Exercise 1.1:

If X to be normally distributed with some values for mean and variance, say, $X \sim N(\mu, \sigma^2)$, is the mean of $Y = e^X$ then

a) $e^{\mu+\sigma^2/2}$ or

b) $e^{\mu-\sigma^2/2}$?

Use the `rnorm()` function and the other functions from the first slide in the introduction session to figure it out.

Exercise 1.2:

Consider the second order polynomial equation

$$X^2 + 3X + 1 = 0 \qquad (a)$$

The general polynomial equation

$$AX^2 + BX + C = 0$$

has the solutions

$$\frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

1. Construct a vector of length 2 that contains the solutions to the equation (a), and display it on the screen with 1 decimal point.
2. Work out how much error you make (in percent) by referring to the solutions with just one decimal. The result will be different for the two solutions.

Exercise 1.3:

Construct the object x as

```
x <- rnorm(100, mean=.5, sd=.3)
```

Perform the following task (check out the help system as needed):

1. Calculate mean and standard deviation of x.
2. Plot a histogram of x.
3. Put the second axis on the right side of the histogram plot instead of on the left.

Exercise 2.1:

a) Study the function paste() via its help page and use it to paste together the string ”R session” and the number 1. What type of object is returned?

b) What happens if we try to add a number to a data frame containing only numbers?

c) What happens if we try to add a number to a data frame with both number columns and character columns?

Exercise 2.2:

a) Create a function that takes a vector of numbers and returns the standardized values; ie. the data with the mean subtracted and rescaled so that the variance is 1.

b) Describe what happens if you apply your function to a vector of length 1, and why.

Exercise 2.3:

Consider the function

$$f(x) = 3\sin\left(\frac{x}{2}\right) + x$$

a) Define a similar function *f* in R. Calculate $f(0)$, $f(-1)$ and $f(\pi)$.
b) Write a script that plots the function *f* over the interval from -7 to 7. Save the script on your system and close it. Then, execute the script in R with the source() command.

Exercise 3.1a:

a) Write a loop that investigates the contents of a matrix my.data. For each row starting from the first, the loop should write a line in the output window which differs according to the contents of the i^{th} row of my.data:

   i. If the i^{th} row of my.data do not contain any negative values, the line should contain the sentence: "The mean of row number i is", where i should be the row number, and then the mean of the i^{th} row of my.data.
   ii. If the i^{th} row contains negative values, but no more than 3 lines with negative numbers have been encountered, the line should contain the sentence "<Row i contains negative values>", where again i is the row number.
   iii. If the i^{th} row contains negative values, and at least 3 lines with negative numbers have been encountered, the line should contain the sentence "Too many negative values".

   If the line "Too many negative values" have been written, the loop should stop writing out lines immediately, otherwise it should stop when the last line of my.data have been investigated an a line has been written.

Exercise 3.1b:

b) Construct a dataset in the following way:

```
set.seed(1786)
data.exercise.3.1<-exp(matrix(rnorm(2000),nrow=100))
index1.temp<-sample(1:100,10)
index2.temp<-sample(1:20,10)
for(i in 1:10){
 data.exercise.3.1[index1.temp[i],index2.temp[i]]<--1
}
```

Assign the dataset data.exercise.3.1 to my.data and check that your algorithm works. You should write out 14 lines.

Exercise 3.2:

a) The function Sys.time() returns the current time. In the slide where we compared run times of elementwise squaring a matrix y containing elements generated with the rnorm() function, and doing it using the object-oriented form of R. We looked at a huge matrix with 1.000.000 elements. What is the effect for small matrices? Redo the runtime exercise with a matrix of dimension 10x10 instead of 1000x1000, and calculate the runtime increase factor. Comment.

b) Now, let k be the number of rows in the quadratic matrix y. calculate runtime increase factors as above for k=10, 20, 50, 100, 200, 500, 800 and 1000, and plot the runtime increase factors as a function of the number of elements in y.

Exercise 3.3:

Create a data frame with 10 rows and 100.000 columns, and fill it with random numbers generated with rnorm(). We want to calculate the mean of all 100.000 columns, and place it in a vector. Calculate the runtime factor from looping over the 100.000 columns and using the mean function, in contrast to using sapply().

Exercise 4.1:

a) Explain what happens in the following matrix construction:

```
matrix(1:6,nrow=3,ncol=3)
```

b) What happens if we do not give any input vector at all when we define a matrix?

Exercise 4.2:

a) Explain what goes on here:

```
x<-matrix(1:12,4)
x
x[cbind(c(1,3,2),c(3,3,2))]
```

b) Calculate the following and explain the difference with (a)

```
x[c(1,3,2),c(3,3,2)]
```

Exercise 4.3

Create matrices row, column and A in the following way:

```
row<-matrix(rep(1:100,100),nrow=100)
column<-matrix(rep(1:100,100),nrow=100,byrow=T)
A<-3*column^3/(1+row*column)
```

a) Find an easy way to calculate the sum

$$\sum_{i=1}^{100}\sum_{j=1}^{100}\frac{3i^3}{1+ij}$$

b) Next, find an easy way to calculate

$$\sum_{i=1}^{100}\sum_{j=1}^{i}\frac{3i^3}{1+ij}$$