1. **Simple start of R** *(5 points)*
   Download and install the R-Software. http://cran.r-project.org → Download CRAN → search a city near you. Choose your system (Windows / Mac / Linux). Helpful introductions to R can be found Rintro.pdf or R-intro.pdf
   (a) Create a vector t "t<-seq(-2*pi,2*pi,by=0.01)"
   Plot several functions in one window $(\sin(t), \cos(t), \exp(\frac{t}{5}), (\frac{t}{5})^2, (\frac{t}{5})^3)$. Try some of the plot arguments: Set ylim, label the axes, set a different colour for each function, vary the line width. Save the plot as a figure.
   For help try "?plot" or "?plot.default"
   (b) Set up a vector of length 20 and create a vector b with a linear relationship to a (e.g. $a = 3b + 7$). Calculate the correlation("cor(a,b)").
   (c) Set up two random vectors a,b of length 20 and calculate the correlation. Repeat this procedure several times to get a feeling for the correlation coefficient. Than vary the length of vector a and b (vary the sample number) and discuss how the correlation coefficient changes (e.g. 10,50,100,1000).
   (d) Repeat the experiment from the previous task 100 times by using a loop. Create before the loop an empty vector ("cor.val<-vector()") and save the correlation of a and b in this vector (e.g. "cor.val[i]<-cor(a,b)") for each realisation. Compute the mean value and plot the histogram of cor.val. What happens with the histogram when the length of a and b is varied (e.g. 10,50,100)? Save two different histograms as a figure and explain the difference between them.

```
# Important R-commands
rnorm(N) # create vector with N normal distribution random numbers
cor(a,b) # calculates the correlation coefficient
hist(a)  # histogram of vector a
mean(a)  # mean value of vector a
```

2. **Short programming questions** *(3 points)*

   Write down the output for the following R-commands:

   a) `0:10`
   b) `a<-c(0,5,3,4); mean(a)`
   c) `max(a)-min(a)`
   d) `paste("The mean value of a is",mean(a),"for sure",sep="_")`
   e) `a*2+c(1,1,1,0)`
   f) `my.fun<-function(n){return(n*n+1)}`
      `my.fun(10)-my.fun(1)`

3. **Lorenz equations** *(3 points)*

Consider the Lorenz equations (which were derived from the Rayleigh-Bernard system)

$$\dot{x} = \sigma(y-x) \tag{1}$$
$$\dot{y} = rx - xz - y \tag{2}$$
$$\dot{z} = xy - bz \tag{3}$$

with $\sigma, r, b > 0$. σ is the Prandtl number. Furthermore, Rayleigh number $R_a \sim \Delta T$, critical Rayleigh number R_c, and $r = R_a/R_c$. Below is the numerical solution of the Lorenz Problem. Display the function in the phase-space and time-dependence for

(a) an initial value $x_0 \in [0,1]$, and a parameter value $r \in [0,1]$
(b) the parameter r using $r = 13, 14$ and $r \in [20, 30]$)

```
# parameters
r=24
s=10
b=8/3
dt=0.01   # time step

# initial conditions:
x=0.1
y=0.1
z=0.1

# provide the solution vector
vx<-c(0)
vy<-c(0)
vz<-c(0)

# time stepping:
for(i in 1:10000){
x1=x+s*(y-x)*dt
y1=y+(r*x-y-x*z)*dt
z1=z+(x*y-b*z)*dt
vx[i]=x1
vy[i]=y1
vz[i]=z1
x=x1
y=y1
z=z1
}
plot(vx,vy,type="l",xlab="x",ylab="y",main="LORENZ ATTRACTOR")
```