```r
# '1. Vektoren'

# 1.1 'Die c() Funktion'

# Numerisch
vec  = 20
vec

# Schriftzeichen
vec = "ips"

# 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:")

# Elemente 2 bis 4 von y mit 0 überschreiben
y[2:4] = 0

# '1.2 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)?

'1.3 Arithmetische Manipulationen von Vektoren (*, /, +, -)'

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
range(a)            # Bereich
max(a)              # Maximum
mean(a)             # Mittelwert

# '1.4 Einige nützliche Funktionen, um Vektoren zu manipulieren'
length()
y = c("i", "i", "a", "a", "E", "E", "E", "E", "U")
length(y)
```

```r
# 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

# Wiederholdung: '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

# 'paste()'
paste(y, "V", sep=".")
#  "i.V" "i.V" "a.V" "a.V" "E.V" "E.V" "E.V" "E.V" "U.V"

# String  manipulation:  'nchar()', 'substring()'
cities = c("London", "Paris", "Hamburg", "Verona", "New York")

# Anzahl der Schriftzeichen pro Element
nchar(cities)
 6 5 7 6 8

# Die ersten 3 Schriftzeichen pro Element
substring(cities, 1, 3)

'2. Data-Frames und Matrizen'
'2.1 Eigeschaften und Indizierung'

# Einlesen
ai = read.table(file.path(pfadu, "ai.txt"))

# Eigenschaften überprüfen
ai

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

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

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

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

# Die Namen der Reihen (Beobachtungen)
rownames(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?
ai[c(2, 5, 8), 2:3]

# Reihen 12 bis 18 von Variablen 1 und 3?
ai[12:18, c(1, 3)]
```

```r
# 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)?
ai[n,]

# Vorletzte Beobachtung (indem 'n' verwendet wird)?
ai[n-1,]

# Letzte 3 Beobachtungen (indem 'n' verwendet wird)?
ai[(n-2):n,]

'2.2 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
```

```r
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

'3. Fragen'
# Für den vorhandenen Data-Frame 'ai' schreiben Sie R-Befehle für:

# F1 von Beobachtung 10
ai[10,1]
# oder
ai$F1[10]

# F1, Beobachtungen 12 und 14
ai$F1[c(12, 14)]

# F1, Beobachtungen 1-5, 16, und 21-25
ai$F1[c(1:5, 16, 21:25)]

# Alle Werte der Lippe, außer dem zweiten
ai$Lippe[-2]

# Alle Werte des Kiefers, außer den ersten 10
ai$Kiefer[-(1:10)]

# Beobachtungen 18, 20, 24
ai[c(18, 20, 24),]

# F1 und Kiefer, Beobachtungen 20-25
ai[20:25, 1:2]

# F1 und Lippe, Beobachtungen 9, 12, 18
ai[c(9, 12, 18), c(1, 3)]

# 'Lippe + Kiefer' aller Beobachtungen
# (also einen Wert pro Beobachtung)
ai$Lippe + ai$Kiefer

# F1 aller Beobachtungen minus 100
ai - 100

# Der F1-Mittelwerte aller Beobachtungen
mean(ai$F1)

# F1 aller Beobachtungen minus den F1-Mittelwert
ai$F1 - mean(ai$F1)

# Lesen Sie den Data-Frame ein 'asp'
asp = read.table(file.path(pfadu, "asp.txt"))
```

```r
# Wieviele Beobachtungen gibt es? Wieviele Variablen?
# Beobachtungen und Variablen
dim(asp)
# Beobachtungen
nrow(asp)
# Variablen
ncol(asp)

# Verwenden Sie die 'head()' Funktion, um die ersten paar Reihen zu sehen
head(asp)

# Identifizieren Sie die Variablen-Namen
# Diese sind: 'd': die Aspirationsdauer,
# 'Wort': das Wort, in dem die Aspiration vorkam
# 'Vpn': die Versuchsperson (Sprecher)
# 'Kons': der Konsonant, in dem die Aspiration vorkam
# 'Bet': ob die Silbe, in der die Aspiration vorkam, betont war, oder nicht.
names(asp)

# Bestätigen Sie, dass die Aspirationsdauer numerisch ist
class(asp$d)

# Die anderen nicht-numerischen Variablen sind Faktoren.
# Bestätigen Sie mit der 'class()' Funktion, dass die Variable 'Wort' ein Faktor ist
class(asp$Wort)

# Berechnen Sie den Dauer-Mittelwert aller Beobachtungen
mean(asp$d)

# Berechnen Sie den Dauer-Mittelwert von Beobachtungen 1 bis 1000
mean(asp$d[1:1000])

# Verwenden Sie die 'table()' Funktion, um die Konsonanten zu tabellieren
table(asp$Kons)

# Verwenden Sie die 'table()' Funktion, um die Wörter zu tabellieren
table(asp$Wort)

# Wie bekommen Sie das erste Schriftzeichen aller Wörter?
substring(asp$Wort, 1, 1)

# Tabellieren Sie das erste Schriftzeichen aller Wörter
table(substring(asp$Wort, 1, 1))
```