# LINUX Kurs

*Einige grundlegende Linux-Befehle*

# *Einige grundlegende Linux-Befehle*

## Inhaltsverzeichnis

## Kleine Geschichte: von Unix bis zu Linux

1971: AT&T Bell Labs, von Ken Thomson und Dennis Ritchie entwickelt
Ken Thomson ist auch der Autor der Programmiersprache C
der Unix-Quellcode wurde für US$ 50 an Universitäten gegeben.
Dadurch haben viele Leuten Unix weiter entwickelt.
1977: Erste BSD (Berkeley Software Distribution) von der Berkeley University CA
1984: AT&T wurde aufgeteilt -> Unix System Laboratories (USL) wurde gegründet, der Quellcode wurde jetzt für US$ 100'000 verkauft.
1984: Richard Stallman startet das GNU-Projekt. **GNU** steht für GNU's not Unix = ein Betriebssystem funktional äquivalent zu Unix aber keine einzige Zeile vom AT&T geschützten Code.
1990: GNU-System nahezu vollständig.
1991: Anfang vom Linux-Kernel durch Linus Torvalds (Helsinki, Finland).
1994: Erste Veröffentlichung von GNU/Linux.

## Die Open Source Philosophie

**G**NU **G**eneral **P**ublic **L**icense (**GPL**) ein Konzept der **F**ree **S**oftware **F**oundation

Die GNU General Public Licence garantiert:
1. den Zugang zum Quellcode
2. die Freiheit, die Software zu kopieren und weiterzugeben
3. die Freiheit, das Programm zu ändern
4. die Freiheit, das veränderte Programm - unter denselben Lizenzbedingungen - zu verbreiten

Die GNU-Philosophie schreibt nicht vor, dass die Weiterverbreitung kostenlos zu geschehen hat!

- Unabhängigkeit von einem einzigen Hersteller
- Tausende von ProgrammiererInnen in der ganzen Welt
- Schnelle Fehlerkorrektur, durch den Autor des Programmes oder man tut es selber!

## Was ist Linux

- eigentlich "nur" der Kernel
  der Kernel verwaltet:
  - die Hardware
  - die Prozesse
  - den Hauptspeicher
  - die Dateisysteme
  - das Netzwerk.

- eine Shell
- eine Sammlung (GNU-) Dienstprogramme
- eine grafische Oberfläche = X Window System / Xfree86™ (netzwerkfähig)
- ein echtes Multiuser- und Multitaskingsystem
- netzwerkfähig
- stabil

## • Die Linux-Baum-Struktur

| | |
|---|---|
| `/` | die Wurzel (root) = das oberste Verzeichnis im Baum |
| `bin` | binary = einige grundlegende Programme wie cp, rm, grep usw. |
| `boot` | Dateien, die zum Hochfahren von Linux benötigt werden, wie der Kernel und die LILO-Dateien (Linux Loader) |
| `cdrom` | Mountpunkt (leeres Verzeichnis) für das CD-ROM; ab SuSE 7.2 nur noch ein symbolischer Link |
| `dev` | device = Gerätdateien, alle Festplatten-Partitionen ide u. scsi, COM-Ports, USB-Port, Maus, CD-ROM, Floppy usw. |
| `etc` | die meisten Konfigurationsdateien des Rechners und der Server-Programme |
| `floppy` | Mountpunkt (leeres Verzeichnis) für das Diskette-Laufwerk; ab SuSE 7.2 nur noch ein symbolischer Link |
| `home` | enthält alle Heimatverzeichnisse der regulären Benutzer |
| `lib` | library = enthält Programm-Bibliotheken und die Kernel-Modulen |
| `lost+found` | leeres Verzeichnis, das für die Reparatur einer Partition benötigt wird |
| `media` | nur ab SuSE 7.2, enthält die Mountpunkte für das CD-ROM, das Disketten-Laufwerk und externe USB/Firewire Festplatte. |
| `mnt` | leeres Verzeichnis, das für das Anhängen von fremden Partitionen (z.B. Windows) benutzt werden kann |
| `opt` | optional = bei SuSE werden hier kommerzielle Programme installiert, wie *netscape*, *staroffice*, *oracle* usw. aber auch die graphischen Oberflächen *kde* und *gnome* |
| `proc` | process = ein virtuelles Verzeichnis (existiert nicht auf der Festplatte und braucht dadurch keinen Platz!), es ist ein Zugang oder eine Schnittstelle zum Kernel, es erlaubt, Werte im Kernel anzuschauen und sogar zu verändern (Firewall, routing) |
| `root` | das Heimatverzeichnis des System-Administrators *root* |
| `sbin` | system binary = Programme, die nur als *root* ausführbar sind |
| `tmp` | temporary = Verzeichnis für temporäre Dateien |
| `usr` | user = ist das grösste Verzeichnis und enthält fast alle Programme, die Dokumentation, Programm-Bibliotheken und Programm-Quellen |
| `var` | variable = dieses Verzeichnis enthält Dateien die sich dauernd verändern und vergrössern wie z.B. Protokoll-Dateien, Email-Warteschlangen, Drucker-Warteschlangen, Proxy-Cache usw. |

## Linux herunterfahren und neu starten

Linux herunterfahren (nur als Administrator = root möglich):
`halt`
oder `shutdown -h now`
oder `init 0`
Linux neu starten (nur als Administrator = root möglich):
`reboot`
oder `shutdown -r now`
oder `<Str><Alt><Entf>` (auch als regulärer Benutzer möglich)

## Text-Konsole wechseln

ohne graphische Oberfläche: `<Alt>F1` bis `<Alt>F6`
mit graphischer Oberfläche / X Window : `<Str><Alt>F1` bis `<Str><Alt>F6`
zurück zur graphischen Oberfläche / X Window: `<Alt>F7`

## X-Server (graphische Oberfläche) beenden

**(aber nur wenn ein Problem aufgetreten ist!):**
`<Str><Alt><Backspace>`

## Besondere Tastatur-Kombinationen in der Bash-Shell

einen Befehl zurückholen: `<↑>` (Pfeil nach oben)
einen spezifischen Befehl zurückholen:
`<Str>r` erste Buchstaben vom Befehl eingeben,
`<Tab>` um den Befehl zu ändern/zu editieren,
`<Return>` wenn der Befehl schon richtig war.
Cursor Positionierung auf der ersten Kolonne: `<Str>a` oder `<Pos1>`
auf der letzten Kolonne: `<Str>e` oder `<Ende>`
ein Programm normal beenden: `exit` oder `<Str>d`
ein Programm unterbrechen: `<Str>c`

## Meta-Zeichen in der Shell

- Trennung zwischen Befehl, Optionen und Parametern: **Leerzeichen**
- beliebige Zeichen ersetzen (>= 0): ***** (wildcard oder Jokerzeichen)
  z.B. `ls /etc/XF*` für die Datei XF86Config im Verzeichnis `/etc` sehen
- einen einzigen Zeichen ersetzen: **?**
- eine Einheit bilden aus einer Zeichenkette die Leerzeichen enthält: **""** oder **''**
  z.B. `xterm -sb -rightbar -T "Das ist ein X-Terminal"`
- Übersetzung für Umgebungsvariablen: **$**
  z.B. `echo HOSTNAME = HOSTNAME`
  `echo $HOSTNAME =` *`Name des Rechners`*
- Übersetzung von Befehlen: **``**
  z.B. `echo hostname -i` 僚 `hostname -i`
  `echo \`hostname -i\'` 僚 *`IP-Adresse_des_Rechners`*
- ein Programm in den Hintergrund schicken: **&** am Ende des Befehls

z.B. `xterm &, emacs &` usw.

- ## Der Prompt-Aufbau (bei SuSE)

  als Benutzer: *benutzer@RECHNER: Pfad* **>**
  als Administrator root: *RECHNER: Pfad* #
  oder `bash-2.05 #` (wenn startx/KDE als root gestartet wurde)

- ## Benutzung der Maus unter dem X Window System

  - Eine Zeichenkette selektieren: linke Maustaste + Bewegung
    oder Doppel-Click mit der linken Maustaste
  - Eine Zeichenkette einfügen: mittlere Maustaste oder Maus-Rad oder bei einer Zwei-Tasten-Maus: die zwei Tasten gleichzeitig drücken
  - scrollen mit einer X-Anwendung: mittlere Maustaste oder Maus-Rad oder bei einer Zwei-Tasten-Maus: die zwei Tasten gleichzeitig drücken
  - Schriftgröße in einem X-Term ändern: <Str>rechte Maustaste

- ## Die wichtigsten ersten Befehle

  **pwd** (print working directory) zeigt das aktuelle Verzeichnis

  Verzeichnis wechseln:
  Absolut: **cd /**[*Pfad*][*Verzeichnis*]
  Relativ: **cd** [*Pfad*][*Verzeichnis*]
  zum persönlichen Verzeichnis `/home/Benutzername` oder **~** :
  `cd` (ohne Parameter)

  Dateien in einem Verzeichnis anschauen:
  **ls**
  `ls` [*Pfad*][*Dateiname*]
  `ls -l` (langes Format)
  `ls -la` (langes Format & versteckte Dateien = Dateien die mit einem **.** anfangen)
  `ls -ltr` (langes Format, nach Zeit sortiert, umgekehrt: die letzte veränderte Datei wird als letzte Datei angezeigt)
  `ls -li` (langes Format & inode)
  `ls -lh` (langes Format, Dateigrössen werden in KB, MB, GB angezeigt)
  `ls -ld` (langes Format, die Verzeichnisse werden angezeigt statt deren Inhalten)
  `ls -R` (rekursiv: zeigt alle Unterverzeichnisse und deren Inhalte)

  Benutzer-Identität wechseln:
  `su [-] [Benutzer]` (switch oder substitute user)
  - = kompletter Login, Identität + Pfad ändern, alle Umgebungsvariable werden übernommen.
  `su` Wenn kein Benutzer angegeben wird, dann wird automatisch root genommen
  `whoami` Unter welchem Benutzer (Konto) arbeite ich?

## • die Online-Hilfe unter Linux

**man** [*Abschnitt*] *Befehl* (man steht für manual)

z.B. `man ls`
`man 1 passwd`
`man 5 passwd`

`man` sections (Abschnitte):

1 Ausführbare Programme oder Shellbefehle
2 Systemaufrufe (Kernelfunktionen)
3 Bibliotheksaufrufe (Funktionen in System-Bibliotheken)
4 Spezielle Dateien (gewöhnlich in `/dev`)
5 Dateiformate und Konventionen, z. B. `/etc/passwd`
6 Spiele
7 Makropakete und Konventionen, z. B. `man`(7), `groff`(7)
8 Systemadministrationsbefehle (in der Regel nur für root)
9 Kernelroutinen [Nicht Standard]

Manpages ausdrucken: `man -t ls | lpr` (mit CUPS `lp` statt `lpr`)

`xman` ⇨ die graphische Variante von man

`whatis` *Befehl* kurze Erklärung des Befehls
z.B. `whatis passwd` ⇨ :passwd (1) und passwd (5)

**apropos** *Befehl/Stichwort* zeigt alle man-pages-Kopfzeilen, die dieses Stichwort enthalten

z.B. `apropos quota`
`apropos tcp`
`apropos -e tcp -e ip`

**info**, `xinfo` info (und `xinfo`, die graphische Variante) sollte der Nachfolger von man werden, mit Links fast wie unter HTML. Es leider im Vergleich zu HTML viel zu kompliziert.

**`info` Befehle:**

`d` directory = Hauptverzeichnis
`m` menu *Befehl* = eine Hilfeseite (Knoten) anzeigen
`h` help = Hilfe
`b` begin of node = Anfang der Seite / Knoten
`e` end of node = Ende der Seite / Knoten
`<Leertaste>` die Seite vorwärts "scrollen", wie `<Bild ↓>`
`<Backspace>` die Seite rückwärts "scrollen", wie `<Bild ↑>`
`s` search = Suche innerhalb der Seite
`u` up node = springt zum Knoten, der hierarchisch eine Ebene höher ist
`n` next node = springt zum nächsten Knoten auf der gleichen Ebene
`p` previous node = springt zum letzten Knoten zurück, auf der gleichen Ebene
`l` last text displayed = springt zum letzten besuchten Knoten, unabhängig von der Ebene
`q` quit = info beenden.

**HOWTOs**
Howtos sind längere Dokumente (100 Seiten und mehr) über ein Thema, wie z.B. Drucker, Soundkarten, ISDN, Firewalls usw. Die meisten Dokumente sind auf englisch, manche sind auch ins Deutsche übersetzt. Sie sind mit **gzip** komprimiert, können aber trotzdem direkt mit **less** angeschaut werden (mindestens unter SuSE). Es gibt auch mini-howtows (viel weniger Seiten).
Inzwischen gibt es die meisten Howtos in HTML-Format.

bei SuSE:
Die englischen Howtos befinden sich unter **/usr/share/doc/howto/en**
Die englischen HTML-Howtos befinden sich unter **/usr/share/doc/howto/en/html**
Die englischen mini-Howtos befinden sich unter **/usr/share/doc/howto/en/mini**

Die deutschen Howtos befinden sich unter **/usr/share/doc/howto/de**
Die deutschen HTML-Howtos befinden sich unter **/usr/share/doc/howto/de/html**
Die deutschen mini-Howtos befinden sich unter **/usr/share/doc/howto/de/mini**

**Achtung!** Unter SuSE 6.4 und manchen andere Distributionen befinden sich die Howtos unter `/usr/doc` statt `/usr/share/doc`

**Paket-Dokumentation:**
Fast jedes Programm besitzt noch zusätzliche Dokumentationen, die sich bei SuSE unter **`/usr/share/doc/packages/`** befindeten.

**SuSE-Hilfe auf dem Rechner:** `susehelpcenter`
oder wenn `apache` installiert ist mit einem Browser `http://localhost`

**SuSE-Hilfe im Internet:** `http://www.suse.de` (Supportdatenbank)

**Hilfe im Internet:** Die "beste" Suschmachine `http://www.google.com`
oder Linux-spezifisch: `http://www.google.com/linux`

## • Befehle mit Dateien und Verzeichnissen

Verzeichnis erstellen

**mkdir** [*Pfad*][*Verzeichnisname*]
mkdir -p [*Pfad*][*Verzeichnisname/Unterverzeichnisname*]

Dateien kopieren

**cp** -p (--preserve) -R (rekursiv) -a (archive = -dpR) -v (verbose)
preserve = Zugriffsrechte, Eigentümer & Zeit werden beibehalten
rekursiv = alle Unterverzeichnisse & Dateien werden kopiert
archive = Dateien werden einschließlich ihrer Struktur kopiert
verbose = zeigt den Kopiervorgang an

cp *Quelldatei Zieldatei*
cp *Dateiname Dateiname*
cp *Verzeichnis+Dateiname Dateiname*
cp *Dateiname Verzeichnis+Dateiname*
cp *Verzeichnis+Dateiname Verzeichnis+Dateiname*
cp *Dateiname Verzeichnis* (der Name der Datei bleibt)
cp *Dateiname_1 Dateiname_2 Dateiname_n Verzeichnis/*
cp [*Verzeichnis*]*Dateiname* **.** (kopiert an den Ort, wo ich mich befinde)
cp *Dateiname*[24] *Ziel* (kopiert alle Dateien, die eine 2 oder eine 4 am Ende des Dateinamens haben)
cp *Dateiname*[!35] *Ziel*
(kopiert alle Dateien außer denen, die eine 3 oder eine 5 am Ende des Dateinamens haben)

Dateien verschieben

**mv** *Quelldatei Zieldatei*
mv *Dateiname Verzeichnis*

Dateien umbenennen

mv *Dateiname neuer-Dateiname*

Dateien verschieben & umbenennen

mv *Dateiname Verzeichnis+neuer-Dateiname*

Dateien löschen

**rm** *Dateiname*
rm -vi *Dateiname* (v=verbose, i=interactive fragt nach einer Bestätigiung)

Verzeichnis löschen

**rm -rv** *Verzeichnisname*
(das Verzeichnis muß nicht leer sein. Gefährlich!)

**rmdir** *Verzeichnisname* (das Verzeichnis muß leer sein)

## • Die Dateien-Zugriffsrechte ändern

| **user=Eigentümer** | **group=Gruppe** | **other=Andere** |
|---|---|---|
| **r**ead **w**rite e**x**ec | **r**ead **w**rite e**x**ec | **r**ead **w**rite e**x**ec |
| read=4 write=2 eXec=1 | read=4 write=2 eXec=1 | read=4 write=2 eXec=1 |

Dateityp: d = directory = Verzeichnis
- = reguläre Datei
l = symbolischer Link
c = zeichenorientiertes Gerät
z.B. ein Modem
b = blockorientiertes Gerät
z.B. eine Festplatte

| $2^2=4$ | $2^1=2$ | $2^0=1$ | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 2 |
| 0 | 1 | 1 | 3 |
| 1 | 0 | 0 | 4 |
| 1 | 0 | 1 | 5 |
| 1 | 1 | 0 | 6 |
| 1 | 1 | 1 | 7 |

read=lesen
write=schreiben
exec=für ein Programm = ausführbar
für ein Verzeichnis = erlaubt in diesem Verzeichnis zu wechseln (cd)

**Zusätzliche Bits:**
User-ID-Bit = 4 / s, GroupID-Bit = 2 / s, Sticky-Bit = 1 / t

**Zugriffsrechte ändern:**

**chmod** [-Rv] u+rwx,g+rwx,o+rwx *[Pfad]Dateiname*
+ zufügen, - wegnehmen, = setzen
-R = rekursiv, -v = verbose

`chmod a=rw` *Dateiname* (setzt lesen und schreiben für alle: a=u+g+o)
`chmod 751 (u=rwx,g=rx,o=x)` *Dateiname*
`chmod 4755` *Dateiname* (setzt der User-ID-Bit)
`chmod 755` *Dateiname* (entfernt wieder der User-ID-Bit)

**Eigentümer und/oder Gruppe wechseln:**

**chown** [-Rv] *neuerEigentümer.neueGruppe [Pfad]Dateiname*
= ändert der Eigentümer und die Gruppe
-R = rekursiv, -v = verbose

`chown` *neuerEigentümer Dateiname* = ändert nur der Eigentümer
`chown .`*neueGruppe Dateiname* = ändert nur die Gruppe
`chown` *neuerEigentümer. Dateiname* = ändert der Eigentümer und die Standardgruppe für diesen Eigentümer

z.B. `chown pierre. /Testverzeichnis`
oder `chown pierre.users /Testverzeichnis`
(was genau das gleiche wie vorher ist)

`chown .users /Testverzeichnis/Datei`

**Zugriffsrechtmasken:**
umask definiert mit welchen Zugriffsrechten Dateien und Verzeichnisse erstellt werden. `umask 022` wird z.B. in `~/.bashrc` geschrieben.
`umask 022` → `777 - 022 = 755` für Verzeichnisse
`umask 022` → `666 - 022 = 644` für reguläre Dateien

# Die Text-Editoren

## vi, vim, gvim und elvis

`vi Datei1 [Datei2]` öffnet Datei1 (und ev. auch Datei2)

| | |
|---|---|
| `i` | Insert Modus = Einfüge-Modus |
| `<Esc>` | Wechsel vom Insert-Modus zum Komando-Modus |
| `x` oder `<Entf>` | einen Zeichen löschen |
| `:wq [`*`Dateiname`*`]` | Write & Quit = beenden und speichern (auch möglich: `ZZ`) |
| `:q!` | Quit = beenden ohne zu speichern |
| `dd` | eine Zeile löschen |
| `yy` | eine Zeile kopieren |
| *n*`yy` | *n* Zeilen kopieren |
| `p` | eine kopierte oder gelöschte Zeile(n) einfügen |
| `P` | wie p, aber vor der Cursor-Position |
| `u` und `<Str>r` | Undo - Rückgängig, `<Str>r` = redo |
| `o` | Insert-Modus & neue Zeile einfügen (nach der Pos.) |
| `O` | Insert-Modus & neue Zeile einfügen (vor der Pos.) |
| `a` | Insert-Modus nach der Cursor-Position |
| `A` | Insert-Modus am Ende der Zeile |
| `J` | Zwei Zeilen zusammenfügen |
| `/`*`Suchmuster`* | nach dem Suchmuster suchen wie bei less |

`:1,$s/`*`text1/text2`*`/g` Ersetzt text1 mit text2 von Zeile 1 bis zur letzten Zeile ($)

| | |
|---|---|
| `w` und `b` | `w` = ein Wort vorwärts, b = ein Wort rückwärts |
| `0` und `$` | `0` = Anfang der Zeile, `$` = Ende der Zeile |

`:0` oder gg und `:$` oder `G` `:0` = Anfang des Dokuments, `:$` = Ende des Dokuments

| | |
|---|---|
| `:e Dateiname` | öffnet eine Datei |
| `:buffers` | zeigt all geöffnete Dateien |
| `:b[uffer]` *n* | öffnet den Puffer *n* im aktuellen Fenster (*n*=1 bis ....) |
| `:n` | öffnet den nächsten Puffer (Datei) |
| `:N` | öffnet der vorherigen Puffer (Datei) |
| `:set number` | zeigt die Zeilennummer an (zurück mit `:set nonumber`) |
| `:`*n* | geht zur Zeile *n* |
| `:sp[lit]` | teilt das Fenster in 2 |
| `<Str>wk, <Str>wj` | wechselt zum oberen Fenster, zum unteren Fenster |
| `<Str>ww` | wechselt zum anderen Fenster |
| `:q` | schließt das aktuelle Fenster |
| `:f` oder `<Str>g` | zeigt der Namen der aktuelle Datei an |
| `q`*a* | startet eine Macro im Register *a* aufzunehmen |
| `q` | beendet das Aufnehmen |
| `@`*a* oder *n*`@`*a* | führt die Macro *a* aus oder führt sie *n* mal aus |
| `set mouse=a` | erlaubt die Benutzung der Maus (unter X), "visual mode" |
| `set bs=2` | erlaubt die "normale" Benutzung von Backspace |
| `syntax on` | schaltet die Farben ein (Syntax abhängig) |
| map *Zeichen Befehle* | bindet ein Zeichen zu mehrere vi-Befehle |
| vmap | map im "visual mode" |

z.B. in der Datei `~/.vimrc`:

```
map  <Esc>#  0i# <Esc>j und map  <Esc>- 0xxj
vmap <Esc># :s/.*/# &/<CR> und vmap <Esc>- :s/^\# //<CR>
```

- **Andere Text-Editoren**

  joe, pico, xedit, nedit, kedit, kwrite und kate (kde), gedit (gnome) usw.

- ## Inhalt von Dateien anschauen

| | |
|---|---|
| **cat** *Dateiname* | zeigt der ganze Inhalt einer Datei |
| **cat** -n *Dateiname* | numeriert alle Zeilen |
| cat -b *Dateiname* | numeriert nur Zeilen die nicht leer sind |
| **tac** *Dateiname* | (zeigt die Datei rückwärts, cat ↔ tac) |
| **tail** *Dateiname* | zeigt die 10 letzten Zeilen einer Datei |
| tail -n *Zahl* oder<br>tail -*Zahl Dateiname* | zeigt die letzten "Zahl" Zeilen der Datei |
| tail +*Zeilennummer* | zeigt von der Zeile mit der Zeilennummer an |
| **tail -f** *Dateiname* | wird dauernd aktualisiert, beenden mit <Str>c<br>z.B. (nur als root)<br>tail -f /var/log/messages |
| **head** *Dateiname* | zeigt die ersten 10 Zeilen einer Datei |
| head -n *Zahl* oder<br>head -*Zahl Dateiname* | zeigt die ersten "Zahl" Zeilen der Datei |
| **less** *Dateiname* | zeigt Zeilenweise eine Datei an |

(Tastatur: *Lehrtaste*=Bild ⇓, b=Bild ⇑, v=Editor, /= Suche n=nächste Suche vorwärts, N=nächste Suche rückwärts, g oder < = Anfang des Dokuments, G oder > = Ende des Dokuments, q = beenden, F = dauern aktualisieren)

| | |
|---|---|
| less +F *Dateiname* | zeigt dauern die letzte Zeilen. |
| **nl** *Dateiname* | numeriert nur die Zeilen die nicht leer sind |
| nl -ba *Dateiname* | numeriert alle Zeilen |

- ## der Pipe-Mechanismus |

z.B.

```
ls -la | less
pstree | less
ps aux | grep Prozessname
ifconfig eth0 | grep "inet addr" | cut -d: -f2 | cut -d" " -f1
```

- ## Standard Ein- und Ausgabe und deren Umleitung

| | |
|---|---|
| **>** *Datei* (1>) | Umleitung der Standardausgabe=Bildschirm in einer Datei (stdout) |
| **>>** *Datei* | Umleitung der Standardausgabe + zufügen ans Ende der Datei |
| **2>** | Umleitung der Standardfehlerausgabe (stderr) |
| **&>** | Umleitung von stdout und stderr |
| **>** *Datei* **2>&1** | Umleitung von stdout und stderr |
| **<** *Datei* (0>) | Umleitung der Standardeingabe (stdin) =Tastatur in einer Datei |

z.B. ls -la > verzeichnis_liste.txt
ls -la /etc >> verzeichnis_liste.txt

| | |
|---|---|
| \| **tee** *Datei* | Umleitung der Standardausgabe + Anzeige am Bildschirm |

| **tee -a** *Datei* wie oben + ans Ende der Datei zufügen

z.B. `ls -l | tee Datei | less`

- ## Festplatten-Befehle

| | |
|---|---|
| `fdisk -l` | zeigt alle Partitionen an, nur als root ausführbar |
| `df` (disk free) | zeigt die angehängte Partitionen mit den Anhängepunkte an |
| `df -h` | wie oben, aber in KB / MB / GB dargestellt |
| `df -i` | ziegt die Anzahl der freue inodes an |
| `kdf` | wie oben, mit graphischer Darstellung unter KDE |
| `du` (disk usage) | zeigt in Byte wieviel Platz jede Datei nimmt |
| `du -sh` | wie oben, aber in KB / MB dargestellt |
| `du -sh` | zeigt nur die Summe, in KB / MB dargestellt |
| `kdu` | wie du, mit graphischer Darstellung unter KDE |
| `kdirstat` | ähnlich wie kdu, noch ein bisschen schöner |

- ## Hard- & Symbolische-Links

Hard-Link: **ln** *[Quellverzeichnis][Dateiname] [Pfad]Linkname*
Symbolischer-Link:
**ln -s** *[Quellverzeichnis][Dateiname] [Pfad]Linkname*
**ln -sf** *Quelle Linkname* `-f` = überschreibt einen existierenden Link

- ## Dateien suchen mit locate und find

**locate** durchsucht eine Datenbank (`/var/lib/locatedb`) nach einem Muster , ist dadurch sehr schnell.

`locate -i Muster` (`-i` = ignore klein und gross schreibung)

**find** *Pfad* `-name` *Datei* immer **""** wenn ein ***** mit find gebraucht wird!!
`find` *Pfad* `-iname` *Datei* ignoriert Groß- und Kleinschreibung
`find` *Pfad* `-type f` `f`= reguläre Dateien, `d` = Verzeichnis, `l`= symlinks
`find` *Pfad* `-user` *Benutzername*
sucht Dateien eines bestimmten Eigentümers
`find` *Pfad* `-maxdepth` *Zahl*
sucht nur eine bestimmte Verzeichnistiefe
`find` *Pfad* `-xdev` sucht nicht in Verzeichnissen von anderen Dateisystemen
`find` *Pfad* `-mmin` *-Minuten*
sucht Dateien, die in den letzten *"Minuten"* geändert/erstellt worden sind
`find` *Pfad* `-follow` folgt den symbolischen Links
`find` *Pfad* *-Suchoption* **`-exec`** `rm` *Befehl* **`{} \;`**
führt ein Befehl mit den gefundenen Dateien aus.

Beispiele:

```
find -type d -maxdepth 1 | sort
find  /etc -name "*XF*"
find  /opt/kde -maxdepth 2 -type f -name "*edit"
find  -follow -mmin -5
find -name "test*" -exec rm -i {} \;
```

- **Programme suchen**

  **which** *Programmname* (benutzt die Pfade, die in PATH definiert sind)
  **whereis** *Programmname* (benutzt vordefinierte Verzeichnisse)

- **Dateien durchsuchen mit grep**

  **grep** [-Option] *Suchmuster Datei*
  grep -w *Suchmuster Datei* words=nur ganzes Wort
  grep -i *Suchmuster Datei* ignoriert Groß- und Klienschreibung
  grep -c *Suchmuster Datei* count=zählt nur die Treffer
  grep -r *Suchmuster Datei* (recursiv) sucht durch alle Unterverzeichnisse
  grep -s ... (supress) unterdrückt die Fehlermeldungen
  grep -v *Suchmuster Datei*
  zeigt die Zeilen mit dem vorhanden Muster nicht an

- **Alle Dateien in einem Verzeichnisbaum nach einem Muster durchsuchen**

```
find / -maxdepth 2 -type f -xdev |xargs grep -wi "welcome to suse"
```

  als Benutzer: `find /etc` **`2> /dev/null`** `| xargs grep` **`-sw`** `pierre`

- **Befehlsvereinfachung durch aliases**

  **alias** zeigt alle definierten aliases an
  **alias *Aliasname*="*Befehl*"** definiert einen neuen alias
  **unalias Aliasname** löscht ein alias
  z.B. `alias mflop="mount /floppy ; ls /floppy"`
  systemweite aliases:
  (als root editieren) die Datei `/etc/profile.local`
  benutzerweite aliases:
  (als Benutzer editieren) die Datei **`~/.alias`**
  (**~** ist ein alias für `/home/`*`Benutzername`*)

- **Die Umgebungsvariablen**

  **env oder printenv** zeigt alle definierten Umgebungsvariablen
  **set** zeigt alle definierten Umgebungs- und Lokalevariablen
  printenv *VARIABLEN-Name* zeigt den Inhalt der Variablen
  echo $*VARIABLEN-Name* zeigt den Inhalt der Variablen
  **export** *VARIABLE*=Wert definiert und exportiert eine Umgebungsvariable
  **unset** *VARIABLEN-Name* löscht eine Variable

- **Die Bash-Konfigurationsdateien**

  systemweit definierte Dateien: für alle Benutzer gültig, kann nur als root editiert werden
  `/etc/motd` **M**essage **O**f **T**he **D**ay, wird nach dem Anmelden gezeigt
  `/etc/profile` solte mit Vorsicht verändert werden!

| | |
|---|---|
| `/etc/profile.local` | (nur bei SuSE) kann angelegt werden |
| `/etc/bashrc` | (bei RedHat und Debian) |
| `/etc/inputrc` | Definition von Tasten-Kombination |

benutzerweit definierte Dateien: nur für einen bestimmten Benutzer gültig

| | |
|---|---|
| `~/.bash_profile` | wird beim Anmelden (login) ausgeführt (RedHat, Debian) |
| `~/.bash_login` | wird beim Anmelden (login) ausgeführt |
| `~/.profile` | wird beim Anmelden (login) ausgeführt (SuSE) |
| `~/.bashrc` | (rc = run commands) wird bei einer "Interaktive-Shell" ausgeführt |
| `~/.bash_logout` | wird beim Abmelden ausgeführt |
| `~/.alias` | (nur bei SuSE) kann angelegt werden, wird durch `~/.bashrc` ausgeführt |
| `~/.inputrc` | Definition von Tasten-Kombination |
| `~/.initrc` | wird vor dem Start des X-Servers ausgeführt |

## • Befehle mit Prozessen

Prozesse anschauen **`ps`**, **`ps l`**, **`ps aux`** oder **`ps aux |less`**
**`ps aux |grep`** *`Prozessname`*
**`ps -fC`** *`Prozessname`*

Prozess beenden **`kill`** *`PID`* oder **`kill -SIGTERM`** *`PID`*
*`PID`* = Prozess-Identifikationsnummer
**`killall`** *`Prozessname`* beendet alle Prozesse mit demselben Namen

Prozess "abschiessen" **`kill -9`** *`PID`* oder **`kill -SIGKILL`** *`PID`*
Mehrere Prozesse mit dem gleichen Namen "abschiessen":
**`killall -9`** `Prozessname`

Prozess-Hierarchie anschauen:
**`pstree`** oder **`ps f`**

Prozesse, die die meiste Prozessorleistung benutzen, anschauen:
**`top`** (wird dauernd aktualisiert)

Programme unter kde(2): **`ksysguard`** & **`kpm`**

Speicherplatz & swap ansehen:
**`free`** oder auch **`cat /proc/meminfo`** oder **`xosview`**

## • Der Befehl "mount" und das automatische Anhängen von Geräten

**`mount -t`** *`Dateisystemtyp Quelle /Mount-Punkt`*
*(Gerätdatei) (Verzeichnis)*

z.B. `mount -t vfat /dev/hda /windows/C`
`mount -t iso9660 /dev/hdc /cdrom`

In der Datei **`/etc/fstab`** sind Informationen über die Dateisysteme. Sie steuert, welche Partitionen beim Hochfahren von Linux automatisch angehängt werden und erlaubt zusätzlich, dass ein regulärer Benutzer mit der "Kurzform von Mount" eine

Partition anhängt:

`mount` *`/Mount-Punkt`*

z.B. `mount /cdrom` oder `mount /media/cdrom` oder `mount /floppy` usw.

**Struktur von /etc/fstab:**

| 1. Spalte | 2. Spalte | 3. Spalte | 4. Spalte | 5. Spalte | 6. Spalte |
|---|---|---|---|---|---|
| Gerätdatei / Partition / Entferter Rechner | Mountpunkt | Dateisystem-typ | Mount-Optionen | für das Programm dump Werte 0 / 1 | fsck 1 root 2 ext2 Part. 0 kein Check |

Beispiele:

```
/dev/hda6  /           ext2   default             1    1
/dev/hda7  /home       ext2   default             1    2
/dev/hda1  /windows/C  vfat   default,umask=000   0    0
/dev/sda4  /mnt/zip    auto   noauto,user         0    0
```

Einige Mount-Optionen für die 4. Spalte:

| | |
|---|---|
| `auto` | wird beim Hochfahren automatisch angehängt |
| `noauto` | wird beim Hochfahren nicht automatisch angehängt |
| `user(s)` | kann von einem regulären Benutzer angehängt werden |
| `ro` | read only = nur Lese-Berechtigung |
| `rw` | read / write = Lese- und Schreib-Berechtigung |
| `umask=xxx` | Zugriffsrecht-Maske: 022⇨755 (Verz.), 022⇨644 (Dateien) |
| `default` | `rw,suid,dev,exec,auto,nouser,async` (`async`=gepuffert) |

**Tip:** Schreib-Berechtigung als Benutzer auf einer vfat Partition: `umask=000`

**mount** (ohne Parameter) zeigt alle angehängten Geräte und Partitionen

über einen entfernten Rechner mit NFS (NFS = Netzwerk-Dateisystem):
**`mount`** *Entfernter-Rechner***:/***Entferntes-Verzeichnis* */***Lokales-Verzeichnis*

## • Daten-Sicherung mit "tar"

**`cp -a`** (-a = archive) *Quell-Verzeichnis Ziel-Verzeichnis*

**`tar`** [option] *`ArchivDatei Dateien oder Verzeichnis`*
(`.tar` oder `.tar.gz` oder `.tgz`)

| | |
|---|---|
| **`tar xzvf`** *`ArchivDatei`* | x=extrahieren z=dekompriemieren v=wortreich f=Archivdatei |
| `tar czvf` *`ArchivDatei`* | c=erzeugen z=komprimieren |
| `tar tzf` *`ArchivDatei`* | t=zeigt den Inhalt des Archivs |
| `tar rvf` *`ArchivDatei Datei_n`* | r=fügt *`Datei_n`* zum Archiv hinzu |

Beispiele:

```
tar cvf archiv1.tar datei1 datei2
```
(erstellt `archiv1.tar` und fügt `datei1` & `datei2` dazu)

```
tar rvf  archiv1.tar datei3
```
(fügt `datei3` zu `archiv1.tar`)

`gzip archiv1.tar`
(komprimiert `archiv1.tar`, der Namen wird: `archiv1.tar.gz`)

`tar tzf archiv1.tar.gz`
(zeigt den Inhalt von archiv1.tar.gz)

`tar xzvf archiv1.tar.gz datei2`
(extrahiert `datei2` aus `archiv1.tar.gz`)

`tar --delete -vf archiv1.tar Pafd/datei3`
(löscht `datei3` von `archiv1.tar`)

`tar -M -cvf /dev/fd0 Verzeichnis`
(kreiert den Archiv auf mehrere Disketten)

`tar xjvf archiv1.tar.bz2`
(entpackt & dekomprimiert mit `bzip2`)

**`gzip -d`** oder **`gunzip`** `archiv1.tar.gz`
(dekomprimiert `archiv1.tar.gz`)

Weitere Archivierungs-Programme: `cpio` (a), `afio` (ap), `ananda` (n) und `tob`

| | |
|---|---|
| `afio` | ist ähnlich wie cpio aber mit Komprimierung -Z (gzip) |
| `tob` | (Tape Oriented Backup) |
| `amanda` | (Advanced Maryland Automated Network Disk Archiver) |
| `arkeia` | (mit graphische Oberfläche, kormezielles Produkt, Serie pay) |

`find . | cpio -o (--create) -v (--verbose) [-F Device oder Datei ]`
`cpio -i (--extract) -v -I Device oder Datei`
`cpio -t (--list)`
`cpio -p (--pass-through)`

z.B. `find /home | cpio -o > /dev/tape`
`cpio -iv < /dev/tape`
`cpio -itF Backupdatei`
(zeigt der Inhalt des Archivs *Backupdatei*)
`cpio -iF Backupdatei Dateiname`
(extrahiert *Dateiname* aus *Backupdatei*)
`find /home | cpio -pvd /backup`
(p=kopiert eine Dateistruktur,
d=kreiert die nötige Unterverzeichnisse)

- ## zip und unzip

zip & unzip sind kompatibel zu pkzip und pkunzip unter Windows.

**`zip`** `Zipdatei.zip Datei(en)`
packt alle Dateien in der Zipdatei.zip

`zip -r Zipdatei.zip Verzeichnis`
rekursiv: packt das Verzeichnis und alle Unterverz.

`zip -d Zipdatei.zip Datei`
löscht die Datei in der Zipdatei.zip

**`unzip -l`** `Zipdatei.zip` zeigt den inhalt der Zipdatei im kurzen Format an

`unzip -lv Zipdatei.zip` zeigt den inhalt der Zipdatei im langen Format an

`unzip Zipdatei.zip` extrahiert die ganze Zipdatei

`unzip Zipdatei.zip Datei`
extrahiert die Datei aus der Zipdatei

- **Software mit "rpm" installieren (ReadHat Package Manager)**

**`rpm -ivh`** *`Paketdatei`***`.rpm`** --install => installieren
`rpm` **`-ivh --force`** installiert neu auch wenn das Paket schon vorhanden ist
`rpm` **`-ivh --nodeps`** überprüft nicht die Abhängichkeiten
`rpm` **`-U`** *`Paketdatei`***`.rpm`** --update => neue Version installieren
`rpm` **`-e`** *`Paketname`* --erase => installiertes Paket löschen
`rpm` **`-qi`** *`Paketname`* --query => Informationen von einem installierten Paket abfragen
`rpm` **`-qpi`** *`Paketdatei`***`.rpm`** => Informationen von einem nicht installierten Paket abfragen
`rpm` **`-qpR`** *`Paketdatei`***`.rpm`** Paket-Abhängigkeiten ausgeben
`rpm` **`-qpl`** *`Paketdatei.`***`rpm`** Liste der Dateien aus der Paketdatei anzeigen
`rpm` **`-qli`** *`Paketname`* Liste der Dateien aus dem installierten Paket anzeigen
`rpm` **`-qa | sort | less`** Zeigt alle installierte Pakete
`rpm -qa | grep` *`Suchmuster`*
Zeigt alle installierte Pakete die das *Suchmuster* im Paketnamen enthalten.

- **Benutzer und Gruppenverwaltung**

`useradd -m` *`Benutzer`* `-m` = erstellt das Heimatsverzeichnis

z.B. `useradd -d /public/ftp -u 555 -s /bin/false ftpuser`
erstellt eine Benutzer `ftpuser` mit der UID 555, die Shell `/bin/false` und das Heimatverzeichnis `/public/ftp`

`usermod [-`*`Optionen`*`]` *`Benutzer`* ändert ein Benutzerkonto

`passwd -S` (Status), `-x` (Max days), `-w` (Warning), `-l` (lock=disable), `-u` (enable)

`userdel -r` *`Benutzer`* `-r` = löscht das Heimatsverzeichnis

`groupadd -g` *`gid Gruppenname`*

`groupdel` *`Gruppenname`*

`groupmod -g` *`gid`* `-n` *`Neuen-Gruppenname Gruppenname`*

`groupmod -A` *`Benutzer Gruppe`* fügt ein Benutzer in einer Gruppe ein

`groupmod -D` *`Benutzer Gruppe`* entfernt Benutzer aus einer Gruppe

## Anhang A: Was tun wenn ein Befehl nicht funktioniert?

Der Fehler hat meistens zu tun mit:

- Sie haben einen relativen Pfad statt einem absoluten Pfad eingegeben.
- Sie haben einen absoluten Pfad statt einem relativen Pfad eingegeben.
- Sie befinden sich in einem falschen Verzeichnis.
- Es fehlt ein Leerzeichen nach dem Befehl oder einer der Optionen.
- Es steht ein Leerzeichen zuviel zwischen dem Pfad und dem Dateinamen.
- Sie haben einen Tippfehler in einem Datei- oder Verzeichnis-Namen.

1. Die Eingabeanforderung (Prompt) kontrollieren:

   - Unter welchem Benutzer arbeite ich?
   - In welchem Pfad befinde ich mich?

   Beispiele:

| | | |
|---|---|---|
| `pierre@sirius:~ >` | Benutzer: | pierre |
| | Rechner: | sirius |
| | Pfad: | `~` ist das gleiche wie `/home/pierre` |
| `pierre@sun:/ >` | Benutzer: | pierre |
| | Rechner: | sun |
| | Pfad: | `/` (also die Wurzel) |
| `sirius:~ #` | Benutzer: | root (wegen dem #) |
| | Rechner: | sirius |
| | Pfad: | `~` ist das gleiche wie `/root` |
| `root@sun:/home/pierre >` | Benutzer: | root |
| | Rechner: | sun |
| | Pfad: | `/home/pierre` |

   Zusätzlich kann mit pwd der Pfad kontrolliert werden und mit `whoami` der Benutzer (Loginname)!

   Beispiele mit Fehlern:

   pierre@sirius:/ > `cp .bashrc testdatei1`
   `cp: cannot stat `.bashrc': Datei oder Verzeichnis nicht gefunden`

   Erklärung: Das Verzeichnis, in dem ich mich gerade befinde, ist die Wurzel "/". Die Datei `.bashrc` befindet sich aber im Heimatverzeichnis "~" und nicht in der Wurzel.

   pierre@sirius:/ > `cp /etc/resolv.conf .`
   `cp: cannot create regular file `./resolv.conf': Keine Berechtigung`

   Erklärung: ich versuche, eine Datei ins aktuelle Verzeichnis zu kopieren. Das aktuelle Verzeichnis ist aber die Wurzel. **Nur `root` kann ausserhalb des Heimat-verzeichnisses schreiben!**

2. Die Syntax des Befehls kontrollieren:

- Steht ein Leerzeichen nach dem Befehl und nach der Option?
- Steht kein Leerzeichen zwischen dem Pfad und dem Dateinamen?

Beispiele:

```
pierre@sirius:~ > ls-la /etc
bash: ls-la: command not found
```

Erklärung: "command not found" bedeutet Befehl nicht gefunden. Den Befehl `ls` gibt es, aber nicht `ls-la`. Es fehlt das Leerzeichen nach dem `ls`.

```
pierre@sirius:~ > cp /etc/ hosts /tmp
cp: omitting directory `/etc/'
cp: cannot stat `hosts': Datei oder Verzeichnis nicht gefunden
```

Erklärung: es steht ein Leerzeichen zuviel zwischen dem Pfad und der Dateiname. Der Kopierbefehl `cp` versucht zuerst das Verzeichnis `/etc` nach `/tmp` zu kopieren, was ohne `-r` oder `-a` nicht geht, zweitens versucht `cp` die Datei `hosts` vom aktuellen Verzeichnis nach `/tmp` zu kopieren und findet sie nicht. Die richtige Syntax ist `cp /etc/hosts .`

3. Ist der Pfad relativ oder absolut?

Beispiel:

```
pierre@sirius:~ > cat etc/fstab
cat: etc/fstab: Datei oder Verzeichnis nicht gefunden
```

Erklärung: Der `/` vor dem `etc` wurde vergessen. So geschrieben ist es ein relativer Pfad, was das gleiche wie `cat /home/pierre/etc/fstab` ist. Diese Datei gibt es natürlich nicht, dafür gibt es die Datei `/etc/fstab`.

```
pierre@sirius:~ > cp /.bashrc  testdatei
cp: cannot stat `/.bashrc': Datei oder Verzeichnis nicht
gefunden
```

Erklärung: `/.bashrc` ist ein absoluter Pfad. Es gibt aber keine Datei `.bashrc` unter der Wurzel /, sondern nur `~/.bashrc`, was das gleiche wie `/home/pierre/.bashrc` ist.

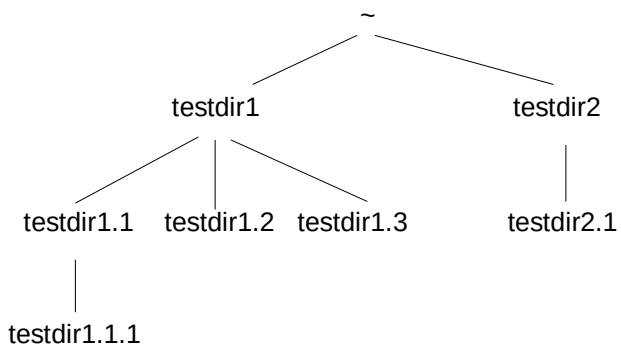
**Haben Sie den Fehler trotzdem nicht gefunden?**

Wenn Sie die Punkte 1 bis 3 sorgfältig überprüft haben und den Fehler nicht gefunden haben, dann können sie den Dozenten rufen.

# Anhang B: Übung mit Dateien und Verzeichnissen

- **Verzeichnis erstellen**

```
cd
mkdir testdir1
mkdir testdir1/testdir1.1
cd testdir1
ls
mkdir testdir1.2 testdir1.3
mkdir testdir1.1/testdir1.1.1
ls
ls testdir1.1
oder  ls testdir*
cd ..
pwd
mkdir -p testdir2/testdir2.1
ls testdir2
ls -R testdir1
```




- **Dateien kopieren**

```
cd
cp -v .bashrc testdir1/file1
ls testdir1
cp -v .bashrc testdir1/testdir1.1/file2
ls testdir1/testdir1.1
cd testdir1/testdir1.1
cp -v ~/.bashrc file3
ls
cp -v file3 file 4
ls
cp -v file3 ../../testdir2/file5
ls  ../../testdir2/
cp  -v file3 $HOME/testdir2/file6
ls  $HOME/testdir2  oder ls  ../../testdir2/
cp -v file3 ~/testdir2/file7
ls ~/testdir2  oder ls  ../../testdir2/
cp -v  file3   ~/testdir2  ! der Dateiname bleibt
ls ~/testdir2
cp -v file4    ~/testdir3
                ! Fehler: kreiert die Datei testdir3 statt in dem Verz. tesdir3
ls ~/testdir2
ls ~
cp -v file4  ~/testdir3/ ! Fehlermeldung: testdir3 kein Verzeichnis
cp -v file4  ~/testdir4/ ! Fehlermeldung: ...Verzeichnis nicht gefunden
cp -v file? ../testdir1.2  ! kopiert file2, file3 & file4
cp -v f* ../testdir1.3   ! mit verbose
cp -v file[23] ~/testdir2/testdir2.1/    ! kopiert nur file2 & file3
cp -v ~/testdir2/file[!67] testdir1.1.1 ! kopiert nur file3 & file5
cd .. (oder cd ~/testdir1)
cp -vr testdir1.1 testdir1.4  ! kopiert ein ganzes Verzeichnis Unterverzeichnis
```

- **Dateien verschieben & umbenennen**

```
cd testdir1.2
ls
mv -v file2 file8                          ! nur umbenennen
mv -v file3 file9
mv -v file8 ../testdir1.1/testdir1.1.1
                                           ! nur verschieben
mv -v file4 ../testdir1.1/testdir1.1.1/file10
                                           ! verschieben und umbenennen
ls
mv -v ../testdir1.3/* .
                                           ! alle Dateien von testdir1.3
                                           nach testdir1.2 verschieben
cd ~/testdir2

ls
mv -v file[!67] ~/testdir1/testdir1.3
                                           ! verschiebt nur file3 und file5
```

- **Dateien löschen**

```
ls testir2.1
rm -v   testdir2.1/file2
rm -vi  testdir2.1/file3  [y/n]
cd ~/testdir1/testdir1.3
ls
rm -vi  *
cd ../testdir1.2
ls
rm -v  file[24]     ! löscht file2 und file4
ls                  ! nur file3 und file9 bleiben noch
```

- **Verzeichnisse löschen**

```
cd  ../testdir1.1
ls
ls testdir1.1.1
rm -rv testdir1.1.1
cd ..
rmdir testdir1.1    ! Fehlermeldung: Das Verzeichnis ist nicht leer
rmdir ~/testdir2/testdir2.1
cd
pwd
rm -rv testdir1 testdir2 testdir3
                    (oder rm -rv testdir?
                     oder rm -rv testdir[123])
```