Aufgabe 2.2: Explorative Datenanalyse mit R (9 Punkte) (STO-deskrSta04 | 3)

(i) Lesen Sie die Datei „KfzDat.csv“ in das data frame `dat.kfz` ein. Die Datei finden Sie in unserem Moodle-Kursraum.

(ii) Verschaffen Sie sich einen Überblick über den Datensatz (z.B. mithilfe der Funktionen `summary` und `str`).

(iii) Ordnen Sie die Werte des Merkmals `TypklasseVK` aus `dat.kfz` einer der Kategorien `0-10, 11-15, 16-20, 21-25, 26-34` zu und überschreiben Sie den Wert des Merkmals `TypklasseVK` mit der entsprechende Kategorie. Überprüfen Sie mit der Funktion `summary`, ob alle Werte einer Kategorie zugeordnet wurden.

(iv) Warum ergibt

```
PS <- pull(dat.kfz, "PS")
length(PS[PS<=100])+length(PS[PS>100])
```

mehr als die Gesamtanzahl aller Beobachtungen `nrow(dat.kfz.ps)`?

Mit welchem Ausdruck kann man die Anzahl der Einträge mit mehr als 100 PS ermitteln?

Hinweis: Mit der Funktion `pull` *kann man ein Merkmal eines Tibbles als Vektor extrahieren.*

(v) Löschen Sie alle Ausreißer mit $PS > 1000$, alle Beobachtungen mit negativem PS-Wert, sowie alle Beobachtungen bei denen irgendein Wert fehlt. Kontrollieren Sie Ihr Ergebnis mit `summary( dat.kfz )`.

(vi) Erstellen Sie ein neues Merkmal `PS.Kat` vom Datentyp Faktor, in der Sie die Werte des Merkmals `PS` einer der Kategorien `0-70, 71-90, 91-110, 111-130, 131-150, 151-200, 201-1000` zuordnen.

(vii) Erstellen Sie eine zweidimensionale Kontingenztabelle für das arithmetische Mittel des Preises je PS-/VK-Typklasse-Kategorie-Kombination.

Erstellen Sie ferner eine Kontingenztabelle für die absoluten Häufigkeiten je PS-/VK-Typklasse-Kategorie-Kombination.

Hinweis: Verwenden Sie die Hilfefunktion, um sich die Varianten der Funktionen `tapply` *und* `table` *für zwei Faktoren anzusehen.*

Lösung:

```
#
library(tidyverse)
# (i) (0,5 Punkte)
# über Import-Oberfläche generiert
dat.kfz <- read_delim("../KfzDat.csv", delim = ";",
    escape_double = FALSE, trim_ws = TRUE)
# (ii) (0,5 Punkte)
str(dat.kfz)

spc_tbl_ [44,066 x 5] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Id         : num [1:44066] 26509194 26538120 26558946 26518914 26519850 ...
 $ Preis      : num [1:44066] 11400 15550 12250 13990 26000 ...
 $ BerufsId   : num [1:44066] NA 566 NA NA 331 422 335 NA NA NA ...
 $ TypklasseVK: num [1:44066] 15 11 17 16 12 15 14 19 22 15 ...
 $ PS         : num [1:44066] 150 75 130 50 126 150 75 100 85 131 ...
 - attr(*, "spec")=
  .. cols(
  ..   Id = col_double(),
  ..   Preis = col_double(),
  ..   BerufsId = col_double(),
  ..   TypklasseVK = col_double(),
  ..   PS = col_double()
  .. )
 - attr(*, "problems")=<externalptr>
```

```r
# man sieht, dass alle Spalten den Datentyp INTEGER besitzen.
# (iii) (1 Punkt)
dat.kfz$TypklasseVK <- cut(dat.kfz$TypklasseVK, breaks=c(0,10,15,20,25,34),right=T, include.lowest = T)
summary(dat.kfz)
```

```
       Id               Preis           BerufsId      TypklasseVK
 Min.   :26498556   Min.   :  4315   Min.   :320     [0,10] : 2697
 1st Qu.:32103335   1st Qu.: 14940   1st Qu.:335     (10,15]:10728
 Median :35154828   Median : 21350   Median :362     (15,20]:19841
 Mean   :34981660   Mean   : 23986   Mean   :431     (20,25]: 8355
 3rd Qu.:37893010   3rd Qu.: 29190   3rd Qu.:509     (25,34]: 2445
 Max.   :40948128   Max.   :236215   Max.   :569
                    NA's   :2        NA's   :29283
       PS
 Min.   : -27
 1st Qu.:  90
 Median : 116
 Mean   : 159
 3rd Qu.: 150
 Max.   :1265
 NA's   :1310
```

```r
# (iv) (2 Punkte)
PS <- pull(dat.kfz, "PS")
length(PS[PS<=100])+length(PS[PS>100])-nrow(dat.kfz)
```

```
[1] 1310
```

```r
# Die NAs in PS werden bei beiden length-Befehlen mitgezählt, da NA bei der Indexselektion
# einen NA-Eintrag ergibt, der mitgezählt wird
length(PS[!is.na(PS) & PS>100])
```

```
[1] 28230
```

```r
# Check:
length(PS[!is.na(PS) & PS<=100])+length(PS[PS>100])-nrow(dat.kfz)
```

```
[1] 0
```

```r
# (v) (1 Punkt)
#
# erstmal alle Beobachtungen mit NAs loeschen
dat.kfz <- na.omit(dat.kfz)
# dann die PS-Ausreisser und negativen PS loeschen
dat.kfz <- dat.kfz |> filter( PS<1000 & PS>0 )
summary(dat.kfz) # es dürfen nirgends NA auftauchen, min(PS) muss positiv sein und max(PS) kleiner als 1000
```

```
       Id               Preis           BerufsId     TypklasseVK         PS
 Min.   :26498556   Min.   :  6386   Min.   :320   [0,10] : 776   Min.   : 40
 1st Qu.:31845200   1st Qu.: 14750   1st Qu.:335   (10,15]:3227   1st Qu.: 90
 Median :34508835   Median : 21000   Median :362   (15,20]:6279   Median :116
 Mean   :34723228   Mean   : 23334   Mean   :430   (20,25]:2514   Mean   :126
 3rd Qu.:37682829   3rd Qu.: 28425   3rd Qu.:509   (25,34]: 619   3rd Qu.:150
 Max.   :40947804   Max.   :209774   Max.   :569                  Max.   :700
```

```r
#
# voila!
#
# (vi) (1 Punkt)
dat.kfz$PS.Kat <- cut(dat.kfz$PS, breaks=c(0,70,90,110,130,150,200,1000),
                      right=T, include.lowest = T)
summary(dat.kfz$PS.Kat)
```

```
     [0,70]     (70,90]    (90,110]   (110,130]   (130,150]   (150,200]
       1398        2128        2571        2191        2223        1914
(200,1e+03]
        990
```

```r
# (vii) (3 Punkte)
tapply( pull(dat.kfz,Preis),
        list( pull(dat.kfz, TypklasseVK), pull(dat.kfz, PS.Kat) ), mean)
```

```
         [0,70] (70,90] (90,110] (110,130] (130,150] (150,200] (200,1e+03]
[0,10]  11808.1 11400.5  11789.5   11866.2   11633.6   11094.3     11570.0
(10,15] 15919.5 15535.7  16262.8   16008.0   15866.5   16094.3     15331.0
(15,20] 21251.7 21504.2  21320.1   21377.8   21575.6   21530.7     21242.7
(20,25] 33220.7 33277.3  32847.9   33466.4   33062.4   32961.8     34586.6
(25,34] 54312.4 56889.5  56470.0   56463.6   54203.4   56844.8     57172.7
```

```r
table(dat.kfz$TypklasseVK, dat.kfz$PS.Kat)
```

```
          [0,70] (70,90] (90,110] (110,130] (130,150] (150,200] (200,1e+03]
  [0,10]      80     131      149       124       131       107          54
  (10,15]    335     533      587       515       528       492         237
  (15,20]    654     983     1238      1037      1027       863         477
  (20,25]    262     393      483       401       431       361         183
  (25,34]     67      88      114       114       106        91          39
```

Aufgabe 2.5: Univariate Darstellung kategorialer Variablen mit R (keine Punkte)

(i) Lesen Sie die Datei „AbsolventenDat.csv“ in den data frame `dat.absolventen` ein. Beachten Sie, dass fehlende Werte teils mit „-“ und teils mit „k.A.“ gekennzeichnet sind. Die Datei finden Sie in unserem Moodle-Kursraum.

(ii) Lesen Sie die Komponente „Geschlecht“ des data frames in den Vektor `geschlecht` ein und erstellen Sie eine Häufigkeitstabelle dieses Vektors.

(iii) Erzeugen Sie mit dem Befehl `barplot` oder mit `geom_bar` ein einfaches Säulendiagramm der Häufigkeitstabelle des Merkmals `geschlecht` ohne weitere Formatierung.

(iv) Definieren Sie einen Vektor `plot.erg`, in den Sie das Ergebnis des `barplot`-Kommandos schreiben. Lesen Sie die Hilfe zum Kommando `barplot`, um zu verstehen, welche Werte der Vektor `plot.erg` enthält.

(v) Verwenden Sie den Vektor `plot.erg` und die Häufigkeitstabelle aus Teilaufgabe (ii), um die Säulen mit dem `text`-Kommando mit dem String

```
paste(round(prop.table(table(geschlecht))*100,1),"%") )
```

der die relativen Häufigkeiten der beiden Geschlechter in Prozent enthält, zu beschriften.

(vi) Schreiben Sie eine Funktion `pretty.barplot( dat.kat )`, die für einen Vektor `dat.kat` mit kategorialen Daten ein Säulendiagramm zeichnet, das mit den relativen Häufigkeiten beschriftet ist. Testen Sie Ihre Funktion mit dem Vektor `geschlecht`.

(vii) Verwenden Sie nun die Option `main` des `barplot`-Befehls, um die `pretty.barplot`-Funktion um das Argument `titel` zu erweitern, so dass der String `titel` als Überschrift des Säulendiagramms ausgegeben wird. Verwenden Sie ferner die Option `axes`, damit keine y-Achse gezeichnet wird. Testen Sie Ihre Funktion mit dem Vektor `geschlecht` und dem Titel „Geschlecht“.

(viii) Im Titel soll nun neben dem übergebenen String `Titel` noch die Anzahl der gültigen Werte ausgegeben werden. Testen Sie Ihre Funktion wieder mit dem Vektor `geschlecht` und dem Titel „Geschlecht“.

Hinweise:

i.) Verwenden Sie die Befehle `length` und `is.na`, um herauszufinden, wie viele gültige Werte der Vektor `geschlecht` enthält.

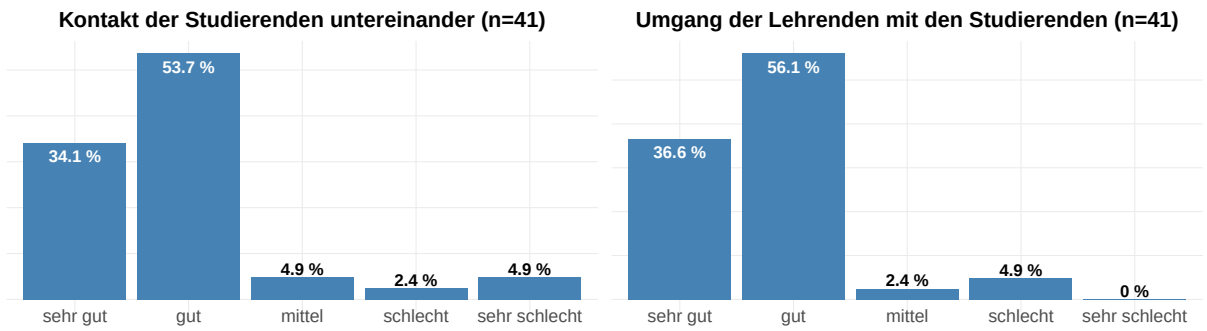
ii.) Mit dem Befehl `paste` können mehrere Strings zu einem zusammen gebaut werden.

(ix) Verwenden Sie die Funktion `pretty.barplot`, um Säulendiagramme für alle kategorialen Komponenten von `dat.absolventen` zu zeichnen.

(x) Verschönern Sie die Säulendiagramme aus Teilaufgabe (ix) noch weiter. Achten Sie dabei auf folgende Details:

- Prinzipiell soll die Prozentzahl oben in weißer Farbe in die dunkelblauen Säulen geschrieben werden. Wenn eine Säule zu klein für die Beschriftung ist, soll die relative Anzahl in schwarzer Farbe über die Säule geschrieben werden.
- Achten Sie auf die korrekte Reihenfolge der Kategorien in dem Säulendiagramm. Beispielsweise sollte die Reihenfolge bei den Beurteilungsfragen „sehr gut“, „gut“, „mittel“, „schlecht“, „sehr schlecht“ sein. Außerdem sollen bei den Beurteilungsfragen auch Ausprägungen auf der x-Achse aufgeführt werden, zu denen es keine Beobachtungen gibt, um die verschiedenen Fragen besser vergleichen zu können.

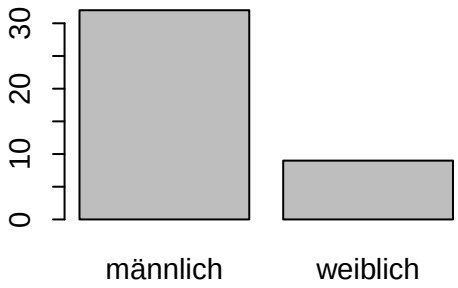
Beispielsweise könnte die Grafik so aussehen:



Hinweis: Wenn Sie wollen, dürfen Sie gerne die Pakete `tidyverse` *und* `ggplot2` *verwenden.*

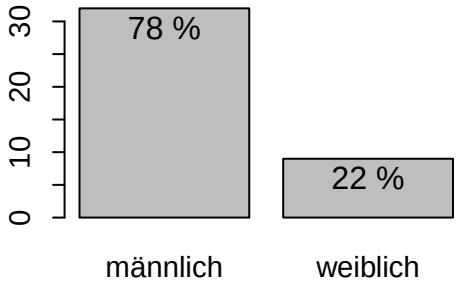
Lösung:

```
#
# (i) (über Oberfläche "Import Dataset" und dann anpassen )
dat.absolventen <- read.csv2("AbsolventenDat.csv",na.strings = c("-","k.A."))
# (ii)
geschlecht <- dat.absolventen$Geschlecht
table(geschlecht)

geschlecht
männlich weiblich
      32        9

# (iii)
barplot(table(geschlecht))
```

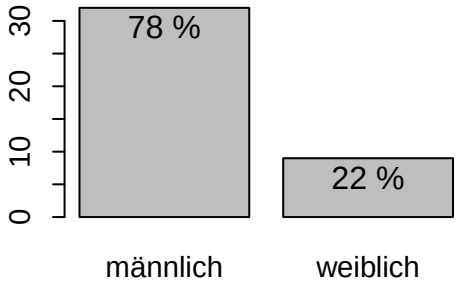



```
# (iv)
plot.erg <- barplot(table(geschlecht))
```

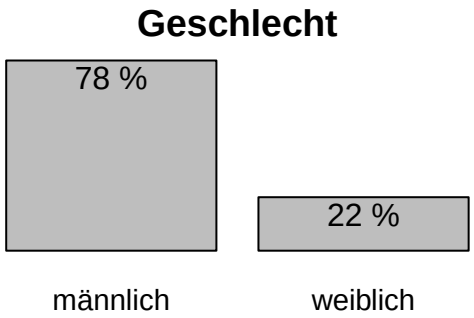
```
# (v)
text( plot.erg[,1], table(geschlecht)-3,
      paste(round(prop.table(table(geschlecht))*100,1),"%") )
```



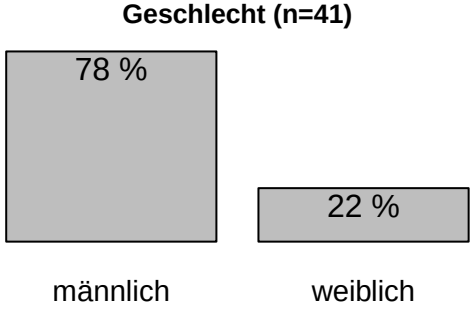

```
# (vi)
pretty.barplot <- function(dat.kat)
{
  plot.erg <- barplot(table(dat.kat))
  text( plot.erg[,1], table(dat.kat)-3,
        paste(round(prop.table(table(dat.kat))*100,1),"%") )
}
pretty.barplot(geschlecht)
```




```
# (vii)
pretty.barplot <- function(dat.kat, Titel)
{
  par( mar = c(5, 4, 4, 2) - 2 ) # nur wg. Rmarkdown
  plot.erg <- barplot(table(dat.kat),
                  main=Titel,
                  axes=FALSE)
  text( plot.erg[,1], table(dat.kat)-3,
        paste(round(prop.table(table(dat.kat))*100,1),"%") )
}
pretty.barplot(geschlecht, "Geschlecht")
```

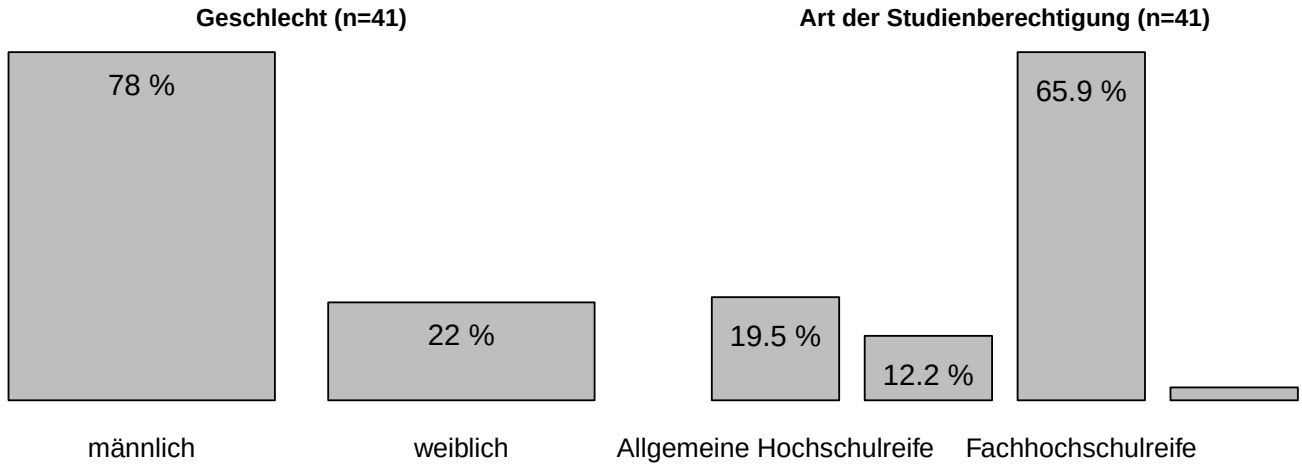



```
# (viii)
pretty.barplot <- function(dat.kat, Titel)
{
  par( mar = c(5, 4, 4, 2) -2 )
  plot.erg <- barplot(table(dat.kat),
                  main=paste(Titel," (n=",length(dat.kat[!is.na(dat.kat)]),")",
                             sep=""),
                  axes=FALSE, cex.main=0.8)
  text( plot.erg[,1], table(dat.kat)-3,
        paste(round(prop.table(table(dat.kat))*100,1),"%") )
}
pretty.barplot(geschlecht, "Geschlecht")
```



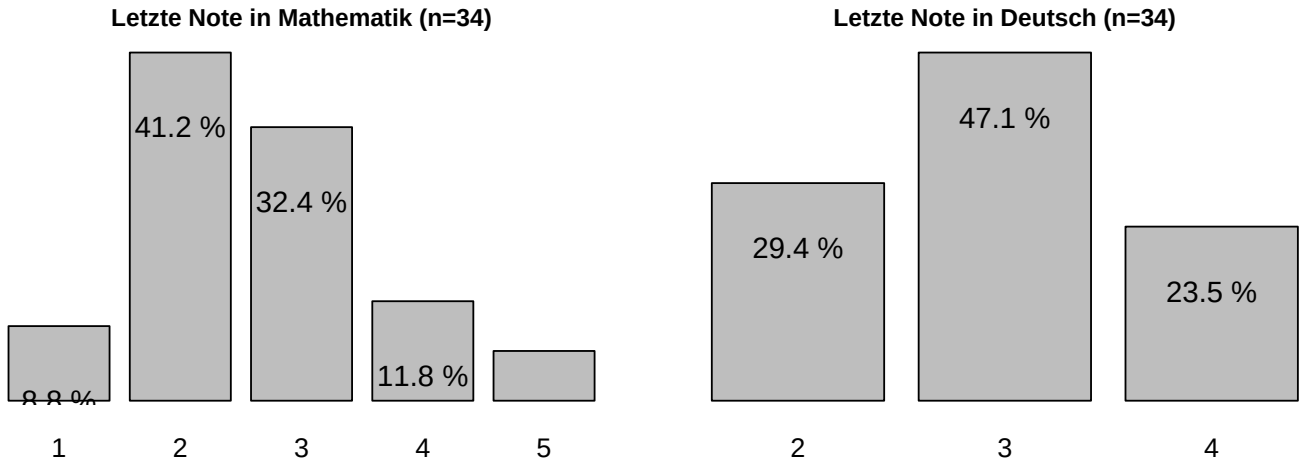
```r
# (ix)
par(mfrow=c(1,2))
pretty.barplot(dat.absolventen$Geschlecht,"Geschlecht")
```

```r
pretty.barplot(dat.absolventen$Art.der.Studienberechtigung,
               "Art der Studienberechtigung")
```



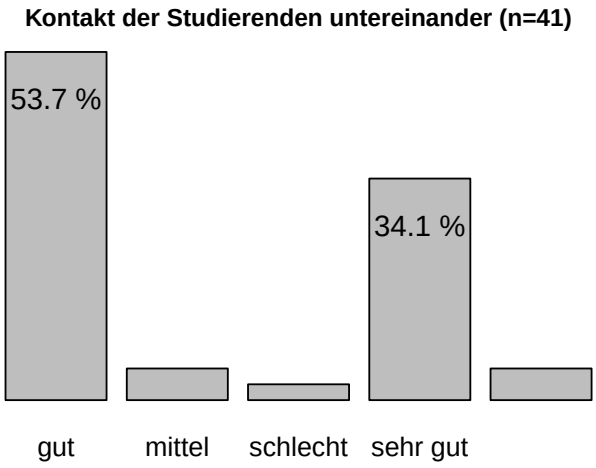

```r
pretty.barplot(dat.absolventen$Letzte.Note.in.Mathematik,"Letzte Note in Mathematik")
```

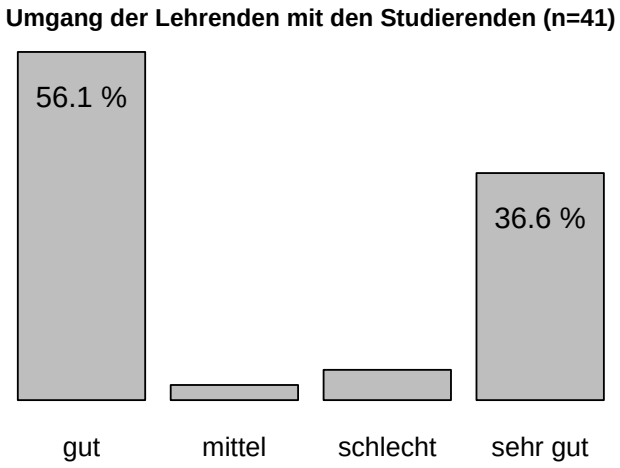
```r
pretty.barplot(dat.absolventen$Letzte.Note.in.Deutsch,"Letzte Note in Deutsch")
```




```r
pretty.barplot(dat.absolventen$Kontakt.der.Studierenden.untereinander,
               "Kontakt der Studierenden untereinander")
```

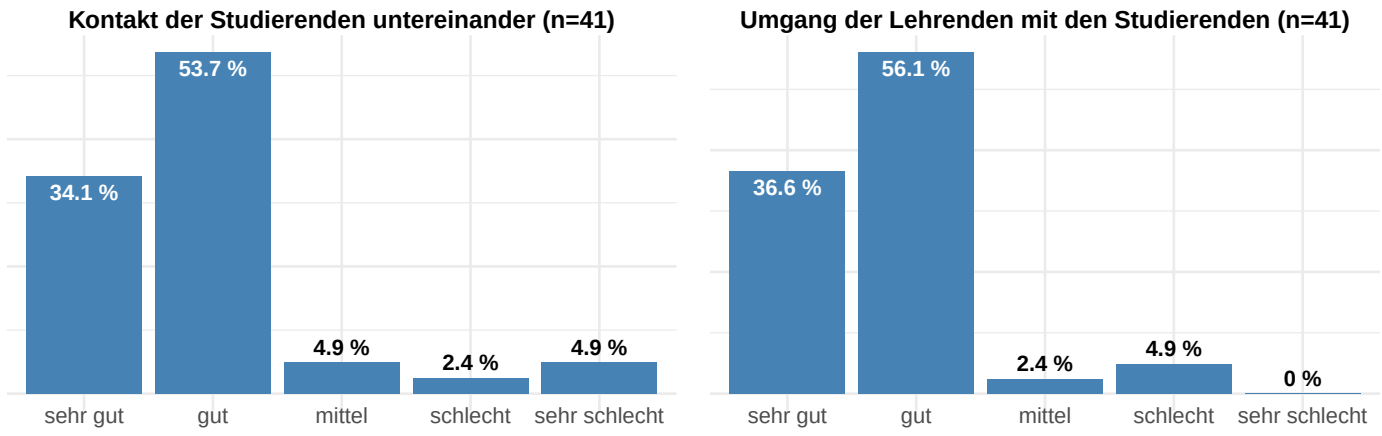
```r
pretty.barplot(dat.absolventen$Umgang.der.Lehrenden.mit.den.Studierenden,
               "Umgang der Lehrenden mit den Studierenden")
```






```r
# (x)
# noch schöneres Säulendiagramm (ich hab's mit ggplot gemacht)

# List of packages required for this analysis
pkg <- c("ggplot2","grid","gridExtra")

# Check if packages are not installed and assign the
# names of the packages not installed to the variable new.pkg
new.pkg <- pkg[!(pkg %in% installed.packages())]

# If there are any packages in the list that aren't installed,
# install them
if (length(new.pkg)) {
  install.packages(new.pkg, repos = "http://cran.rstudio.com")
}
# benötigte Pakete laden
library(ggplot2)
library(grid)
library(gridExtra)
#
even.prettier.barplot <- function(dat.kat, Titel)
{
  # Prozentuale Häufigkeiten des kategoriellen Vektors in data frame umwandeln,
  # weil der ggplot-Befehl eine data frame erwartet
  df <- as.data.frame(table(dat.kat)/length(dat.kat[!is.na(dat.kat)])*100)
  names(df)[1]="Cat"    # Kategorie
  lbl.vjust <- df$Freq  # Häufigkeit
  lbl.col <- rep("white",length(df$Freq)) # weiße Beschriftung bzw.
  lbl.col[lbl.vjust<8] = "black"          # schwarze Beschriftung, falls weniger
  # als acht Prozent
  lbl.vjust[lbl.vjust<8] = -0.5             # bis 8% über die Säule schreiben
  lbl.vjust[lbl.vjust>=8] = 1.5             # ab 8% in die Säule schreiben
  plot <- ggplot(data=df, aes(x=Cat,y=Freq) ) + ggtitle(paste(Titel," (n=",length(dat.kat[!is.na(dat.kat)])
    geom_bar(stat="identity", fill="steelblue")+
    geom_text(aes(label=paste(round(Freq,1),"%"), fontface = "bold"), vjust = lbl.vjust, color= lbl.col, siz
    theme_minimal()+
    theme(plot.title = element_text(size = 10,hjust = 0.5,margin = margin(t = 15, b = -10), face="bold"))+
    theme(axis.title.x = element_blank(), axis.text.x  = element_text(size=9,margin = margin(t = -5, b = 20)
    theme(axis.title.y = element_blank(), axis.text.y  = element_blank())

  return (plot)
}
attach(dat.absolventen)
# Umsortierung der Levels
Kontakt.der.Studierenden.untereinander <- factor(Kontakt.der.Studierenden.untereinander,
                                                 levels = c("sehr gut", "gut", "mittel", "schlecht", "sehr
Umgang.der.Lehrenden.mit.den.Studierenden <- factor(Umgang.der.Lehrenden.mit.den.Studierenden,
                                                    levels = c("sehr gut", "gut", "mittel", "schlecht", "sel
```

```
plot1 <- even.prettier.barplot(Kontakt.der.Studierenden.untereinander,
                               "Kontakt der Studierenden untereinander")
plot2 <- even.prettier.barplot(Umgang.der.Lehrenden.mit.den.Studierenden,
                               "Umgang der Lehrenden mit den Studierenden")
grid.arrange(plot1, plot2, ncol=2)
```




```
detach(dat.absolventen)
```