Thilo Ernst
Ulrich Pötter
Ruhr-Universität Bochum
Sommersemester 2006
Rechentechniken der Datenanalyse

Arbeitsblatt 4

1) Ablaufsteuerung (flow control): Häufig ist es nützlich, die Ausführung von Berechnungen von Bedingungen abhängig zu machen.

```
if (bedingung) anweisung
```

bzw.

```
if (bedingung) anweisung1
else anweisung2
```

Hier wird `anweisung1` ausgeführt, falls `bedingung` zutrifft (d.h. falls deren Auswertung `TRUE` ergibt). Andernfalls wird `anweisung2` ausgeführt. Wenn es jeweils um mehr als einen Befehl geht:

```
if (bedingung) {
    befehl1a
    befehl1b
    ...}
else {
    befehl2
    ...}
```

Wichtig ist, dass `bedingung` einen einzelnen Wahrheitswert (d.h. entweder `TRUE` oder `FALSE`) ergibt, und nicht etwa einen Vektor von Wahrheitswerten, der mehrere Elemente enthält. Um dies sicherzustellen, sind die Funktionen `all()`, `any()` und `identical()` hilfreich.

```
a <- 1:10
a >= 5
any(a >= 5)
all(a >= 5)
a > 0
any(a > 0)
all(a > 0)
a==1:10
identical(a, 1:10)
identical(a, 2:11)
all(a==1:10)
```

Wiederholungen lassen sich bspw. mit `while` realisieren:

```
i <- 0
while (i < 10) i <- i+1
i
```

Zum Arbeiten auf den einzelnen Elementen eines Vektors dient `for`:

```
a <- 1:100
b <- NULL
for (i in a) b <- c(b, sum(1:i))
b
```

2) Funktionen: `mean(x)` ist ein Funktionsaufruf. `mean` ist der Name der Funktion, `x` ein Argument. Häufig ist es sinnvoll, eine immer gleichbleibende Folge von Befehlen in einer selbstdefinierten Funktion zusammenzufassen. Ein triviales Beispiel:

```
quadrat <- function(x) {
    x^2
}
quadrat(3)
```

Eine Funktion gibt das Ergebnis der letzten Anweisung zurück. Der Rückgabewert lässt sich aber mit `return()` explizit festlegen.

Funktionen können beliebig viele Argumente entgegennehmen:

```
potenz <- function(x, potenz) {
    return(x^potenz)
}
potenz(2,4)
```

Die Argumente einer Funktion lassen sich auch mit voreingestellten Werten versehen, wodurch sie zu optionalen Argumenten werden. Die Angabe eines optionalen Arguments ist dann beim Funktionsaufruf nicht unbedingt erforderlich:

```
potenz <- function(x, potenz=2) {
    return(x^potenz)
}
potenz(2)
potenz(2,3)
potenz(2,potenz=3)
potenz(potenz=3,x=2)
```

Offenbar können die Argumente einer Funktion aslo sowohl anhand ihrer Position in der Liste der Argumente als auch über ihren Namen identifiziert werden.

Nebenbei bemerkt sind auch alle Operatoren in R Funktionen:

```
"+"(1,2)
"^"(2,4)
```

3) Entwickeln Sie eine Funktion `hzs`, die sich mit `h <- tapply(GENDER, X1110299, hzs)` anwenden lässt und pro Haushalt einen der folgenden Werte zurückgibt:

- 0, falls der Haushalt nur aus Frauen besteht
- 1, falls der Haushalt nur aus Männern besteht
- 2, falls der Haushalt aus Frauen und Männern besteht.

Dabei soll `GENDER` als `GENDER <- 2-D11102LL` definiert sein.

4) Modifizieren Sie die Funktion `hzs` aus Punkt 3 so, dass die obigen Werte nur noch für Mehrpersonenhaushalte zurückgegeben werden. Für Einpersonenhaushalte sollen -1 (Frau) oder -2 (Mann) zurückgegeben werden.

5) Generieren Sie möglichst unaufwändig einen Vektor, der die Dateinamen aller Datenfiles des PSID-Datensatzes enthält (allerdings ohne die Dateinamen einfach komplett einzutippen).

Nützlich können dabei bspw. `:`, `c()`, `for`, `paste()` und `as.character()` sein.