia$size,daphnia$medium,mean))

# Array - simplest array is matrix. Prepare 4 x 5 matrix, containing numbers from 1 to 20.
Supply: data, ncol, nrow. By default fills by columns!
a<-matrix(1:20,4,5)

#Indexing of matrix: as for dataframe
#Take element from second row, third column
a[2,3]

#Take elements from first and second row, third, fourth and fifth column
a[1:2,3:5]

#Take element from all except first rows, second and third column
a[-1,2:3]

#Take fourth element of the matrix
a[4]

#Take fourth column
a[,4]

#Name columns "col1" etc.
> colnames(a)<-c("col1","col2","col3","col4","col5")

#Name rows "row1" etc.
```

```
> rownames(a)<-c("row1","row2","rowl3","row4")

#Take elements from second column, rows 2 and 3
> a[c("row2","rowl3"),"col2"]

#Differences between matrix and dataframe
#In matrices all elements have to be the same mode. Here R coerces everything into
character.
>ac<-a
> ac[,"col2"]<-c("a","b","c","d")
> ac

#it is treated as an array, not as a bunch of vectors. Compare result of names() for
daphnia and a
>names(a)
>names(daphnia)

#Also matrix$columnname does not work.

#Change matrix into a dataframe with as.data.frame()
#check output of functions names(), mean() for a and b

 b<-as.data.frame(a)

 >b[,"col2"]<-c("a","b","c","d")

 >names(b)
 >b$col2
 >mean(b)
 >mean(a)


 #Substitute values in col2 of b by 34,23,2,1
 > b$col2<-c(34,23,2,1)

#Add last column (with numbers 1:4) to the matrix a with function cbind()
>cbind(a, lastcol<-1:4)

#Add a column to the dataframe b with function cbind()
#To the same defining a new vector
>cbind(b, lastcol<-1:4)
>b$lastcol <- 1:4
>b


#Compute mean of rows and mean of columns for a and for b
>apply(a,1,mean)
> apply(a,2,mean)
>apply(b,1,mean)
> apply(b,2,mean)

#Above is faster done with specific functions:
rowMeans(a)
colMeans(a)
```