UNIVERSITÄT SIEGEN

Fakultät IV
Betriebssysteme und verteilte Systeme
Prof. Dr. rer. nat. Roland Wismüller

Aufgabenblatt 2
Musterlösung

Vorlesung Betriebssysteme und nebenläufige Programmierung
Sommersemester 2025

Aufgabe 1: Unix Kommandos: cat, cp, tar, gzip, sort, wc

a)
```
bslab01% tar -zxvf u1.tar.gz          # auspacken (-z das Archiv durch gzip filtern)
```
oder
```
bslab01% gzip -d u1.tar.gz            # alternativ geht auch das Kommando „gunzip“
bslab01% tar -xvf u1.tar
```

Es werden zwei Dateien erzeugt: *name.txt* und *UnixKommandos.pdf*.

```
bslab01% ls -l
-rw-r--r--  1   bs106  users      409   Okt 11  2017   name.txt
-rw-r--r--  1   bs106  users   173034   Sep 21  2017   UnixKommandos.pdf
...
```

b)
```
bslab01% cat name.txt                 # Ausgabe des Inhaltes der Datei
bslab01% cat name.txt | more          # Anzeige der Datei seitenweise (jeweils ein Bildschirm)
```

Das Symbol „|“ bezeichnet eine Pipe. Auf der UNIX-Kommandoebene dienen Pipes zur Verkettung von Kommandos und werden durch einen senkrechten Strich („|“) dargestellt. Stellen Sie sich eine Pipe als Kommunikationskanal zwischen Prozessen vor. Ihr Hauptanwendungszweck liegt darin, dass ein Prozess (**cat** Kommando) einem anderen Prozess (**more** Filterprogramm) Eingabedaten zur Verfügung stellt.

```
bslab01% less name.txt
```

Der **less**-Befehl gibt die Eingabe seitenweise auf den Bildschirm aus. Sie verlassen ihn durch die Taste „q“. Wenn Sie etwas durch den **less**-Befehl anzeigen lassen, können Sie sich mit den „Cursor hoch“ und „Cursor runter“-Tasten in der Ausgabe bewegen.

c)
```
bslab01% sort -o sortiert.txt name.txt
```
oder
```
bslab01% sort name.txt -o sortiert.txt
```

d)
```
bslab01% sort -r name.txt
```

e)
```
bslab01% wc name.txt
```

f)
```
bslab01% wc -w name.txt
```

g)
```
bslab01% sort name.txt > nameS.txt
```
oder
```
bslab01% sort < name.txt > nameS.txt
```
Umleitung der Ausgabe von **sort** in die Datei **nameS.txt**.

Aufgabe 2: Unix Kommandos, Ein-/Ausgabe-Umleitung/Dateibearbeitung/Pipes

a)
```
bslab01% mkdir -p Uebungen/Uebung2/Programme
bslab01% cd Uebungen/Uebung2
bslab01% chmod u+rwx,og-rwx Programme # oder: chmod 0700 Programme
bslab01% ls -l                        # Rechte überprüfen
```

b)
```
bslab01% ln -s /usr/bin $HOME/bin
bslab01% ls -l $HOME/bin
lrwxrwxrwx 1 bs007 users 8 2009-11-02 12:56 bin -> /usr/bin
```

Bemerkung: Wenn Sie schon ein **bin** Unterverzeichnis unter **$HOME** haben, dann wird **$HOME/bin/bin** erzeugt.

c)
```
bslab01% cd $HOME/Uebungen/Uebung2
bslab01% cp /usr/bin/[lz]* Programme  # Dateien werden kopiert, aber nicht alle Rechte
bslab01% ls Programme | wc -w         # Anzahl der Dateien
95                                    # mögliche Ausgabe
```

Um nur die ausführbaren Programme, die mit dem Buchstaben „l“ oder „z“ beginnen, zu kopieren:

```
bslab01% cp `find /usr/bin/[lz]* -perm +0111` Programme
```

Zur überprüfung was kopiert wurde / würde (mit möglicher Ausgabe):

```
bslab01% ls -l `find /usr/bin/[lz]* -perm +0111 -print`  # -print ist optional
-rwxr-xr-x 1 root root    12180 2004-04-06 03:10 ./bin/last
...
-rwsr-x--- 1 root trusted 10036 2004-04-06 08:55 ./bin/ziptool
-rwxr-xr-x 1 root root     1885 2004-04-06 03:12 ./bin/zmore
lrwxrwxrwx 1 root root        8 2004-07-02 13:58 ./bin/zsh -> /bin/zsh
-rwxr-xr-x 1 root root    18512 2004-10-25 00:29 ./bin/zsoelim
```

d)
```
bslab01% ls Programme > index
bslab01% ls /usr/bin > indexBin
```

e)
```
bslab01% diff index indexBin | more
bslab01% diff index indexBin | grep '>' | wc -l
```

f)
```
bslab01% cat indexBin >> index
bslab01% sort -u index > index.sortiert
```

Alternativen zum letzten Kommando:

```
bslab01% sort -u -o index.sortiert index
bslab01% sort -u < index > index.sortiert
```

Bemerkung: **index.sortiert** und **indexBin** sollen gleich sein.

g)
```
bslab01% grep 'ch' indexBin
```

Grep („**G**lobal search for a **r**egular **e**xpression and **p**rint out matched lines“) ist das gebräuchlichste Kommando, um in Dateien nach bestimmten Mustern zu suchen. Das Kommando schreibt alle Zeilen der Eingabe, die das gesuchte Muster enthalten, auf die Standardausgabe.

Häufig interessiert man sich nicht für den exakten Inhalt der Zeile, die das Muster enthält, sondern einzig ob und ggf. wie oft das Muster enthalten ist, zum Beispiel:

```
bslab01% grep 'ch' indexBin >& /dev/null   # Keine Ausgabe! Standard- und Fehlerausgabe
                                           # werden in das „Null-Gerät“ umgeleitet.
bslab01% echo $?                           # Ergebniswert des letzten Kommandos
0                                          # d.h. Ergebnis „true“: Muster wurde gefunden

bslab01% grep -c 'ch' indexBin             # Anzeige der Anzahl von Zeilen, in denen das Muster
                                           # 'ch' gefunden wurde
```

h)
```
bslab01% ls /usr/bin | grep '^[lz]'> idx_lz
bslab01% diff idx_lz index
```

Hinweis: Das Programm **kdiff3** erlaubt das komfortable Vergleichen und Zusammenführen von Dateien bzw. Ordnern.

Aufgabe 3: Ein einfaches Shell-Skript

a) –

b)
```
bslab01% chmod +x hallo
bslab01% ./hallo
```

c)
```
echo Guten Morgen $USER !
```

d)
```
echo Guten Morgen $USER !
echo   # neue Zeile wegen besserer Lesbarkeit

# Bash–Variable
echo "Alle Argumente: $*"
echo "Anzahl der Argumente: $#"
echo "Skriptname ist: $0"
echo

echo Inhalt:
echo "------------------"
cat $0
echo "---Ende-----------"
```

Skript ausführen mit:
```
bslab01% sh ./hallo                    # Anstelle von sh kann man auch bash
```
oder
```
bslab01% chmod a+x hallo               # Ausführbar für user, group und other
bslab01% ./hallo          # und beliebige Argumente
```
oder
```
bslab01% chmod u+x hallo               # Ausführbar für user
bslab01% ./hallo          # und beliebige Argumente
```

Aufgabe 4: Shell Skript

```
tageszeit=`date +%H`      # wichtig: +%H ohne Leerzeichen !
if [ $tageszeit -lt 12 ] # oder: if test $tageszeit –lt 12
then
  echo "Guten Morgen!"
else
  if [ $tageszeit -ge 12 -a $tageszeit -lt 13 ]
  then
    echo "Guten Appetit!"
  else
    if [ $tageszeit -ge 17 ]
    then
      echo "Guten Abend!"
    else
      echo "Guten Tag!"
    fi
  fi
fi
```

Die eckigen Klammern sind eine andere semantische Schreibweise für **`test $tageszeit -lt 12`** mit den Argumenten „aktuelle Stunde“ (gespeichert in der Variablen **`tageszeit`**), „**`-lt`**“ (kleiner als) und „**`12`**“.

Gibt das Programm test den Status 0 (d.h. „wahr“) zurück, so wird alles zur Anweisung **`then`** gehörende ausgeführt. Gibt **`test`** den Status 1 (d.h. „falsch“) zurück, so wird alles zur Anweisung **`else`** gehörende ausgeführt. Da anschließend an die jeweiligen Schlüsselwörter eine Liste von Anweisungen folgen kann, wird mit der Anweisung **`fi`** (ein umgedrehtes **`if`**) ein if-Block beendet.