

Sommersemester 2022

Jens Klenke

R Propädeutikum

Lösung Übungsaufgaben 2

1 Übungsaufgaben zur Logik.

1.1 Überprüfen Sie in R ob die folgenden Ausdrücke TRUE oder FALSE sind?

- $5 \geq 5$

```
5 >= 5
```

```
## [1] TRUE
```

- $5 > 5$

```
5 > 5
```

```
## [1] FALSE
```

- $T = 5$

```
T == 5
```

```
## [1] FALSE
```

- $T \wedge F \vee F \wedge T$

```
T&F | F&T
```

```
## [1] FALSE
```

- $F \wedge F \wedge F \vee T$

```
F&F&F | T
```

```
## [1] TRUE
```

- $(\neg(5 > 3) \vee A = B)$

```
(!(5 > 3) | "A" == "B")
```

```
## [1] FALSE
```

- $\neg(((T > F) > T) \wedge \neg T)$

```
!(((T > F) > T) & !T)
```

```
## [1] TRUE
```

1.2 Es sei $z \leftarrow c(1, 2, NA, 4)$. Überprüfen Sie die folgenden Aussagen mittels einer Logikabfrage in R.

- Die Länge des Vektors z ist ungleich 2.

```
z <- c(1, 2, NA, 4)
```

```
length(z) != 2
```

```
## [1] TRUE
```

- Die Länge der logischen Überprüfungen, ob die einzelnen Elemente gleich 2 sind, ist 4.

```
length(z == 2)
```

```
## [1] 4
```

- Der Vektor z hat die Klasse 'numeric'.

```
is.numeric(z)
```

```
## [1] TRUE
```

- Einige Elemente des Vektors z sind 'NA'.

```
is.na(z)
```

```
## [1] FALSE FALSE TRUE FALSE
```

- Das zweite Element des Vektors z ist 'numeric'.

```
is.numeric(z[2])
```

```
## [1] TRUE
```

- Das Minimum und das Maximum sind ungleich.

```
(min(z) != max(z))
```

```
## [1] NA
```

```
(min(z, na.rm = TRUE) != max(z, na.rm = TRUE))
```

```
## [1] TRUE
```

1.3 Es sei $M \leftarrow \text{matrix}(1:9, \text{ncol} = 3)$. Was ergeben folgende Ausdrücke:

```
M <- matrix(1:9, ncol = 3)
```

```
sum(M[, 1]) == 6
```

```
## [1] TRUE
```

```
max(M[, 2]) <= 5
```

```
## [1] FALSE
```

```
M[2, 2] != 4 & M[2, 2] > 6
```

```
## [1] FALSE
```

2 Übungsaufgaben zu Dataframes

2.1 Verschaffen Sie sich einen Überblick über den Datensatz `mtcars` (dieser ist in base R bereits geladen). Aus wie vielen Variablen besteht der Datensatz? Welche Klasse haben die einzelnen Variablen?

```
str(mtcars)
```

```
## 'data.frame': 32 obs. of 11 variables:
## $ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
## $ cyl : num 6 6 4 6 8 6 8 4 4 6 ...
## $ disp: num 160 160 108 258 360 ...
## $ hp : num 110 110 93 110 175 105 245 62 95 123 ...
## $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
## $ wt : num 2.62 2.88 2.32 3.21 3.44 ...
## $ qsec: num 16.5 17 18.6 19.4 17 ...
## $ vs : num 0 0 1 1 0 1 0 1 1 1 ...
## $ am : num 1 1 1 0 0 0 0 0 0 0 ...
## $ gear: num 4 4 4 3 3 3 3 4 4 4 ...
## $ carb: num 4 4 1 1 2 1 4 2 2 4 ...
```

Der Datensatz besteht aus elf Variablen, die alle der Klasse `numeric` angehören.

2.2 Lassen Sie sich folgende Subsets von `mtcars` ausgeben:

- nur die Variable `mpg`

```
mtcars$mpg
```

```
## [1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3
## [14] 15.2 10.4 10.4 14.7 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3
## [27] 26.0 30.4 15.8 19.7 15.0 21.4
```

- nur die ersten drei Zeilen

```
mtcars[1:3, ]
```

```
##           mpg cyl disp  hp drat   wt  qsec vs am gear carb
## Mazda RX4  21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710  22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
```

- nur die ersten drei Variablen

```
mtcars[, 1:3]
```

```
##           mpg cyl  disp
## Mazda RX4      21.0   6 160.0
## Mazda RX4 Wag  21.0   6 160.0
## Datsun 710      22.8   4 108.0
## Hornet 4 Drive  21.4   6 258.0
## Hornet Sportabout 18.7   8 360.0
## Valiant         18.1   6 225.0
## Duster 360      14.3   8 360.0
## Merc 240D       24.4   4 146.7
## Merc 230        22.8   4 140.8
## Merc 280        19.2   6 167.6
## Merc 280C       17.8   6 167.6
## Merc 450SE      16.4   8 275.8
## Merc 450SL      17.3   8 275.8
## Merc 450SLC     15.2   8 275.8
## Cadillac Fleetwood 10.4   8 472.0
## Lincoln Continental 10.4   8 460.0
## Chrysler Imperial 14.7   8 440.0
## Fiat 128        32.4   4  78.7
## Honda Civic     30.4   4  75.7
## Toyota Corolla  33.9   4  71.1
## Toyota Corona   21.5   4 120.1
## Dodge Challenger 15.5   8 318.0
## AMC Javelin     15.2   8 304.0
## Camaro Z28      13.3   8 350.0
## Pontiac Firebird 19.2   8 400.0
## Fiat X1-9       27.3   4  79.0
## Porsche 914-2   26.0   4 120.3
## Lotus Europa    30.4   4  95.1
## Ford Pantera L  15.8   8 351.0
## Ferrari Dino    19.7   6 145.0
## Maserati Bora   15.0   8 301.0
## Volvo 142E     21.4   4 121.0
```

- nur die ersten beiden Beobachtungen der Variablen cyl und hp

```
mtcars[1:2, c(2, 4)]
```

```
##           cyl  hp
## Mazda RX4      6 110
## Mazda RX4 Wag  6 110
```

- alle Beobachtungen deren Ausprägung der Variable `hp` größer ist als 200

```
mtcars[mtcars$hp > 200,]
```

```
##           mpg cyl disp  hp drat   wt  qsec vs am gear
## Duster 360   14.3  8  360 245 3.21 3.570 15.84 0  0    3
## Cadillac Fleetwood 10.4  8  472 205 2.93 5.250 17.98 0  0    3
## Lincoln Continental 10.4  8  460 215 3.00 5.424 17.82 0  0    3
## Chrysler Imperial 14.7  8  440 230 3.23 5.345 17.42 0  0    3
## Camaro Z28    13.3  8  350 245 3.73 3.840 15.41 0  0    3
## Ford Pantera L 15.8  8  351 264 4.22 3.170 14.50 0  1    5
## Maserati Bora 15.0  8  301 335 3.54 3.570 14.60 0  1    5
##           carb
## Duster 360      4
## Cadillac Fleetwood 4
## Lincoln Continental 4
## Chrysler Imperial 4
## Camaro Z28      4
## Ford Pantera L  4
## Maserati Bora    8
```

2.3 Erstellen Sie einen Dataframe `persons` mit den Variablen `Name` (character), `Height` (cm, numeric) und `Weight` (kg, numeric) von 5 fiktiven Personen.

```
persons <- data.frame(Name = c("Jerry", "Beth", "Summer", "Morty", "Rick"),
                      Height = c(180, 170, 170, 155, 175),
                      Weight = c(75, 73, 55, 52, 67))
```

- Lassen Sie sich das Körpergewicht der 3. Person anzeigen.

```
persons[3, ]$Weight
```

```
## [1] 55
```

- Lassen Sie sich nun die Körpergröße aller Personen anzeigen.

```
persons$Height
```

```
## [1] 180 170 170 155 175
```

- Fügen Sie die Variable “Augenfarbe” hinzu. Die Ausprägungen sollten vom Typ `character` sein. Schauen Sie sich den veränderten dataframe an.

```
persons$Eyecolor <- c("black", "blue", "black", "green", "black")
persons
```

```
##      Name Height Weight Eyecolor
## 1  Jerry    180     75    black
## 2   Beth    170     73     blue
## 3 Summer    170     55    black
## 4  Morty    155     52    green
## 5   Rick    175     67    black
```

3 Übungsaufgaben zu bedingte Anweisungen

- 3.1 Schreiben Sie Code, der die Wurzel (`sqrt()`) eines Vektors `x` der Länge 1 berechnet, wenn der Wert in `x` nicht negativ ist.**

```
x <- 2 # - 2, testen!
if(x >= 0) sqrt(x)
```

```
## [1] 1.414214
```

- 3.2 Erstellen Sie Code, welcher die Wurzel der Elemente eines Vektors `x` berechnet, wenn alle Werte in `x` nicht negativ sind.**

Hinweis: Nutzen Sie eine Funktion wie `min()` oder `sum()`.

```
x <- c(4, 16, 64)
if(min(x) >= 0) sqrt(x)
```

```
## [1] 2 4 8
```

```
# ODER
if(!sum(x < 0)) sqrt(x)
```

```
## [1] 2 4 8
```

3.3 Schreiben Sie Code, der die Struktur (str()) eines Objekts df wiedergibt, sofern df zur Klasse data.frame gehört. Andernfalls soll die Länge des Objekts wiedergegeben werden.

```
df <- data.frame(A = 1:3)
if(class(df) == "data.frame") {
  str(df)
} else {
  length(df)
}
```

```
## 'data.frame':  3 obs. of  1 variable:
## $ A: int  1 2 3
```

4 Übungsaufgaben zu Schleifen

4.1 Schreiben Sie eine Schleife, welche die Zahlen von 1 bis 15 aufaddiert.

```
x <- 0
for(i in 1:15) {
  x <- x + i
}
x
```

```
## [1] 120
```


4.2 Erstellen Sie eine Matrix M von folgender Gestalt:

$$M = \begin{pmatrix} 1 & 4 & 7 & 10 & 13 \\ 2 & 5 & 8 & 11 & 14 \\ 3 & 6 & 9 & 12 & 15 \end{pmatrix}$$

- Schreiben Sie eine Schleife, welche für jede Spalte die Spaltensumme berechnet und ausgibt.

```
M <- matrix(1:15, ncol = 5)
```

```
# Schleife
```

```
for(i in 1:ncol(M)) {  
  print(sum(M[,i]))  
}
```

```
## [1] 6
```

```
## [1] 15
```

```
## [1] 24
```

```
## [1] 33
```

```
## [1] 42
```

4.3 Mit `rnorm(1)` ziehen Sie eine Zufallszahl aus der Standardnormalverteilung (in der Konsole ausprobieren!). Schreiben Sie eine Schleife, welche solange ausgeführt wird, bis ein Wert gezogen wird, der größer als 1 ist.

Geben Sie in jedem Durchlauf die gezogene Zahl mit `cat(x, "\n")` aus. (Hinweis: `\n` steht für einen Zeilenumbruch)

```
set.seed(420)
```

```
x <- rnorm(1)  
while(x <= 1) {  
  cat(x, "\n")  
  x <- rnorm(1)  
}
```

```
## 0.2677109
```

```
## -0.9367075
```

```
## 0.5962061
```

```
## -0.3120436
## 0.3979791
## -0.5147962
## -0.5518658
## -0.4605003
## -2.948791
## -0.7671461
## 0.9385212
## -0.2870975
```

5 Übungsaufgaben zu Funktionen

5.1 Die Dichte der Standardnormalverteilung lautet

$$\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

. Schreiben Sie eine Funktion `stdnv`, welche die Dichte von `x` berechnet und zurückgibt.

- *Hinweis:* ‘`?exp`’, ‘`?pi`’
- *Hinweis:* Wenn die Funktion korrekt ist, sollten ‘`stdnv(x)`’ und ‘`dnorm(x)`’ die gleichen Ergebnisse liefern.

```
stdnv <- function(x){
  return(1/sqrt(2*pi) * exp(-x^2/2))
}

stdnv(0.1337)
```

```
## [1] 0.3953925
```

```
dnorm(0.1337)
```

```
## [1] 0.3953925
```

5.2 Schreiben Sie eine Funktion, welche die Argumente `z` sowie `opt` erwartet. Im Funktionskörper soll mit einer If-Anweisung gesteuert werden, welche Operation auf `z` ausgeführt werden soll:

WENN `opt` gleich “`add`” ist, *DANN* addiere die Elemente von `z`, *WENN* `opt` gleich “`mult`” ist, dann multipliziere die Elemente von `z`, andernfalls führe keine Operation aus.

Am Ende soll die Funktion das jeweilige Ergebnis wiedergeben.

```
myFun <- function(z, opt) {  
  if (opt == "add") {  
    result <- sum(z)  
  } else if (opt == "mult") {  
    result <- prod(z)  
  }  
  return(result)  
}
```

```
x <- 1:5  
myFun(z = x, opt = "mult")
```

```
## [1] 120
```

```
myFun(z = x, opt = "add")
```

```
## [1] 15
```

5.3 Schreiben Sie eine Funktion, die den MSE (mean squared error) von zwei Vektoren y und \hat{y} (die Argumente) berechnet. Der MSE ist definiert als

$$\frac{1}{n} \sum_{i=1}^n (\hat{Y}_i - Y_i)^2$$

.

Testen Sie Ihre Funktion anhand der beiden Vektoren $y = 2, 4, 2, 5, 7$ und $\hat{y} = 2.3, 3.5, 2.1, 5.5, 7.6$ (das Ergebnis sollte 0.192 lauten).

```
y <- c(2, 4, 2, 5, 7)  
yhat <- c(2.3, 3.5, 2.1, 5.5, 7.6)  
mse <- function(y, yhat) {  
  1/length(y) * (sum((yhat - y)^2))  
}  
mse(y = y, yhat = yhat)
```

```
## [1] 0.192
```