Einführung in R: Teil 1

Ekaterina Akimova-Höpner

2024-09-07

Einführung

1. Einfache mathematische Operationen

```
4 + 5
```

```
## [1] 9
```

```
sqrt(36)
```

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

```
4 + 2 * 2
```

```
## [1] 8
```

```
log10(100)
```

```
## [1] 2
```

```
log(100, base = 4)
```

```
## [1] 3.321928
```

```
50 + pi
```

```
## [1] 53.14159
```

2. Variablen

2.1 Variablen zuweisen

```
x <- 5
a <- 2 + 9
b <- (x + a) * 2
x <- b - 2

alter <- 30
name <- "Max"


mein_vergleich <- 6 > 8
neue_var <- (x < 10 && x > 5) | b == 65

heights <- c(163, 170, NA, 167, NA)
```

Exercise 0:

Was ist der aktuelle Wert von `b`?

2.2 Datentypen überprüfen

```
class(alter)
```

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

```
class(name)
```

```
## [1] "character"
```

```
class(mein_vergleich)
```

```
## [1] "logical"
```

3. Einfache Funktionen

3.1 Eingebaute Funktionen

```
heights <- c(163, 170, NA, 167)
mean(heights)
```

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

```
which(is.na(heights))
```

```
## [1] 3
```

```r
?mean

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

```
## [1] 166.6667
```

Exercise 1.1

a) Was ist die Summe der ersten 100 positiven ganzen Zahlen? Die Formel für die Summe der ganzen Zahlen von 1 bis n lautet $\frac{n(n+1)}{2}$.

```r
n <- 100

n*(n+1)/2
```

```
## [1] 5050
```

b) Was ist die Summe der ersten 1000 positiven ganzen Zahlen?

Exercise 1.2

Probiere `sum(seq(1,n))` aus. Was macht `seq()`? Was macht `sum()`?

3.2 Eigene Funktionen erstellen

```r
# Eine einfache Funktion, die zwei Zahlen addiert
addiere <- function(a, b) {
  return(a + b)
}

addiere(5, 7)
```

```
## [1] 12
```

```r
numbers_to_n <- function(n) {
  summe <- n*(n+1)/2
}
numbers_to_n(100)
```

4. Hilfe in R

```r
?seq
vignette(package = "dplyr")
```

Packages

```
install.packages(c("ggplot2", "dslabs", "dplyr"))
```

5. Einfache Navigation in Dataframes

Erkunde den Datensatz "murders"

```
library(dslabs)
data(murders)

print(murders)
```

```
##                   state abb        region population total
## 1               Alabama  AL         South    4779736   135
## 2                Alaska  AK          West     710231    19
## 3               Arizona  AZ          West    6392017   232
## 4              Arkansas  AR         South    2915918    93
## 5            California  CA          West   37253956  1257
## 6              Colorado  CO          West    5029196    65
## 7           Connecticut  CT     Northeast    3574097    97
## 8              Delaware  DE         South     897934    38
## 9  District of Columbia  DC         South     601723    99
## 10              Florida  FL         South   19687653   669
## 11              Georgia  GA         South    9920000   376
## 12               Hawaii  HI          West    1360301     7
## 13                Idaho  ID          West    1567582    12
## 14             Illinois  IL North Central   12830632   364
## 15              Indiana  IN North Central    6483802   142
## 16                 Iowa  IA North Central    3046355    21
## 17               Kansas  KS North Central    2853118    63
## 18             Kentucky  KY         South    4339367   116
## 19            Louisiana  LA         South    4533372   351
## 20                Maine  ME     Northeast    1328361    11
## 21             Maryland  MD         South    5773552   293
## 22        Massachusetts  MA     Northeast    6547629   118
## 23             Michigan  MI North Central    9883640   413
## 24            Minnesota  MN North Central    5303925    53
## 25          Mississippi  MS         South    2967297   120
## 26             Missouri  MO North Central    5988927   321
## 27              Montana  MT          West     989415    12
## 28             Nebraska  NE North Central    1826341    32
## 29               Nevada  NV          West    2700551    84
## 30        New Hampshire  NH     Northeast    1316470     5
## 31           New Jersey  NJ     Northeast    8791894   246
## 32           New Mexico  NM          West    2059179    67
## 33             New York  NY     Northeast   19378102   517
## 34       North Carolina  NC         South    9535483   286
## 35         North Dakota  ND North Central     672591     4
## 36                 Ohio  OH North Central   11536504   310
```

```
## 37           Oklahoma  OK         South   3751351   111
## 38             Oregon  OR          West   3831074    36
## 39       Pennsylvania  PA     Northeast  12702379   457
## 40       Rhode Island  RI     Northeast   1052567    16
## 41     South Carolina  SC         South   4625364   207
## 42       South Dakota  SD North Central    814180     8
## 43          Tennessee  TN         South   6346105   219
## 44              Texas  TX         South  25145561   805
## 45               Utah  UT          West   2763885    22
## 46            Vermont  VT     Northeast    625741     2
## 47           Virginia  VA         South   8001024   250
## 48         Washington  WA          West   6724540    93
## 49      West Virginia  WV         South   1852994    27
## 50          Wisconsin  WI North Central   5686986    97
## 51            Wyoming  WY          West    563626     5
```

```r
head(murders)
```

```
##        state abb region population total
## 1    Alabama  AL  South    4779736   135
## 2     Alaska  AK   West     710231    19
## 3    Arizona  AZ   West    6392017   232
## 4   Arkansas  AR  South    2915918    93
## 5 California  CA   West   37253956  1257
## 6   Colorado  CO   West    5029196    65
```

```r
str(murders)
```

```
## 'data.frame':    51 obs. of  5 variables:
##  $ state     : chr  "Alabama" "Alaska" "Arizona" "Arkansas" ...
##  $ abb       : chr  "AL" "AK" "AZ" "AR" ...
##  $ region    : Factor w/ 4 levels "Northeast","South",..: 2 4 4 2 4 4 1 2 2 2 ...
##  $ population: num  4779736 710231 6392017 2915918 37253956 ...
##  $ total     : num  135 19 232 93 1257 ...
```

```r
names(murders)
```

```
## [1] "state"      "abb"        "region"     "population" "total"
```

```r
nrow(murders)
```

```
## [1] 51
```

```r
table(murders$region)
```

```
## 
##     Northeast         South North Central          West 
##             9            17            12            13
```

5.2 Spalten und Zeilen auswählen

```r
# Eine Spalte auswählen
murders$state
```

```
##  [1] "Alabama"              "Alaska"               "Arizona"
##  [4] "Arkansas"             "California"           "Colorado"
##  [7] "Connecticut"          "Delaware"             "District of Columbia"
## [10] "Florida"              "Georgia"              "Hawaii"
## [13] "Idaho"                "Illinois"             "Indiana"
## [16] "Iowa"                 "Kansas"               "Kentucky"
## [19] "Louisiana"            "Maine"                "Maryland"
## [22] "Massachusetts"        "Michigan"             "Minnesota"
## [25] "Mississippi"          "Missouri"             "Montana"
## [28] "Nebraska"             "Nevada"               "New Hampshire"
## [31] "New Jersey"           "New Mexico"           "New York"
## [34] "North Carolina"       "North Dakota"         "Ohio"
## [37] "Oklahoma"             "Oregon"               "Pennsylvania"
## [40] "Rhode Island"         "South Carolina"       "South Dakota"
## [43] "Tennessee"            "Texas"                "Utah"
## [46] "Vermont"              "Virginia"             "Washington"
## [49] "West Virginia"        "Wisconsin"            "Wyoming"
```

```r
# Eine Zeile auswählen
murders[2, ]
```

```
##    state abb region population total
## 2 Alaska  AK   West     710231    19
```

```r
# Eine spezifische Zelle auswählen
murders[3, "population"]
```

```
## [1] 6392017
```

5.3 Beispiele der einfachen Manipulationen

```r
murders_sued <- gsub("South", "Sued", murders$region)
murders$region <- murders_sued

murders$murder_rate <- murders$total / murders$population
murders$rate_percent <- murders$total * 100 / murders$population

lowest <- min(murders$murder_rate)
highest <- max(murders$murder_rate)

murders_west <- murders[which(murders$region == "West"),]

murders_west[which(murders_west$murder_rate == min(murders_west$murder_rate) |
                murders_west$murder_rate == max(murders_west$murder_rate)), "state"]
```

```
## [1] "Arizona" "Hawaii"
```

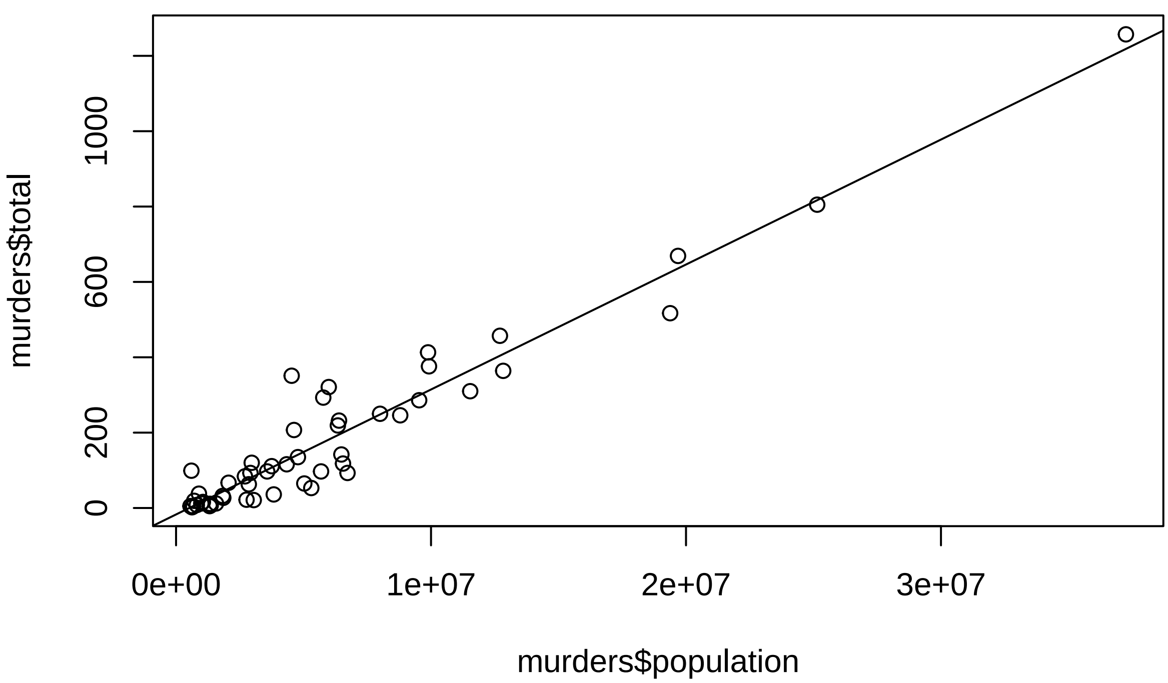
```r
murders[which(murders$murder_rate == max(murders$murder_rate)), "state"]
```

```
## [1] "District of Columbia"
```

```r
murders_20_100 <- murders[which(murders$total > 20 & murders$total < 100),]

murders_1000000 <- murders[which(murders$population > 1000000),]
murders_kleiner1000000 <- murders[which(murders$population <= 1000000),]

t.test(murders_1000000$murder_rate, murders_kleiner1000000$murder_rate)
```

```
##
##  Welch Two Sample t-test
##
## data:  murders_1000000$murder_rate and murders_kleiner1000000$murder_rate
## t = -0.39345, df = 7.1948, p-value = 0.7054
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -5.299978e-05  3.780729e-05
## sample estimates:
##    mean of x    mean of y
## 2.659969e-05 3.419593e-05
```

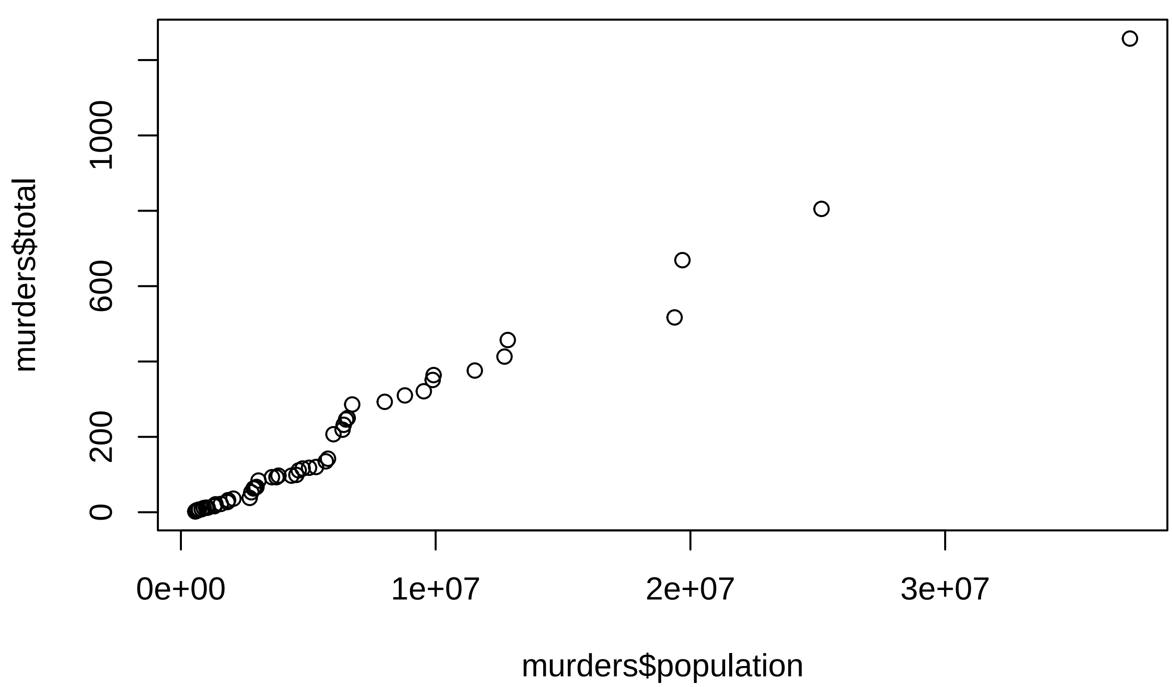
```r
plot(murders$population, murders$total)
abline(lm(total ~ population, murders))
```



```r
cor(murders$population, murders$total)
```

```
## [1] 0.9635956
```

```r
qqplot(murders$population, murders$total)
```



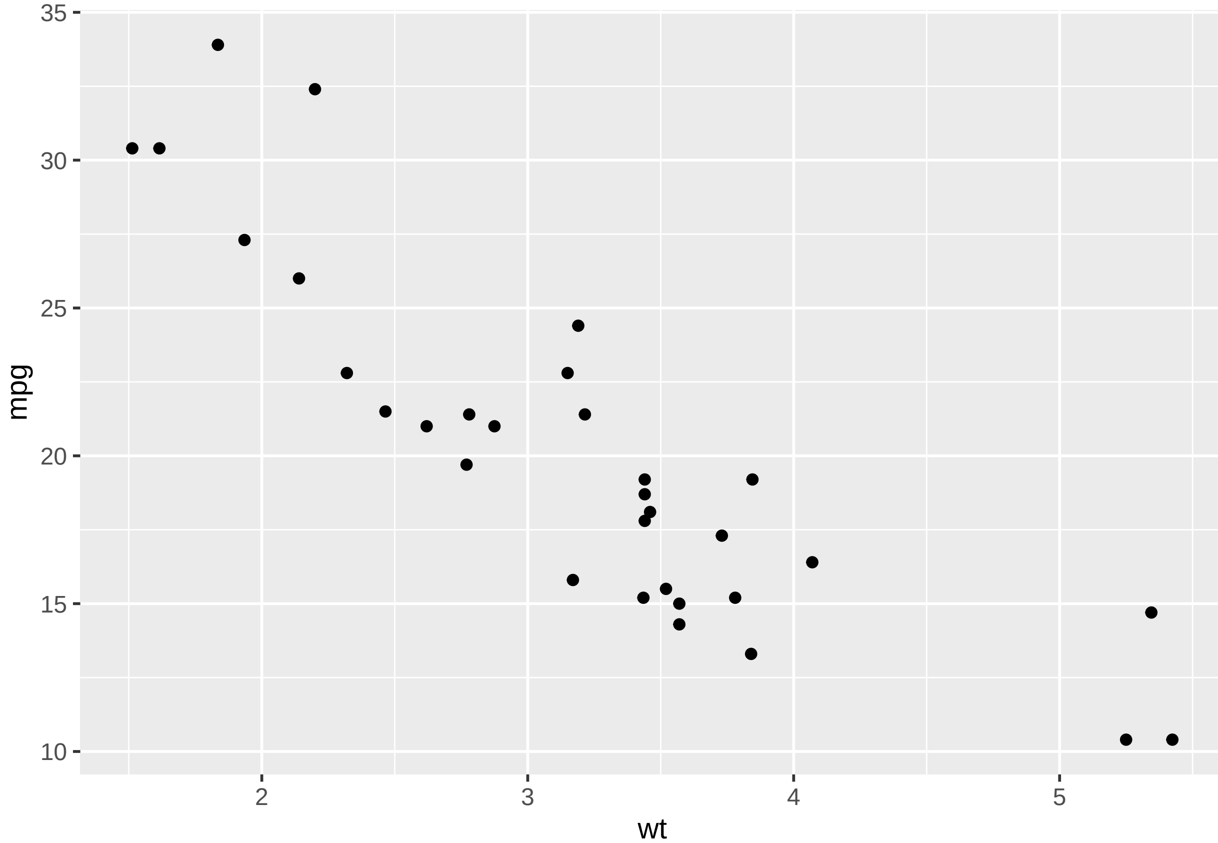
6. Wrangling mit dplyr

```r
library(dplyr)

data(mtcars)

data <- mtcars

data$auto_marke <- rownames(data)

rownames(data) <- NULL

# write.table(data, "output_data/mtcars.txt", sep = "\t")

selected_data <- select(data, mpg, cyl, hp, auto_marke)

filtered_data <- filter(data, mpg > 20)

arranged_data <- arrange(data, desc(mpg))
```

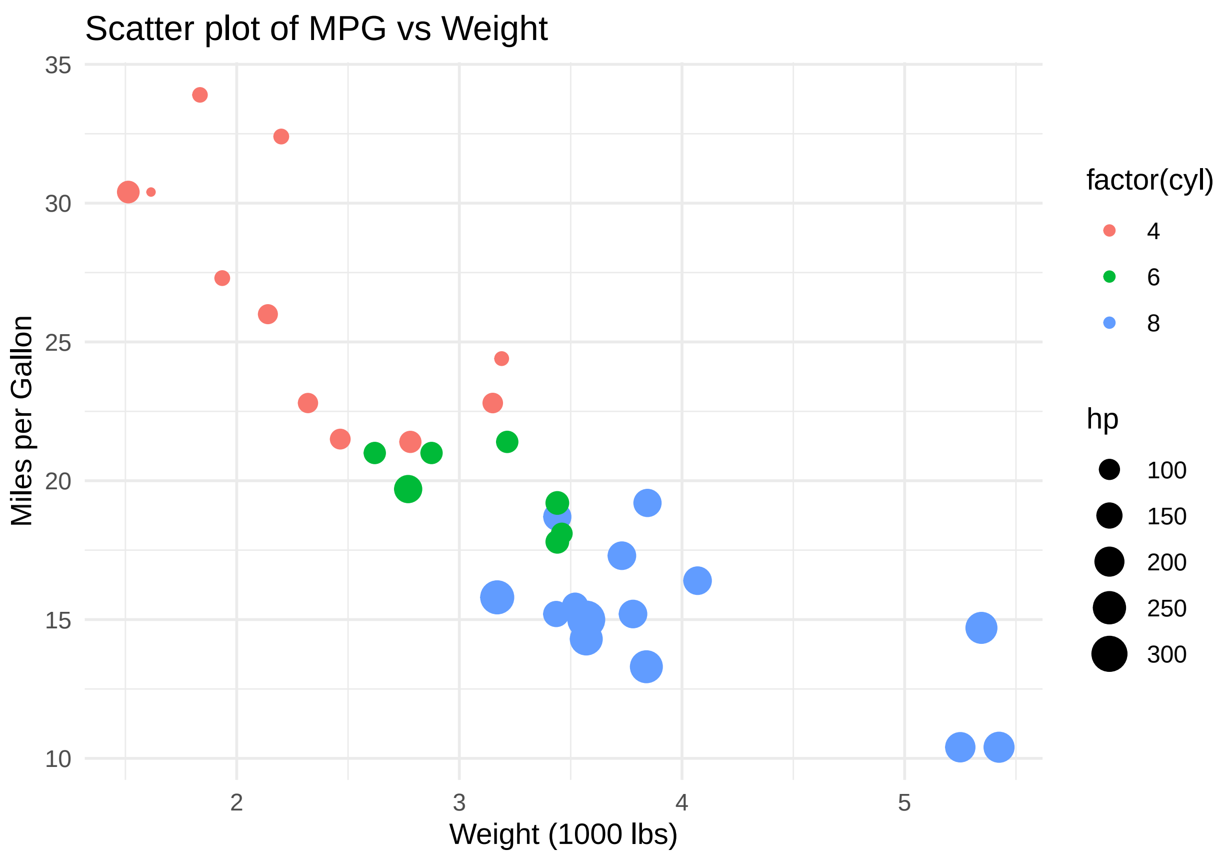
```r
mutated_data <- mutate(data, hp_to_weight = hp / wt)

summary_data <- summarise(data, avg_mpg = mean(mpg),
                          avg_hp = median(hp))

# mean over all columns
summary_data_all <- data %>% summarise(across(where(is.numeric), mean, na.rm = TRUE))


summary_data_piped <- data %>%
  filter(mpg > 20) %>%
  mutate(hp_to_weight = hp / wt) %>%
  summarise(avg_hp = mean(hp))


grouped_summary <- data %>%
  group_by(cyl) %>%
  summarise(
    avg_mpg = mean(mpg),
    avg_hp = mean(hp)
  )

###############################################################################


df1 <- data.frame(id = 1:5, value1 = letters[1:5])
df2 <- data.frame(id = 3:7, value2 = LETTERS[3:7])

inner <- inner_join(df1, df2, by = "id")

left <- left_join(df1, df2, by = "id")

right <- right_join(df1, df2, by = "id")

full <- full_join(df1, df2, by = "id")
```

7. Daten-Visualisierung

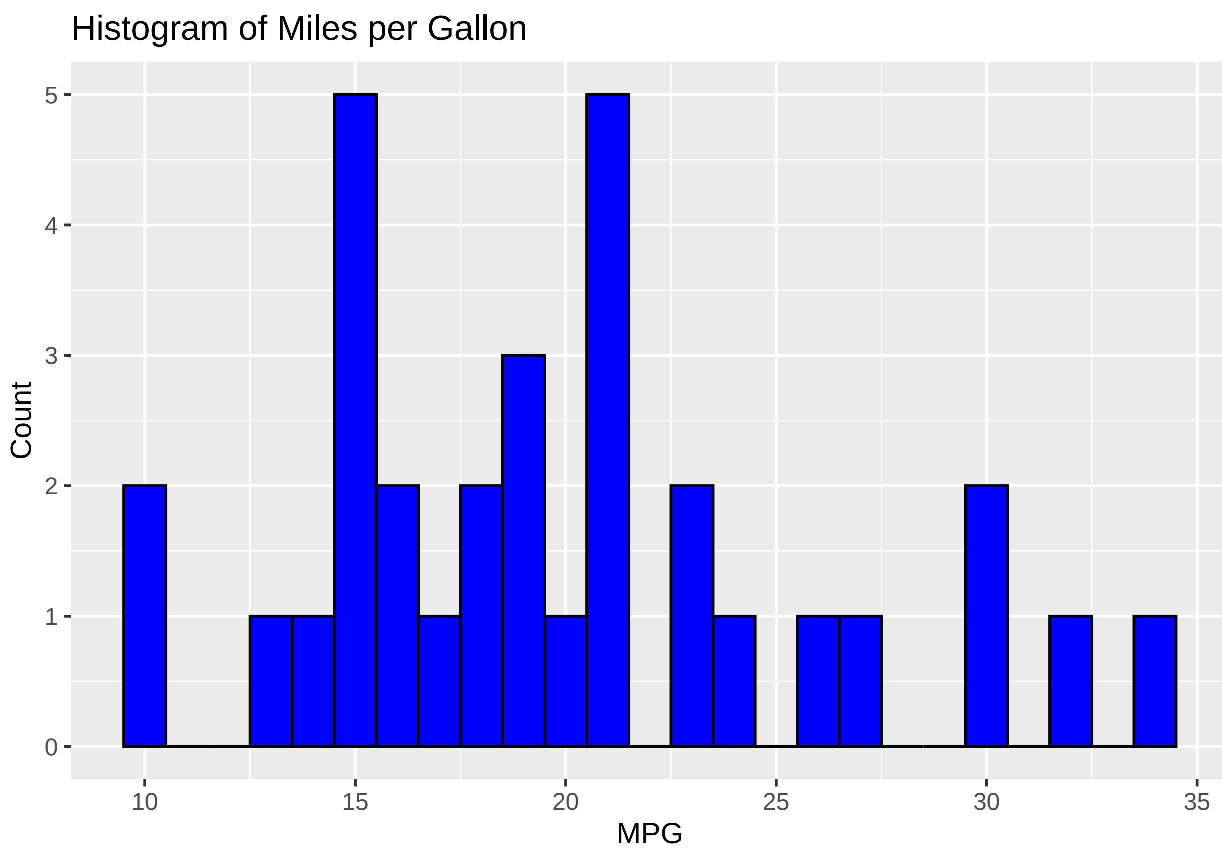
```r
library(ggplot2)

ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point()
```



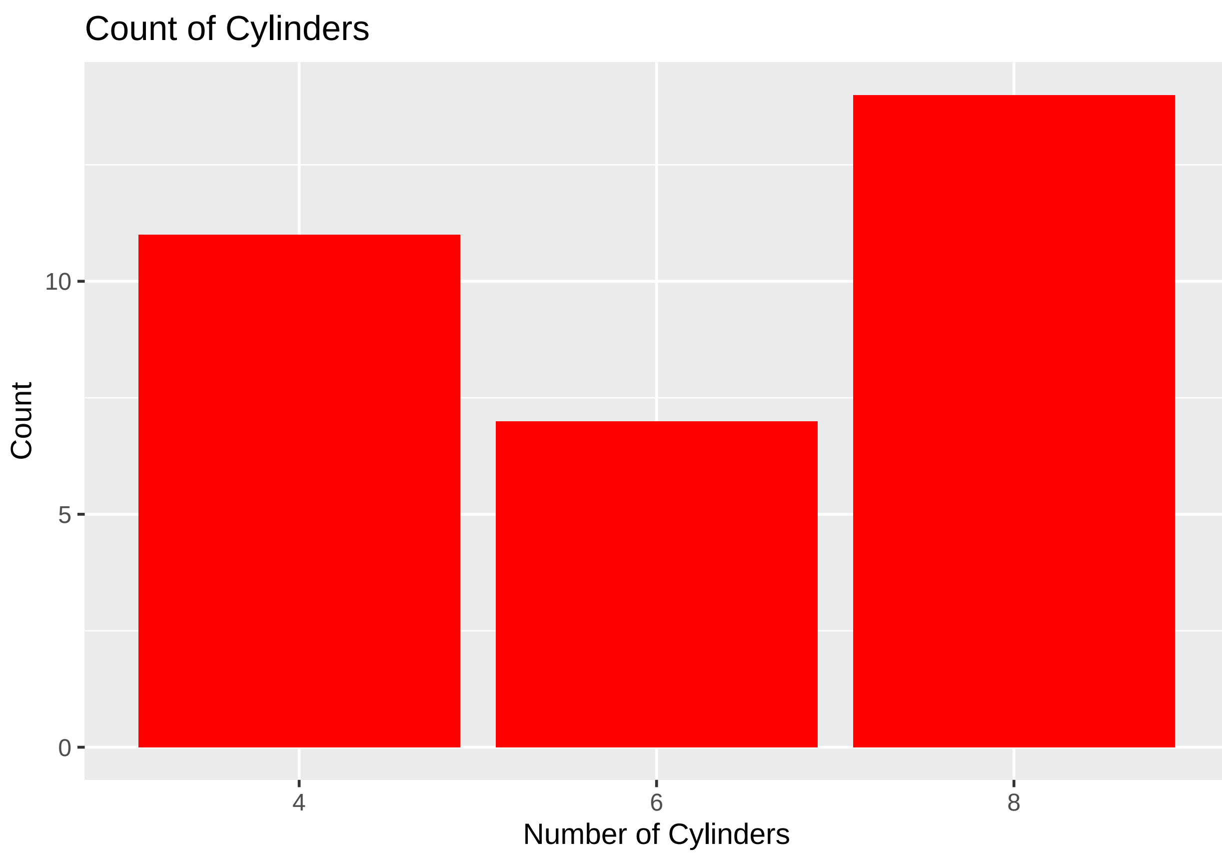

```r
ggplot(mtcars, aes(x = wt, y = mpg, color = factor(cyl), size = hp)) +
  geom_point() +
  labs(title = "Scatter plot of MPG vs Weight", x = "Weight (1000 lbs)", y = "Miles per Gallon") +
  theme_minimal()
```



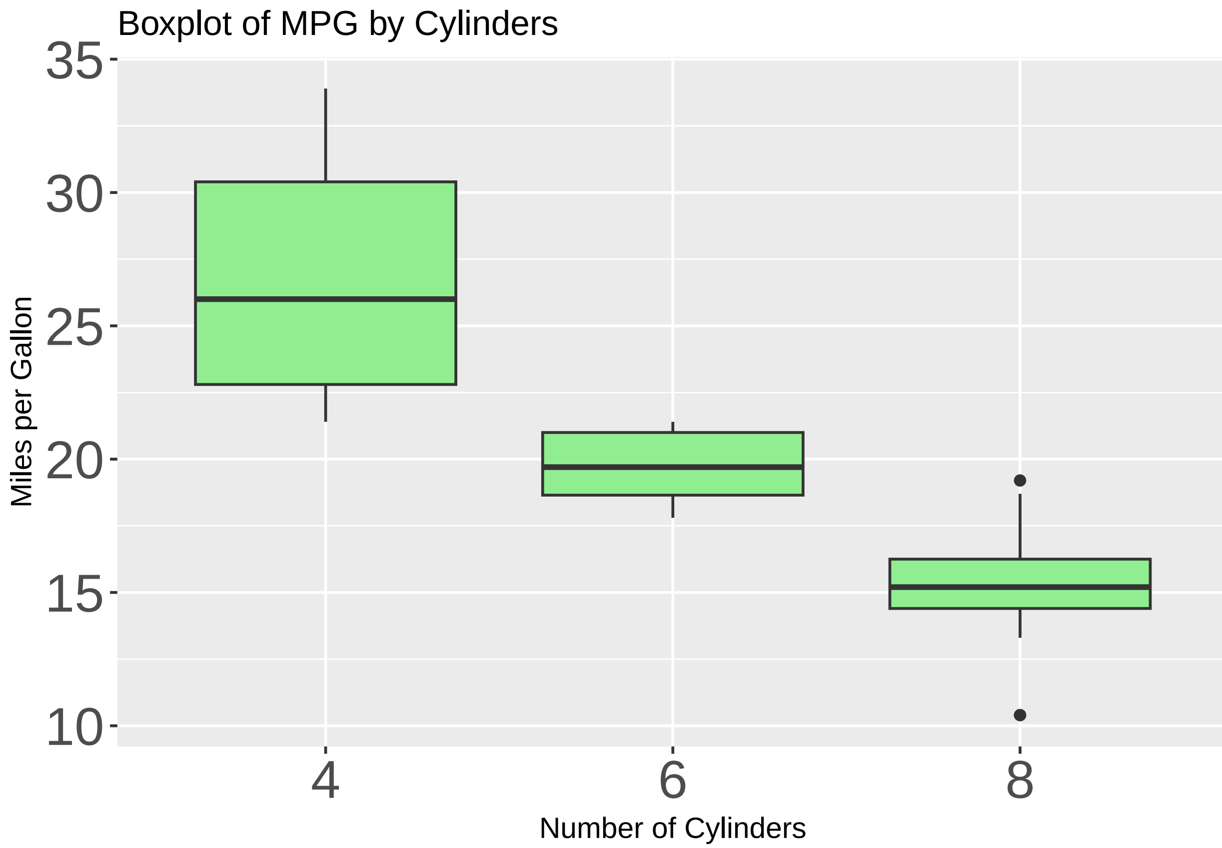

```r
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(binwidth = 1, fill = "blue", color = "black") +
  labs(title = "Histogram of Miles per Gallon", x = "MPG", y = "Count")
```




```r
ggplot(mtcars, aes(x = factor(cyl))) +
  geom_bar(fill = "#FF0000") +
  labs(title = "Count of Cylinders", x = "Number of Cylinders", y = "Count")
```




```
mtcars %>% ggplot(aes(x = factor(cyl), y = mpg)) +
  geom_boxplot(fill = "lightgreen") +
  labs(title = "Boxplot of MPG by Cylinders", x = "Number of Cylinders", y = "Miles per Gallon") +
  theme(axis.text = element_text(size = 20))
```



Farbenpaletten in R http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/#a-colorblind-friendly-palette

8. Übungen

Übung 1

1. Tabelle "survey_example.txt" einlesen.
2. Mittelwert von height_cm für männliche und für weibliche StudienteilnehmerInnnen ausrechen.
3. Fehlerquelle identifizieren.
4. Verwendet gsub(), um das Problem zu lösen. Tipp: "[0-9]+" steht für beliebige Anzahl an Zahlen, "\1" steht stellvertretend für eine Zahlengruppe.
5. Zeichnet ein Boxplot mit den Körpergrößen der Männer und der Frauen.

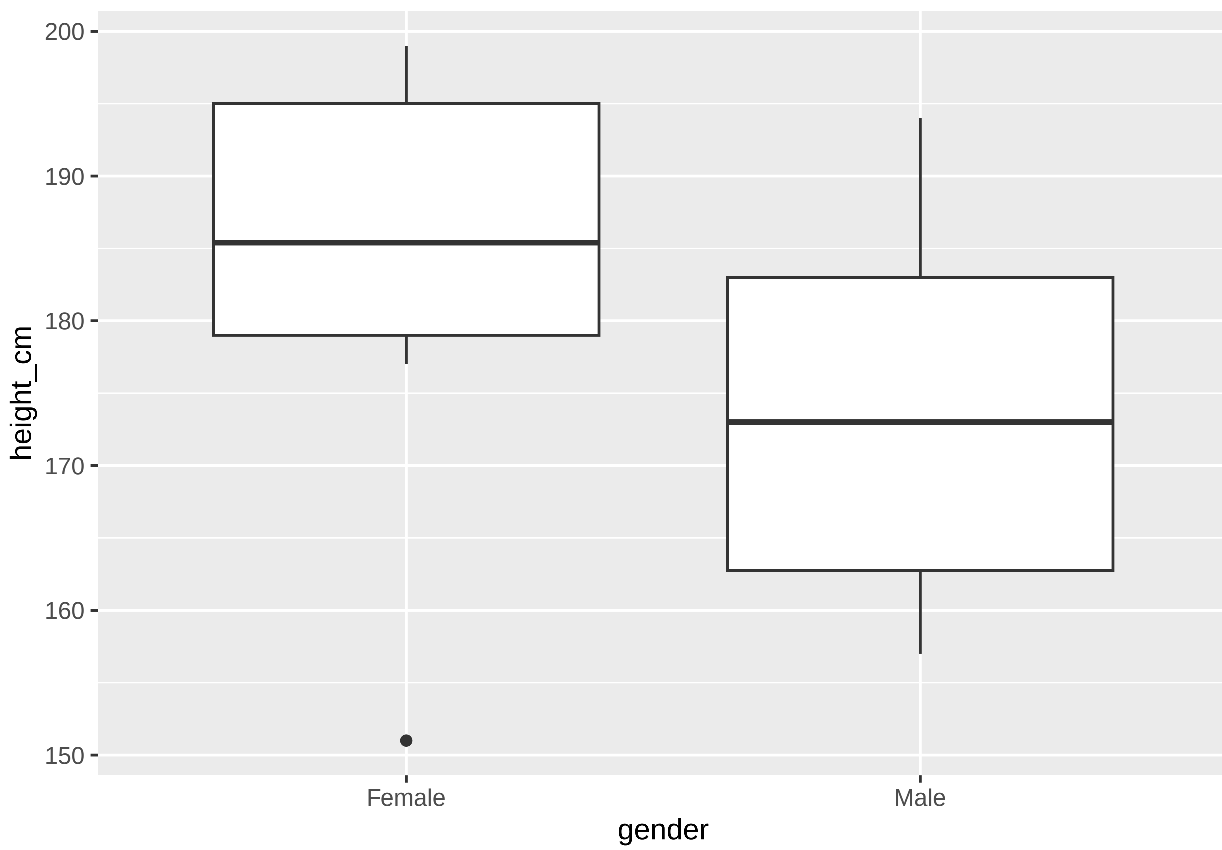
Lösung:

```
heights <- read.table("survey_example.txt")
new_heights <- gsub("([0-9]+)cm", "\\1", heights$height_cm)
new_heights <- gsub("([0-9]+),([0-9]+)", "\\1.\\2", new_heights)

heights %>% mutate(height_cm = as.numeric(new_heights)) %>%
  group_by(gender) %>% summarise(avg = mean(height_cm))
```

```
## # A tibble: 2 x 2
##   gender   avg
```

```
##    <chr>   <dbl>
## 1 Female   184.
## 2 Male     174.
```

```r
heights %>% mutate(height_cm = as.numeric(new_heights)) %>%
  ggplot(aes(x = gender, y = height_cm)) +
  geom_boxplot()
```



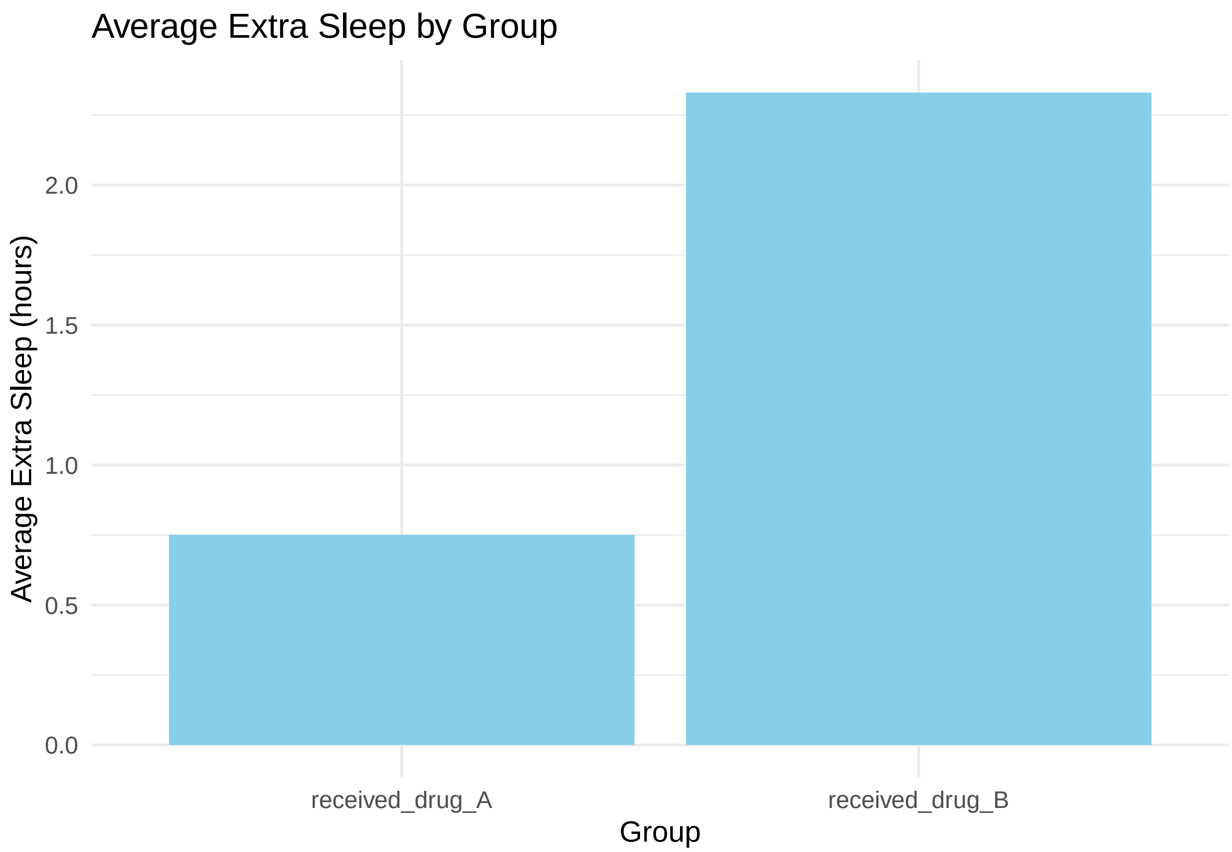
Übung 2

1. Erkundet den eingebauten Datensatz data(sleep).
2. Ändert die Gruppenbezeichungen von 1/2 zu "received_drug_a"/"received_drug_b". Tipp: ihr könnt recode() innerhalb von mutate() anwenden.
3. Berechnet den Mittelwert von extra_sleep für jede Gruppe.
4. Zeigt eure Ergebnisse in einem bar plot. Tipp: verwendet "stat ="identity" Argument.

Lösung:

```r
data(sleep)
?sleep

average_sleep <- sleep %>%
  mutate(group = recode(group, `1` = "received_drug_A", `2` = "received_drug_B")) %>%
  group_by(group) %>%
```

```
  summarise(mean_sleep = mean(extra))

ggplot(average_sleep, aes(x = factor(group), y = mean_sleep)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(x = "Group", y = "Average Extra Sleep (hours)", title = "Average Extra Sleep by Group") +
  theme_minimal()
```



Übung 3

0. library(dslabs).
1. Installiert package "ggrepel".
2. Erkundet den eingebauten Datensatz data(divorce_margarine).
3. Gibt es eine Korrelation zwischen der Anzahl der Scheidungen und dem Margarinekonsum?
4. Versucht die Problematik in einem Bild/Scatter plot/... darzustellen. Tipp: Schaut euch "geom_text_repel" function an.

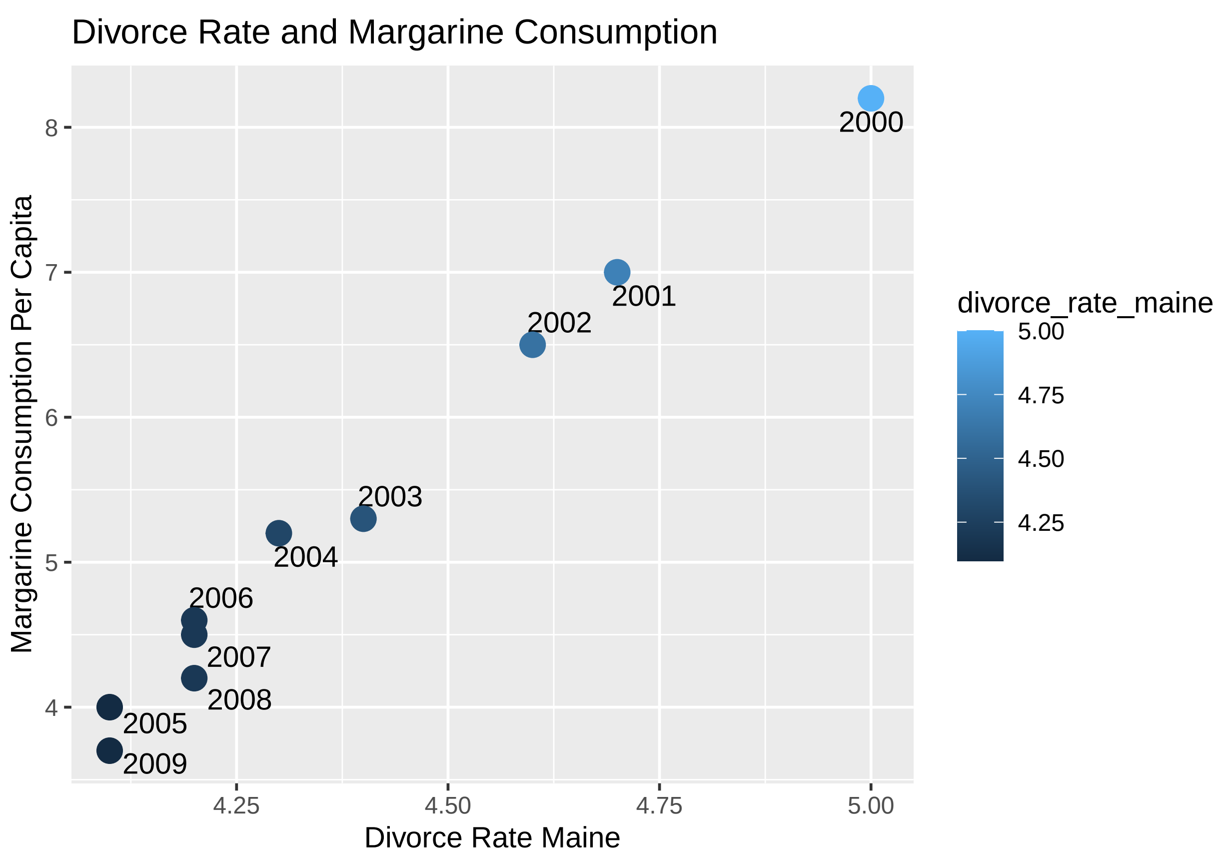
Lösung:

```
install.packages("ggrepel")
```

```
install.packages("ggrepel")
```

```
## Installiere Paket nach '/home/ekaterina/R/x86_64-pc-linux-gnu-library/4.1'
## (da 'lib' nicht spezifiziert)
```

```r
library(dslabs)
library(ggrepel)
data(divorce_margarine)

divorce_margarine %>%
  ggplot(aes(x = divorce_rate_maine, y = margarine_consumption_per_capita, label = year)) +
  geom_text_repel(nudge_x = 0.005) +
  geom_point(aes(color=divorce_rate_maine), size = 4) +
  xlab("Divorce Rate Maine") +
  ylab("Margarine Consumption Per Capita") +
  ggtitle("Divorce Rate and Margarine Consumption")
```



Übung 4

1. Erstellt einen "toy"-Datensatz:

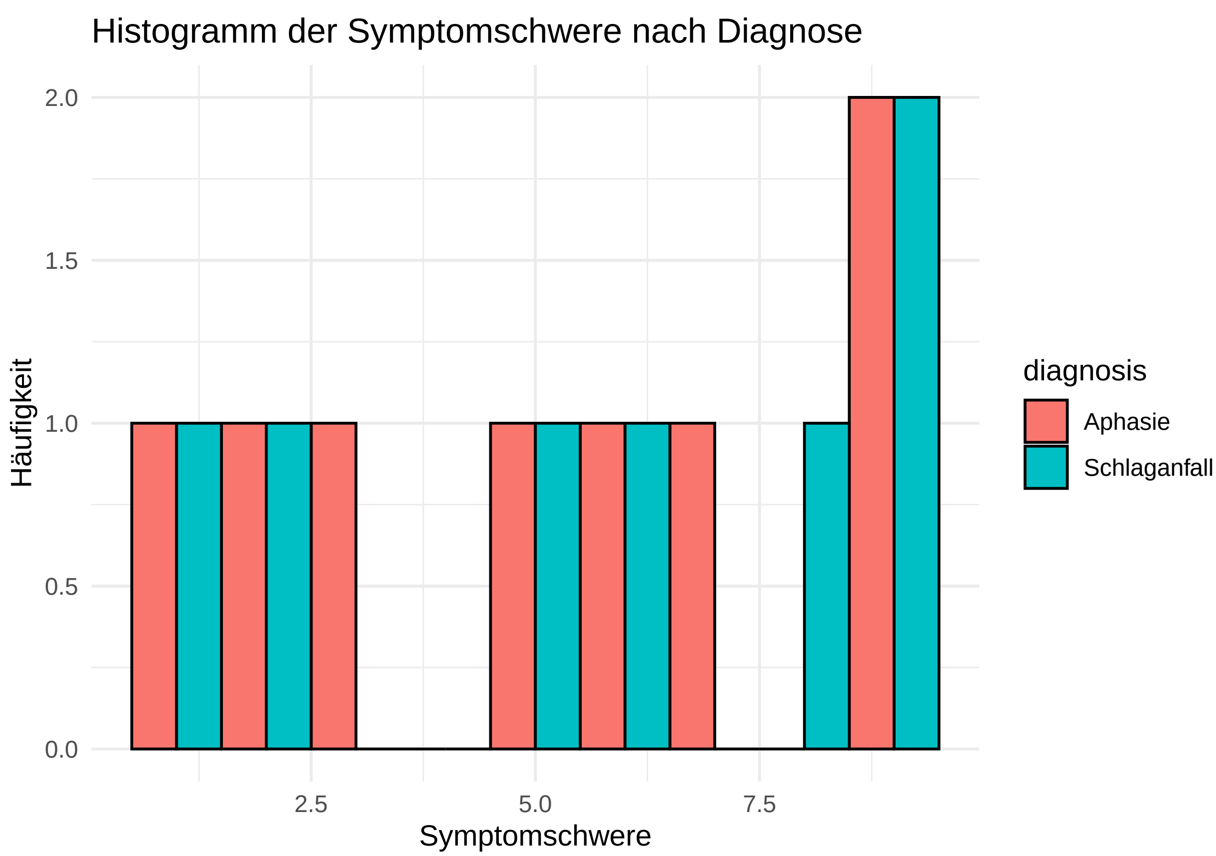
```r
neuro_data <- data.frame(
  patient_id = 1:100,
  diagnosis = sample(c("Aphasie", "Dysarthrie", "Schlaganfall"), 100, replace = TRUE),
  symptom_severity = sample(1:10, 100, replace = TRUE),
  therapy_sessions = sample(1:30, 100, replace = TRUE)
)
```

2. Filtert für Patienten mit Schlaganfall oder Aphasie und mehr als 20 Therapie-Sitzungen.

3. Erstellt ein Histogramm der Symptomschwere.

Lösung:

```
filtered_data <- neuro_data %>%
  filter(diagnosis %in% c("Aphasie", "Schlaganfall"),
         therapy_sessions > 20)

ggplot(filtered_data, aes(x = symptom_severity, fill = diagnosis)) +
  geom_histogram(binwidth = 1, position = "dodge", color = "black") +
  labs(x = "Symptomschwere", y = "Häufigkeit", title = "Histogramm der Symptomschwere nach Diagnose") +
  theme_minimal()
```



Übung 5

1. Führt folgende Befehle aus:

```
install.packages(c("maps", "mapdata"))
```

```
library(ggplot2)
library(maps)
library(mapdata)

# Beispiel Sehenswürdigkeiten in Wien, Salzburg und Graz
```

```r
landmarks <- data.frame(
  name = c("Stephansdom, Wien", "Festung Hohensalzburg, Salzburg", "Schlossberg, Graz"),
  lat = c(48.2082, 47.7990, 47.0702),
  lon = c(16.3738, 13.0430, 15.4395)
)

# Landkarte von Österreich
au_map <- map_data("worldHires", region = "Austria")
```

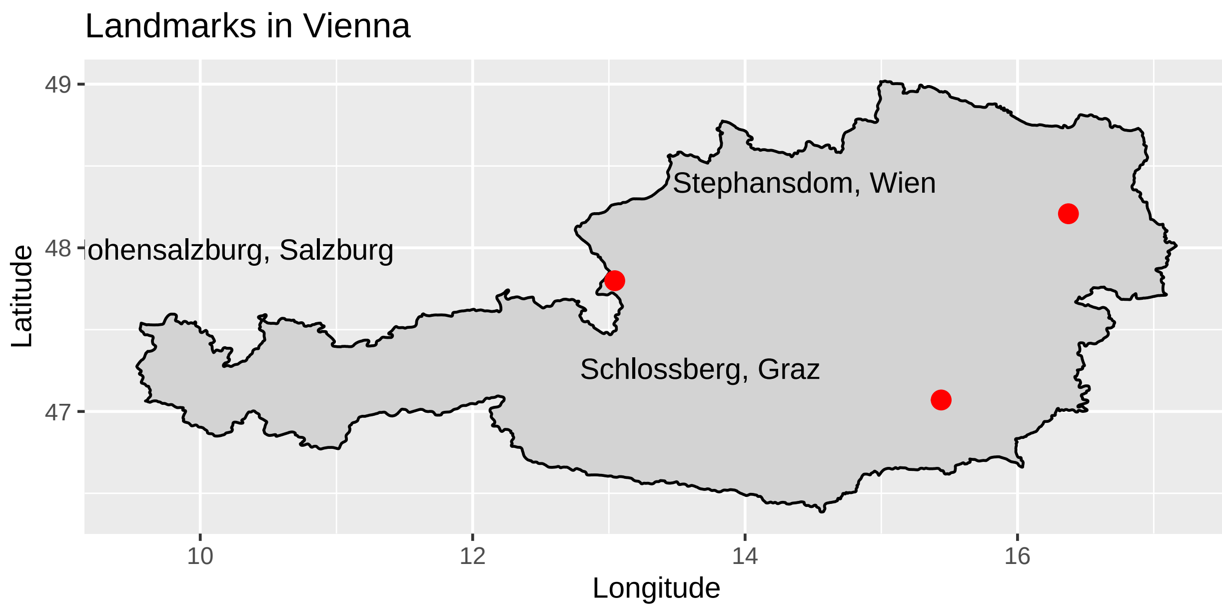
2. Erkundet ggmap-Dokumentation
3. Vervollstängigt den Code:

```r
ggplot() +
  geom_map(data = XXX, map = XXX,
           aes(x = long, y = lat, map_id = region),
           fill = "lightgrey", color = XXX) +
  geom_point(data = landmarks, aes(x = XXX, y = XXX), XXX = "red", size = 3) +
  geom_text(data = landmarks, aes(x = XXX, y = XXX, label = name), vjust = -1, hjust = 1.5, color = "bla
  labs(title = "Landmarks in Vienna", x = "Longitude", y = "Latitude") +
  coord_fixed(ratio = 1.2)
```

Lösung:

```r
ggplot() +
  geom_map(data = au_map, map = au_map,
           aes(x = long, y = lat, map_id = region),
           fill = "lightgrey", color = "black") +
  geom_point(data = landmarks, aes(x = lon, y = lat), color = "red", size = 3) +
  geom_text(data = landmarks, aes(x = lon, y = lat, label = name), vjust = -1, hjust = 1.5, color = "bla
  labs(title = "Landmarks in Vienna", x = "Longitude", y = "Latitude") +
  coord_fixed(ratio = 1.2)
```

```
## Warning in geom_map(data = au_map, map = au_map, aes(x = long, y = lat, :
## Ignoring unknown aesthetics: x and y
```


Landmarks in Vienna
49
48
47
Latitude
10
12
14
16
Longitude
Stephansdom, Wien
ohensalzburg, Salzburg
Schlossberg, Graz