

```
##### '0 'R starten, R
        beenden''
```

```
# Wenn man R startet, beginnt man einen Session. Man beendet einen Session mit
```

```
q()
```

```
# (Hier die Frage 'Save workspace image' mit 'cancel' beantworten).
```

```
##### '1. Objekte in R'
```

```
# Ein neues Objekt wird mit '=' (assign) oder '<-' erzeugt
```

```
# Es gibt numerische Objekte
```

```
x = 3
```

```
y <- 4
```

```
ls()
```

```
# Und auch Schriftzeichen-Objekte in ""
```

```
z = "etwas"
```

```
# Wenn Sie den Inhalt des Objektes sehen wollen, einfach
```

```
# den Objekt-Namen eingeben
```

```
x
```

```
# 3
```

```
# Neue Objekte erzeugen mit dem selben Inhalt
```

```
x = y = z = 4
```

```
# Objekte werden in R gleich überschrieben ohne Warnmeldung
```

```
y = 4
```

```
y
```

```
# 4
```

```
y = "Phonetik"
```

```
# y
```

```
# "Phonetik"
```

```
# Was ist das für ein Objekt?
```

```
class(x)
```

```
class(y)
```

```
##### '2. Vektoren und
        Indizierung'
```

```
# 2.1 'Die c() Funktion'
```

```
# Numerischer Vektor 'x' mit 6 Elementen
```

```
x = c(3, 4, 6, 89.3, 0, -10)
```

```
# Schriftzeichen-Vektor mit 4 Elementen
```

```
labs = c("E", "A", "E", "i:")
```

```
# '2.2 Indizierung'
```

```
x[2]
```

```
# Element 2 von x
```

```
x[2:4]
```

```
# Elemente 2 bis 4
```

```
x[c(1,4)]
```

```
# Elemente 1 und 4
```

```
x[-2]
```

```
# Alle Elemente außer Element 2
```

```
x[2:4] = 0
```

```
# Elemente 2 bis 4 auf 0 setzen
```

```

# Alle Elemente von 'x' ohne den 2en und 5en?

# Elemente von 'x' in der anderen Reihenfolge (also von 6 bis 1)?

#####'3. Arithmetik: (*, /, +,
-)'

a = c(10, 4, 20)
a * 10

b = c(5, 2, 7)
a + b
# 15 6 27

sum(a)                # Summe aller Elemente
sqrt(a)               # Wurzel pro Element
a^3                   # Jedes Element hoch 3
log(a)                # Logarithmus
exp(a)                # Exponential

#####'4. Vektoren
manipulieren'
# length(), seq(), rep(), unique(), table()

# Vektorenlänge (wieviele Elemente in einem Vektor?): length()
y = c("i", "i", "a", "a", "E", "E", "E", "E", "U")
length(y)

# Wie könnte man das letzte Element von 'y' listen?
# Das vorletzte? Das Vorletzte und Letzte zusammen?

# Intervalle: 'seq()'
seq(10, 20, length=5)    # 5 Intervalle zwischen 10 und 20
seq(10, 20, by=1.5)      # In Intervallen von 1.5

# Wiederholung: 'rep()'
a = c(10, 4, 20)
rep(a, 4)
rep(a, each=2)

# 'unique()'
y = c("i", "i", "a", "a", "E", "E", "E", "E", "U")
unique(y)
# "i" "a" "E" "U"

# Tabelle, 'table()'
table(y)
y
# a E i U
# 2 4 2 1

#####'5. Data-Frames und
Matrizen'
'5.1 Eigenschaften und Indizierung'

# Einlesen
pfadu = "http://www.phonetik.uni-muenchen.de/~jmh/lehre/Rdf"
ai = read.table(file.path(pfadu, "ai.txt"))

```

```
# Eigenschaften überprüfen
ai

# Was ist das für ein Objekt?
class(ai)

# die ersten paar Reihen (oder Beobachtungen)
head(ai)

# Reihen und Spaltenanzahl
nrow(ai)
ncol(ai)
dim(ai)

# Die Spaltennamen oder Variablen
names(ai)
# das gleiche
colnames(ai)

# Zugriff auf Elemente
# 'ai[m,]' = Reihe m
# 'ai[,m]' = Spalte m

# Reihe (Beobachtung) 2
ai[2,]

# Beobachtungen 2 bis 10
ai[2:10,]

# Beobachtungen 2, 3, und 8
ai[c(2, 3, 8),]

# oder
vec = c(2, 3, 8)
ai[vec,]

# Spalte 2 (Variable 2)
ai[,2]

# Für Data-Frames kann man mit '$Namen' auf die Spalten zugreifen
names(ai)
# "F1"      "Kiefer" "Lippe"

# das gleiche wie ai[,2]
ai$Kiefer

# Beobachtungen 2 bis 10 von Variable 2
ai[2:10,2]

# oder
ai$Kiefer[2:10]

# Reihen 2, 5, 8 von Variablen 2 und 3?

# Reihen 12 bis 18 von Variablen 1 und 3?

# Ein Minuszeichen in '[-n, ]' oder '[,-n]': alle außer n
# Alle Spalten außer Spalte 1
ai[,2:3]
```

```

# oder
ai[,-1]

# Anzahl der Beobachtungen
n = nrow(ai)

# Letzte Beobachtung (indem 'n' verwendet wird)?

# Vorletzte Beobachtung (indem 'n' verwendet wird)?

# Letzte 3 Beobachtungen (indem 'n' verwendet wird)?

'5.2 Anwendung arithmetischer Funktionen'
# (Angenommen, dass die Variablen vom Data-Frame
# numerisch sind
class(ai[,1])
# oder
class(ai$F1)
# "integer"
class(ai[,2])
# "numeric"
class(ai[,3])
# "numeric"

# 20 von allen Werten abziehen
ai - 20

# Variable 1 Mal 5
ai[,1] * 5
# oder
ai$F1 * 5

neu = ai - 20

ai - neu
# Der obige Befehl funktioniert, wenn die Dimensionen der
# Data-Frames/Matrizen gleich sind
dim(ai)
dim(neu)
# daher auch
ai[1:3,2:3] / neu[10:12,1:2]
# da
dim(ai[1:3,2:3])
# und
dim(neu[10:12,1:2])
# gleich sind.

# Dies funktioniert nicht:
ai[1:3,2:3] / neu[10:12,]
# / only defined for equally-sized data frames

##### Üben und auswendig
        lernen

# 'Allgemein'
# ls()
# class()
# Welche Objekte sind im Verzeichnis?
# was ist das für ein Objekt?

```

```
# 'Vektoren'
# c()                                # Elemente in einen Vektor verbinden

# 'Arithmetik'
# x * 20
# x / 20
# x + 20
# x - 20

# 'Vektoren-Manipulation'
# length()                          # Wieviele Elemente in einem Vektor
# seq()                             # Intervalle erzeugen
# rep()                             # Elemente wiederholen
# unique()                          # Type/Token listen
# table()                           # Tabellieren

# 'Data-Frames'
# head()                            # Die ersten paar Reihen
# nrow()                            # Anzahl der Reihen
# ncol()                            # Anzahl der Spalten
# dim()                             # Dimensionen (Reihen x Spalten)
# names()                           # Variablen-Namen
# levels()                          # Stufen einer Variable

# 'Indizierung'
# Vektoren
# x[2]                              # Element 2 von x
# x[2:4]                            # Elemente 2 bis 4
# x[c(1,4)]                         # Elemente 1 und 4

# Data-Frames
# x[a,b]                            # Beobachtungen a von Variable b
```