2.6.F Skripte/Schleifen – Ergänzungen und Bilder

2.6.F.1 Anwendung: Liste von Dateien

```
set *.txt
for DATEI in "$@" # oder: for DATEI
do
    echo ">>$DATEI<<"
done
#-----------------------------------
for DATEI in *.txt
do
    echo ">>$DATEI<<"
done
#-----------------------------------
set *.txt
while [ "$1" ]
do
    echo ">>$1<<"
    shift
done
```

2.6.F.2 Anwendung: Zeilen einer Datei x.txt auswerten

```
while read ZEILE
do
    echo ">>$ZEILE<<"
done < x.txt
```

2.6.F.3 Anwendung: Zahlen von 0 bis 20 erzeugen

```
for((x=0; x<21; ++x))
do
    echo ">>$x<<"
done
#-----------------------------------
for x in {0..20}
do
    echo ">>$x<<"
done
#-----------------------------------
declare -i x=0
while [ "$x" -lt 21 ]
do
    echo ">>$x<<"
    let ++x
done
#-----------------------------------
declare -i x=0
until [ "$x" -ge 21 ]
do
    echo ">>$x<<"
```

```
22      let ++x
23  done
24  #-------------------------------
25  declare -i x=0
26  while ((x<21))
27  do
28      echo $x
29      ((++x))
30  done
```

2.6.F.4 Abarbeiten der Positionsparameter mit einer Zählschleife, eval-Befehl

Wenn man mit den Positionsparametern `$0` bis `$9` zu tun hat, möchte man sie vielleicht mit einer Zählschleife abarbeiten, obwohl es dafür den viel besser passenden `shift`-Befehl gibt.

Mit dem `!`-Zeichen *vor* dem Namen der Zählvariablen bekommt man den Inhalt des Positionsparameters:

Terminal

```
schueler@debian964:~$ set Anton Berta Caesar  # $1, $2, $3
schueler@debian964:~$ echo "$@"
Anton Berta Caesar   # alle drei ("$*" ist ein String, "$@" drei)
schueler@debian964:~$ echo $3
Caesar               # das soll herauskommen, aber mit Hilfe von i
schueler@debian964:~$ i=3
schueler@debian964:~$ echo $i
3                    # netter Versuch
schueler@debian964:~$ echo ${!i}
Caesar               # so geht es wirklich
```

Man kann solche Probleme (indirekter Bezug über einen Variablennamen) auch mit dem Programm `eval` lösen. `eval` erlaubt es, einen Teil einer Befehlszeile zweimal auszuwerten.

Terminal

```
schueler@debian964:~$ set Anton Berta Caesar  # $1, $2, $3
schueler@debian964:~$ i=3
schueler@debian964:~$ set -x
schueler@debian964:~$ eval "x=\${$i}" # A) x=$3, B) x=Caesar
+ eval 'x=$3'
++ x=Caesar
schueler@debian964:~$ echo $x
Caesar
```

Schritt A war dabei das Konstruieren der Zeile, Schritt B das Ausführen.