Datentypen und Datensätze

[Termin 2]

Miguel Alvarez

kamapu.net

22. Oktober 2025

R als Taschenrechner

- Mathematische Operatoren
- Klammern

```
5+6*11
```

```
[1] 71
```

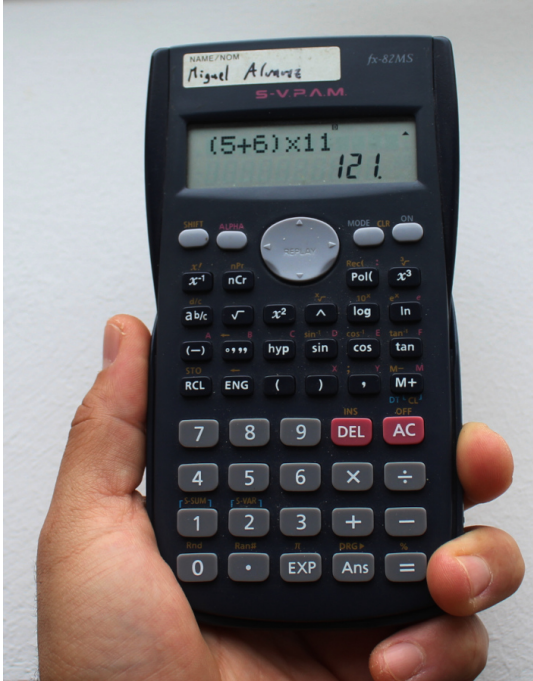
```
(5+6)*11
```

```
[1] 121
```

- Zuweisung <-

```
M <- 5
M + 2
```

```
[1] 7
```



Logische Operatoren

```
==
!=
>
>=
<
<=
&
|
%in%
!
any()
all()
```

```
10 > 15
```

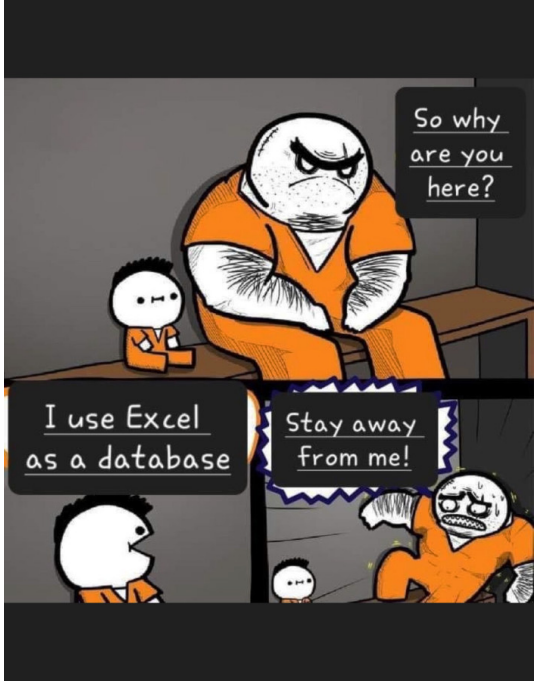
```
[1] FALSE
```

```
10 < 15
```

```
[1] TRUE
```

Vektoren

Welche Datentypen kennst du?



Vektoren

Vektoren

Der Vektor ist die grundlegende Datenstruktur in **R**

- Länge `length()`
- Typ `class()`
- Evtl. Namen `names()`

```
c(1:10)
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
rep(5, times = 10)
```

```
 [1] 5 5 5 5 5 5 5 5 5 5
```

```
LETTERS[1:5]
```

```
[1] "A" "B" "C" "D" "E"
```

Vektoren

Indexieren

- Eckige Klammern
- Index
  - `integer`
  - `logical` (Bedingung)
  - `character` (Namen)

```
# Mit integer
letters[15]
```

```
[1] "o"
```

```
# Mit logischen Werten
letters[!letters %in% c("a", "b", "c")]
```

```
 [1] "d" "e" "f" "g" "h" "i" "j" "k" "l"
[10] "m" "n" "o" "p" "q" "r" "s" "t" "u"
[19] "v" "w" "x" "y" "z"
```

```
# Mit Namen
names(letters) <- letters
letters["m"]
```

```
  m
"m"
```

Vektoren

Datentypen

- integer
- numeric
- logical
- factor
- character

```
A <- c(1:10)
is.numeric(A)
```

```
[1] TRUE
```

```
B <- as.character(A)
B
```

```
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"
 [8] "8"  "9"  "10"
```

```
is.numeric(B)
```

```
[1] FALSE
```

Vektoren

Sonderklassen

- NA
- NaN
- NULL
- Inf
- -Inf

```
5/0
```

```
[1] Inf
```

```
log(0)
```

```
[1] -Inf
```

```
sqrt(-1)
```

```
[1] NaN
```

Objekte

Funktion

`foo(par1 = arg1, ..., parn = argn)`

Funktionen und Argumente (Parameter) werden dokumentiert.
Achte auf Standardeinstellungen (default values).

```
A <- c(1, NA, 3, 5)
mean(A)
```

```
[1] NA
```

```
mean(A, na.rm = TRUE)
```

```
[1] 3
```

Objekte

Matrix

- Typ von Inhalt (`mode()`).
- Zwei Dimensionen.

```
M <- matrix(1:20, nrow = 4)
M
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    5    9   13   17
[2,]    2    6   10   14   18
[3,]    3    7   11   15   19
[4,]    4    8   12   16   20
```

```
class(M)
```

```
[1] "matrix" "array"
```

```
mode(M)
```

```
[1] "numeric"
```

```
length(M)
```

```
[1] 20
```

```
dim(M)
```

```
[1] 4 5
```

Objekte

Arrays

- Mehrdimensionale Matrizen.

```
iris3[1:2, , ]
```

```
, , Setosa

     Sepal L. Sepal W. Petal L. Petal W.
[1,]      5.1      3.5      1.4      0.2
[2,]      4.9      3.0      1.4      0.2

, , Versicolor

     Sepal L. Sepal W. Petal L. Petal W.
[1,]      7.0      3.2      4.7      1.4
[2,]      6.4      3.2      4.5      1.5

, , Virginica

     Sepal L. Sepal W. Petal L. Petal W.
[1,]      6.3      3.3      6.0      2.5
[2,]      5.8      2.7      5.1      1.9
```

Objekte

Liste

Liste (Sammlung) von Objekten, inklusive Listen.
Achte, dass `data.frame` eine spezielle Form von `list` ist.

```
MeineListe <- list(
    A = 1:10,
    B = matrix(1:10, nrow = 2),
    C = "Dies ist eine Liste")
MeineListe
```

```
$A
 [1]  1  2  3  4  5  6  7  8  9 10

$B
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10

$C
[1] "Dies ist eine Liste"
```

Objekte

Datensatz

Spaltenorientierte Tabelle (`data.frame`)

```
head(iris)
```

```
  Sepal.Length Sepal.Width Petal.Length
1          5.1         3.5          1.4
2          4.9         3.0          1.4
3          4.7         3.2          1.3
4          4.6         3.1          1.5
5          5.0         3.6          1.4
6          5.4         3.9          1.7
  Petal.Width Species
1         0.2  setosa
2         0.2  setosa
3         0.2  setosa
4         0.2  setosa
5         0.2  setosa
6         0.4  setosa
```

Vielen Dank!

```
library(fortunes)
fortune(283)
```

```
The good way to do it is to include the following comment at the beginning:
# This is a holy Script, please edit it not
   -- Kenn Konstabel (on "... how to protect R Script files from inadvertent editing by
      users.")
      R-help (April 2011)
```