Schnellübersicht

Linux

Die praktische Referenz

UTE HERTZOG

Markt+Technik

1 Installation und Updates
2 Grundlagen
3 Dateien und Verzeichnisse
4 Hardwareverwaltung
5 Systemverwaltung
6 Netzwerkverwaltung
7 Shells und Shellskripte
8 Editoren und Werkzeuge
9 Konfigurationsdateien und Daemonen
10 Verschlüsselung und Signatur

3 Dateien und Verzeichnisse

3.1 Mit Dateien und Verzeichnissen arbeiten

3.1.1 Verzeichnisse anlegen

Der Befehl mkdir

Mit diesem Befehl erzeugen Sie neue Verzeichnisse.
`$ mkdir [-option(en)] verzeichnis(se)`
Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-p` oder `--parents`	Rekursives Anlegen von Verzeichnissen.
`-m` *rechte* oder `--mode=`*rechte*	Angabe der Zugriffsrechte für das anzulegende Verzeichnis
`-v`	Ausgabe von jedem angelegten Verzeichnis.

Tabelle 3.1: des Befehls `mkdir`

Der folgende Befehl erzeugt das Verzeichnis `projekt` unterhalb des Verzeichnisses `/`.
`$ mkdir /projekt`

►► **Info:** *Ein neu angelegtes Verzeichnis hat immer automatisch zwei Unterverzeichnisse: die Verzeichnisse `.` (das für das aktuelle Verzeichnis steht) und `..` (das für das übergeordnete Verzeichnis steht).* ◄◄

Die Option `-p` legt im folgenden Beispiel sowohl das Unterverzeichnis `auftrag2004` als auch das Verzeichnis `projekt` an:
`$ mkdir -p /projekt/auftrag2004`
Mithilfe der Option `-m` können Sie dem neu angelegten Verzeichnis die gewünschten Zugriffsrechte zuweisen.
`$ mkdir -m 755 /projekt`
Mit diesem Befehl erhält der Besitzer das Schreib-, Lese- und Ausführrecht und die Gruppe und die Anderen das Lese- und Ausführrecht.

►► **Info:** *Zugriffsrechte steuern, wer welche Befehle mit einer Datei oder einem Verzeichnis durchführen darf. Sie werden in Abschnitt 3.3.1 ausführlich behandelt.* ◄◄

3.1.2 Dateien anlegen bzw. aktualisieren

Dateien können unter Linux mit einem Editor oder Programm angelegt werden (vgl. Kapitel 8).

Der Befehl touch

Alternativ ist es möglich, leere Dateien mit dem Befehl `touch` zu erzeugen und das letzte Änderungs- oder Zugriffsdatum von bestehenden Dateien zu ändern.
`$ touch [-option(en)] [datum] datei(en)`

Eine neue Datei wird mit folgendem Befehl angelegt:

```
$ touch dat1
```

Die Datei `dat1` darf vorher nicht existiert haben, sonst wurde nur ihr letztes Änderungsdatum aktualisiert.

►► **Info:** *Unter Linux werden sowohl das Erstellungs- oder das letzte Änderungsdatum als auch das Zugriffsdatum mitprotokolliert. Das Erstellungs- bzw. letzte Änderungsdatum erhalten Sie mit dem Befehl* `ls -l` *angezeigt, während der Befehl* `ls -lu` *das letzte Zugriffsdatum ausgibt. Das Zugriffsdatum wird geändert, wenn auf eine Datei oder ein Verzeichnis zugegriffen wurde, ohne dass eine Änderung erfolgte. Das ist zum Beispiel der Fall, wenn eine Datei mit dem Befehl* `more` *angezeigt wird.* ◄◄

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-a` **oder** `--time=atime` **oder** `--time=access` **oder** `--time=use`	Änderung des Datums des letzten Zugriffs.
`-c` **oder** `--no-create`	Änderung nur bei bereits vorhandenen Dateien; es werden keine neuen Dateien erzeugt.
`-d` *zeit* **oder** `--date` *zeit*	Änderung des Datums der letzten Änderung. *zeit* kann in verschiedenen Formaten verwendet werden und kann Monatsnamen, am- und pm-Formate und weitere Angaben enthalten.
`-m` **oder** `--time=mtime` **oder** `--time=modify`	Änderung des Datums der letzten Änderung.
`-r` *datei* **oder** `--reference` *datei*	Änderung des Datums mithilfe einer Referenzdatei. Das Datum der Referenzdatei wird übertragen.
`-t` *zeit* `[--time=modify` **oder** `--time=mtime` **oder** `--time=atime` **oder** `--time=access` **oder** `--time=use]`	Angabe des Änderungs- bzw. des Zugriffszeitpunkts (Schlüsselwort `atime`, `access` oder `use`) im Format `[[CC]YY]MMDDhhmm[.SS]`. ■ C = Jahrhundert ■ Y = Jahr ■ M = Monat ■ D = Tag ■ h = Stunde ■ m = Minute ■ S = Sekunde

Tabelle 3.2: Optionen des Befehls `touch`

Das nachfolgende Beispiel zeigt, wie das Änderungsdatum einer Datei manipuliert wird:

```
$ touch -t 03030101 dat1
$ ls -l dat1
-rw-r--r--  1  root  other  65  Mar  3  01:01   dat1
```

3.1.3 Verzeichnisse wechseln

Der Befehl cd

Mit diesem Befehl können Sie in ein anderes Verzeichnis wechseln.
`$ cd [verzeichnis]`
Im nachfolgenden Beispiel wird in das Verzeichnis `/etc` gewechselt:
`$ cd /etc`
Wenn Sie den Befehl ohne Argument verwenden, wechseln Sie in Ihr Homeverzeichnis. Wenn Sie als *root* angemeldet sind, wechseln Sie in das Verzeichnis `/root`.
Mit dem Befehl
`$ cd ..`
wechseln Sie in das darüber liegende Verzeichnis.
Sie können als Argument des Befehls entweder einen relativen oder einen absoluten Pfadnamen verwenden. Der relative Pfadname geht von dem Verzeichnis aus, indem Sie sich im Moment befinden. Im nachfolgenden Beispiel wechseln Sie mit einer relativen Pfadangabe vom Verzeichnis `/home/her/berichte` in das Verzeichnis `/home/her/grafik`:
`$ cd ../grafik`
Alternativ können Sie auch mit der absoluten Pfadangabe in das Verzeichnis wechseln, wobei der absolute Pfad in diesem Beispiel mehr Schreibaufwand erfordert:
`$ cd /home/her/grafik`
Der absolute Pfad beginnt immer ab dem Wurzelverzeichnis `/` und ist völlig unabhängig von der aktuellen Position im Verzeichnisbaum.
Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-L`	Erzwingt, dass symbolischen Links gefolgt wird.
`-P`	Die physikalische Verzeichnisstruktur wird verwendet, anstatt symbolischen Links zu folgen.

Tabelle 3.3: Optionen des Befehls `cd`

3.1.4 Verzeichnisse auflisten

Der Befehl pwd

Dieser Befehl gibt den Namen des Verzeichnisses aus, in dem sich der Benutzer im Moment befindet:
`$ pwd`
`/usr/bin`

Der Befehl ls

Dieser Befehl listet den Inhalt von Verzeichnissen auf.
`$ ls [-option(en)] datei(en) | verzeichnis(se)`

Wenn Sie kein Argument verwenden, wird der Inhalt des aktuellen Verzeichnisses angezeigt:

```
$ ls -l
total 8
-rw-r-----   1 root  other  146 Jul  8 11:28 bericht
drwxr-xr-x   2 root  other  512 Jul  8 11:28 seminar
```

Die Option `-l` bewirkt ein ausführliches Auflisten des Verzeichnisinhalts mit folgenden Informationen:

- Das erste Zeichen der ersten Spalte gibt den Dateitypen aus:
  - `-` bedeutet, dass es sich um eine normale Datei handelt: zum Beispiel entweder eine ASCII-Text-Datei oder eine Applikationsdatei oder eine binäre Datei.
  - `d` bedeutet, dass es sich um ein Verzeichnis handelt.
  - `l` bedeutet, dass es sich um einen symbolischen Link handelt (vgl. Abschnitt 3.2.2).
  - `b` bedeutet, dass es sich um eine blockorientierte Gerätedatei handelt.
  - `c` bedeutet, dass es sich um eine zeichenorientierte Gerätedatei handelt.
  - `p` bedeutet, dass es sich um eine spezielle Named Pipe-Datei (FIFO) für die Interprozesskommunikation handelt.
  - `s` bedeutet, dass es sich um eine Socket-Datei für die Interprozesskommunikation handelt.

▶▶ **Info:** *Block- und zeichenorientierte Gerätedateien werden in Kapitel 4 ausführlich beschrieben.* ◀◀

- Der restliche Teil der ersten Spalte enthält die Zugriffsberechtigungen. Diese werden in Abschnitt 3.3 ausführlich behandelt.
- Die dritte Spalte enthält den so genannten Link Counter. Dieser gibt bei Dateien die Anzahl der Hard Links (vgl. Abschnitt 3.2.1) und bei Verzeichnissen die Anzahl der Unterverzeichnisse zurück.
- Die vierte Spalte enthält den Besitzer der Datei.
- Die fünfte Spalte gibt die Gruppe an, die Rechte an der Datei hat.
- Die sechste Spalte enthält die Dateigröße in Bytes: Eine Datei belegt immer mindestens zwei Blöcke zu 512 Bytes auf der Festplatte, auch wenn ihre tatsächlich angezeigte Größe kleiner ist, wie zum Beispiel bei der Datei `obst`. Die erste Ausgabe total 8 des Befehls ls –l zeigt die Anzahl der durch den Verzeichnisinhalt belegten Festplattenblöcke (zu 512 Bytes) an (vgl. Kapitel 9).

▶▶ **Info:** *Gerätedateien zeigen anstelle einer Größe in Bytes zwei Zahlen an: die Major und Minor Device Number (vgl. Kapitel 4).* ◀◀

- Die nächsten Spalten zeigen das Datum und die Uhrzeit der Erstellung bzw. der letzten Änderung der Datei oder des Verzeichnisses an. Wenn die Datei älter als ein halbes Jahr ist, wird anstelle der Uhrzeit die Jahreszahl ausgegeben.
- Die letzte Spalte enthält den Namen der Datei oder des Verzeichnisses.

▶▶ **Info:** *Unter Linux wird alles als Datei betrachtet. Verzeichnisse, Gerätedateien, symbolische Links sind nur Sonderformen von Dateien. Selbst Prozesse werden im Verzeichnis* `/proc` *als Verzeichnisse oder Dateien dargestellt und teilweise auch dort verwaltet (vgl. Kapitel 5).* ◀◀

Weitere wichtige Optionen des Befehls `ls` sind:

Option	Bedeutung
`-1` oder `--format=single-column`	Formatierte Ausgabe: Jeder Dateiname wird in einer eigenen Zeile angezeigt.
`-a` oder `--all`	Auch versteckte (mit einem Punkt beginnenden) Dateien anzeigen.
`-A` oder `--almost-all`	Auch versteckte Dateien anzeigen, mit Ausnahme der Verzeichnisse . und ..
`--author`	Den Autor der Datei ausgeben.
`-B` oder `--ignore-backups`	Dateien, die mit dem Zeichen ~ enden, gelten als Backup-Dateien und werden nicht ausgegeben.
`-b` oder `--escape`	Nicht druckbare Zeichen als Oktalzahl anzeigen. Verwendbar zum Beispiel bei Dateinamen, die mit Umlauten von Rechnern mit einem anderen Zeichensatz erzeugt wurden.
`--block-size=`*`groesse`*	Blöcke einer bestimmten Größe anzeigen.
`-C` oder `--format=vertical`	Mehrspaltige Ausgabe.
`-c` oder `--sort=status` oder `--time=status`	Letzte Veränderung der Statusinformationen sortiert anzeigen (gemeinsam mit Option `-lt`). Statt des Schlüsselworts `status` kann auch das Schlüsselwort `ctime` verwendet werden.
`--color=[`*`wann`*`]`	Farbe zum Unterscheiden der Dateitypen zum Zeitpunkt ***wann*** verwenden; wann kann `never` (nie), `always` (immer) und `auto` (automatisch) sein.
`-D` oder `--dired`	Formatierte Ausgabe für den Modus `dired` des Editors `emacs`.
`-d` oder `--directory`	Ausgabe des Inhalts von Unterverzeichnissen unterdrücken und symbolischen Links nicht folgen.
`-F` oder `--classify` oder `--indicator-style=classify`	Kennung für den Dateityp hinter dem Dateinamen ausgeben: / steht für Verzeichnis, * für ausführbare Dateien, @ für symbolische Links, | für Pipe-Dateien, und = für Socket-Dateien.
`-f`	Ausgabe erfolgt nicht sortiert nach Dateinamen, sondern nach der physikalischen Position im Verzeichnis.
`--full-time`	Vollständige Ausgabe der Zeiten anstatt der standardmäßigen Abkürzungen.
`-G` oder `--no-group`	Keine Ausgabe des Gruppennamens.
`-H` oder `--dereference-command-line`	In der Befehlszeile angegebene symbolische Links werden verfolgt.
`--dereference-command-line-symlink-to-dir`	In der Befehlszeile angegebene und auf Verzeichnis weisende symbolische Links werden verfolgt.
`-h` oder `--human-readable`	Datei- und Verzeichnisgrößen werden in lesbarem Format ausgegeben, zum Beispiel 2K, 3M oder 1G.
`--si`	Entspricht der Ausgabe von `-h`, verwendet aber den Teiler 1000 statt 1024.
`-I` oder `--ignore=`*`muster`*	Keine Anzeige von Einträgen, die mit dem Muster *`muster`* übereinstimmen.
`-i` oder `--inode`	Inode-Nummer anzeigen.
`-L` oder `--dereference`	Anzeige der Eigenschaften der Originaldatei bei symbolischen Links.

Tabelle 3.4: Optionen des Befehls `ls`

Option	Bedeutung
`-l` oder `--format=long`	Ausführliches Listing.
`-m` oder `--format=comma`	Ausgabe von möglichst vielen Dateinamen in einer Zeile, durch Kommata getrennt.
`-n` oder `--numeric-uid-gid`	Dieselbe Ausgabe wie `-l`, aber statt Benutzer- und Gruppennamen werden die numerischen UIDs und GIDs angezeigt.
`-N` oder `--literal`	Anzeige von »rohen« Dateinamen, das heißt, Kontrollzeichen werden nicht besonders behandelt.
`-o`	Dieselbe Ausgabe wie `-l`, aber keine Anzeige der Gruppe.
`-p` oder `--file-type` oder `--indicator-style=file-type`	Kennung für den Dateityp hinter dem Dateinamen ausgeben: / steht für Verzeichnis, @ für symbolische Links, | für Pipe-Dateien, und = für Socket-Dateien.
`-q` oder `--hide-control-chars`	Nicht darstellbare Zeichen werden durch ein Fragezeichen ? abgebildet.
`--show-control-chars`	Nicht darstellbare Zeichen werden angezeigt, wie sie sind (standardmäßige Vorgabe, wenn das Programm nicht `ls` ist und die Ausgabe auf ein Terminal erfolgt).
`-Q` oder `--quote-name`	Dateinamen werden in doppelten Hochkommata eingefasst und nicht darstellbare Zeichen werden als Oktalzahl dargestellt.
`--quoting-style=`***wort***	Es werden die durch ***wort*** spezifizierten Anführungszeichen verwendet, zum Beispiel `literal`, `locale`, `shell`, `shell-always`, `c`, `escape` usw.
`-R`	Den Inhalt von Unterverzeichnissen rekursiv anzeigen.
`-r` oder `--reverse`	Anzeige mit umgekehrter Sortierreihenfolge.
`-S` oder `--sort=size`	Ausgabe nach Dateigröße sortiert.
`-s` oder `--size`	Die Größe der Dateien wird in Blöcken zu 512 Byte ausgegeben. Die Variable POSIXLY_CORRECT muss dann ebenfalls gesetzt sein.
`-T` oder `--tabsize=`*`spalten`*	Definiert alle Tabstops auf die Zeichenanzahl von *`spalten`* anstelle des Vorgabewerts 8.
`--time-style=`*`stil`*	Die Änderungs- bzw. Zugriffszeiten der aufgelisteten Dateien werden im angegebenen Stil *`stil`* angezeigt, zum Beispiel: `full-iso`, `iso`, `locale`, `posix-iso`, `+FORMAT`.
`-t` oder `--sort=time`	Sortierreihenfolge verändern: nicht alphabetisch sortieren, sondern nach Änderungsdatum.
`-U` oder `--sort=none`	Die Sortierung wird wie bei der Option `-f` unterdrückt, die Ausgabe erfolgt aber im erweiterten Format.
`-u` oder `--time=access` oder `--sort=access`	Datum des letzten Zugriffs auf die Datei anzeigen, anstelle des Änderungsdatums (gemeinsam mit Option `l`). Anstelle des Schlüsselworts `access` kann auch das Schlüsselwort `atime` oder `use` verwendet werden.
`-v` oder `--sort=version`	Sortierung nach Version.
`-w` oder `--width=`*`spalten`*	Verwendung der durch `spalten` spezifizierten Bildschirmbreite.
`-X` oder `--sort=extension`	Sortierung alphabetisch nach Dateinamenserweiterung.
`-x` oder `--format=across` oder `--format=horizontal`	Dateinamen werden in Zeilen statt in Spalten angezeigt.

Tabelle 3.4: Optionen des Befehls `ls` *(Forts.)*

►► **Info:** *Ein Inode ist eine Datenstruktur, die sämtliche Informationen zu einer Datei oder einem Verzeichnis enthält.* ◄◄

Nachfolgend wird der Inhalt des Verzeichnisses `/etc` ausführlich aufgelistet, dabei wird das Datum der letzten Änderung der Dateien und Verzeichnisse angezeigt:

```
$ ls -l /etc
insgesamt 1236
drwxr-xr-x 3 root   root     4096 2007-07-18 19:57 acpi
-rw-r--r-- 1 root   root     2803 2007-07-18 19:55 adduser.conf
-rw-r--r-- 1 root   root       46 2007-07-27 23:31 adjtime
-rw-r--r-- 1 root   root      195 2007-07-18 20:39 aliases
drwxr-xr-x 3 root   root     4096 2007-07-18 20:26 alsa
...
```

Im nächsten Beispiel wird statt dem letzten Änderungsdatum das letzte Zugriffsdatum angezeigt:

```
$ ls -lu /etc
insgesamt 1236
drwxr-xr-x 3 root   root     4096 2007-07-29 19:10 acpi
-rw-r--r-- 1 root   root     2803 2007-07-18 21:25 adduser.conf
-rw-r--r-- 1 root   root       46 2007-07-27 23:31 adjtime
-rw-r--r-- 1 root   root      195 2007-07-29 19:12 aliases
drwxr-xr-x 3 root   root     4096 2007-07-29 19:10 alsa
...
```

Nun wird statt dem letzten Änderungsdatum bzw. Zugriffsdatum das Datum der letzten Änderung der Statusinformationen (z.B. der Zugriffsrechte) angezeigt:

```
$ ls -lc /etc
insgesamt 1236
drwxr-xr-x 3 root   root     4096 2007-07-29 19:33acpi
-rw-r--r-- 1 root   root     2803 2007-07-18 22:25 adduser.conf
-rw-r--r-- 1 root   root       46 2007-06-27 21:32 adjtime
-rw-r--r-- 1 root   root      195 2007-07-29 19:15 aliases
drwxr-xr-x 3 root   root     4096 2007-07-29 14:22 alsa
...
```

Im folgenden Beispiel wird der Inhalt des Verzeichnisses `/etc` mit allen Angaben, einschließlich Inodenummern, aufgelistet:

```
$ ls -li /etc
insgesamt 1236
145600 drwxr-xr-x 3 root   root     4096 2007-07-18 19:57 acpi
145435 -rw-r--r-- 1 root   root     2803 2007-07-18 19:55 adduser.conf
45280 -rw-r--r-- 1 root   root       46 2007-07-27 23:31 adjtime
145455 -rw-r--r-- 1 root   root      195 2007-07-18 20:39 aliases
145887 drwxr-xr-x 3 root   root     4096 2007-07-18 20:26 alsa
...
```

Der Befehl lsattr

Dieser Befehl gibt die erweiterten Attribute von Dateien auf einem Linux-Dateisystem vom Typ `ext2` aus. Die Syntax des Befehls lautet:

```
$ lsattr [-option] datei(en)
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-a`	Listet alle Dateien in Verzeichnissen auf, einschließlich der mit einem Punkt beginnenden Dateien.
`-d`	Listet Verzeichnisse wie normale Dateien auf, anstelle deren Inhalt auszugeben.
`-R`	Listet Attribute von Verzeichnissen und deren Inhalte rekursiv auf.
`-v`	Listet die Version-/Generationsnummer von Dateien auf.

Tabelle 3.5: Optionen des Befehls `lsattr`

Im Beispiel wird der Inhalt des aktuellen Verzeichnisses mit der Generationsnummer der darin enthaltenen Dateien aufgelistet:

```
$ lsattr -v
1280237839          ./bericht.txt
185023843           ./test.txt
```

►► **Info:** *Die erweiterten Attribute wurden mit dem Dateisystemtyp* `ext2` *eingeführt, um mehr Merkmale über Dateien speichern zu können, in Standardsystemen aber nicht benutzt.* ◄◄

3.1.5 Dateien anzeigen

Der Befehl cat

Mit diesem Befehl können Sie sich den Inhalt von kleinen Dateien anzeigen lassen. Zur Anzeige von großen Dateien ist er nicht geeignet, da der Befehl die Datei auf einmal bis zum Ende ausgibt:

```
$ cat [-option(en)] datei(en)
```

Zum Beispiel:

```
$ cat prot10
```

Zusätzlich wird der Befehl verwendet, um während einer Batchverarbeitung durch ein Skript Dateien auszugeben, zum Beispiel auf eine Pipeline, und die Ausgabe weiter zu bearbeiten, zum Beispiel um Daten zu extrahieren oder umzusortieren. Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-A` oder `--show-all`	Anzeige aller Steuerzeichen und Ausgabe eines Dollarzeichens $ am Ende jeder Zeile und von Tabulatoren als ^I (entspricht `-vET`).
`-b` oder `--number-nonblank`	Ausgabe der Zeilennummer am Anfang jeder nicht leeren Zeile.
`-e`	Anzeige aller Steuerzeichen außer LF (Zeilenschaltung) und Tabulatoren und Ausgabe eines Dollarzeichens $ am Ende jeder Zeile (entspricht `-vE`).
`-E` oder `--show-ends`	Ausgabe eines Dollarzeichens $ am Ende jeder Zeile.
`-n` oder `--number`	Ausgabe der Zeilennummer am Anfang jeder Zeile.
`-s` oder `--squeeze-blank`	Ausgabe zusätzlicher Leerzeilen wird unterdrückt.
`-t`	Anzeige aller Steuerzeichen außer LF (Zeilenschaltung) und Tabulatoren und Anzeige von Tabulatoren als ^I (entspricht `-vT`).

Tabelle 3.6: Optionen des Befehls `cat`

Option	Bedeutung
`-T` oder `--show-tabs`	Anzeige von Tabulatoren als ^I.
`-u`	Wird ignoriert und ist nur aus Gründen der Abwärtskompatibilität mit Unix vorhanden.
`-v` oder `--show-nonprinting`	Anzeige aller Steuerzeichen außer LF (Zeilenschaltung) und Tabulatoren.

Tabelle 3.6: Optionen des Befehls `cat` *(Forts.)*

Eine weitere wichtige Einsatzmöglichkeit des Befehls ist das Aneinanderketten von Dateien:

```
$ cat prot1 prot2 prot3 > protgesamt
```

Hier werden die Dateien `prot1`, `prot2` und `prot3` auf die Datei `protgesamt` kopiert.

```
$ cat protneu >> protgesamt
```

Hier wird die Datei `protneu` an die Datei `protgesamt` angehängt.

```
$ cat > datneu
```

In diesem Beispiel wird eine neue Datei `datneu` aus der Standardeingabe erzeugt. Die Eingabe muss mit der Tastenkombination Strg + D beendet werden.

►► **Info:** *Enthält eine Datei ein Zeichen Strg + D (Sonderzeichen für »End of file«), dann beendet* `cat` *bei diesem Zeichen die Dateiausgabe. Der Befehl* `cat` *ist deshalb nicht zum Ausgeben binärer Dateien geeignet.* ◄◄

Der Befehl more

Dieser Befehl listet den Inhalt von Dateien bildschirmweise auf:

```
$ more [-option(en)] datei(en)
```

Zum Beispiel:

```
$ more prot1000
Protokoll der Besprechung vom 4.7.2003
Anwesende:
.....
---More---(15%)
```

Sie haben die Möglichkeit, den Befehl mithilfe von internen Befehlen, die auf `vi`-Befehlen basieren (vgl. Kapitel 8), zu steuern. Die Tabelle führt die wichtigsten internen Befehle auf:

Eingabe	Funktion
Leertaste	Eine Bildschirmseite weiterblättern.
b oder Strg b	Eine Bildschirmseite zurückblättern.
↵	Eine Zeile weiterblättern.
? oder h	Hilfe anzeigen.
Q oder q oder Abbruch	Den Befehl beenden.
seitenzahl	Anzahl von Seiten (*seitenzahl*) weiterblättern.
/suchbegriff	Nach Suchbegriff suchen und die dazugehörige Seite anzeigen.
f	Eine Bildschirmseite überspringen.
s	Eine Textzeile überspringen.

Tabelle 3.7: Interne Befehle des Befehls `more`

Eingabe	Funktion
=	Ausgabe der aktuellen Zeilennummer.
N	Letzte Suche wiederholen.
.	Letzten Befehl wiederholen.
V	Aufruf des Editors `vi` in der Datei an der aktuellen Zeile.

Tabelle 3.7: Interne Befehle des Befehls `more` *(Forts.)*

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-num` *zahl*	Stellt die Bildschirmgröße auf *zahl* Zeilen ein.
`-d` oder `--number-nonblank`	Die Hinweise »Press space to continue, 'q' to quit.« und »Press 'h' for instructions« werden angezeigt und kein Warnton ertönt, wenn eine falsche Taste gedrückt wird.
`-l`	Normalerweise wird Strg + L (Seitenvorschub) als Spezialzeichen behandelt und hinter jeder Zeile angehalten, die einen Seitenvorschub enthält. Diese Option verhindert dieses Verhalten.
`-f`	Zählen der Textzeilen anstelle der Bildschirmzeilen, das heißt lange Zeilen werden nicht umgebrochen.
`-p`	Es wird nicht gescrollt, stattdessen wird der Bildschirm mit jeder Seite neu aufgebaut.
`-c`	Es wird nicht gescrollt, stattdessen wird die Anzeige von oben her erneuert und jede vorhandene Zeile überschrieben.
`-s`	Zeigt mehrere Leerzeilen als eine einzige an.
`-u`	Unterdrückt das Unterstreichen von Zeichen.
`+/`*muster*	Definiert eine Zeichenkette, die gesucht wird, bevor eine Datei angezeigt wird.
`+`*zahl*	Anzeige wird ab der Zeile mit der Nummer *zahl* begonnen.

Tabelle 3.8: Optionen des Befehls `more`

Der Befehl less

Dieser Befehl listet den Inhalt von Dateien ebenfalls bildschirmweise auf. Er kennt aber mehr Funktionen als der Befehl `more`:

```
$ less [-option(en)] datei(en)
```

Zum Beispiel:

```
$ less prot10
Protokoll der Besprechung vom 4.7.2003
Anwesende:
.....
less prot10 1-32/128 (15%)
```

Sie haben die Möglichkeit, auch diesen Befehl mithilfe von internen Befehlen zu steuern. Viele der Befehle lassen sich auch mit Zahlen kombinieren, die dann für eine Zeilennummer, eine Datei usw. stehen. Die Tabelle führt die wichtigsten internen Befehle auf:

Eingabe	Funktion
H oder h	Hilfe anzeigen.
Q oder q oder Abbruch	Den Befehl beenden.
Befehle zum Navigieren	
Leertaste oder Strg V oder f oder Strg F	Um die spezifizierte Zeilenzahl weiterblättern, standardmäßig um eine Bildschirmseite.
b oder Strg V oder ESC v	Eine Bildschirmseite oder spezifizierte Anzahl von Zeilen zurückblättern.
w	Ein Fenster zurückblättern und *zahl* auf die Zeilenzahl des Fensters zurücksetzen.
ESC Leertaste	Ein Fenster weiterblättern, aber am Ende der Datei nicht anhalten.
d oder Strg D	Eine halbe Bildschirmseite weiterblättern und *zahl* auf die Zeilenzahl der halben Fensterhöhe zurücksetzen.
u oder Strg U	Eine halbe Bildschirmseite zurückblättern und *zahl* auf die Zeilenzahl der halben Fensterhöhe zurücksetzen.
Zahl z	Wie bei der Leertaste, blättert aber um die gewünschte Zeilenzahl *zahl* weiter, die dann auch für die Leertaste gilt.
↵ oder Strg E oder e oder j oder Strg N	Eine Zeile oder spezifizierte Anzahl von Zeilen weiterblättern.
y oder Strg Y oder k oder Strg K oder Strg P	Eine Zeile oder spezifizierte Anzahl von Zeilen zurückblättern.
ESC →	Eine halbe Bildschirmbreite oder *zahl* Spalten nach links.
ESC ←	Eine halbe Bildschirmbreite oder *zahl* Spalten nach rechts.
F	Unaufhörlich vorwärtsblättern, zum Beispiel zum Anzeigen von Protokolldateien, die ständig um Einträge ergänzt werden; entspricht dem Befehl `tail -f`.
r oder Strg R oder Strg L	Bildschirm neu aufbauen.
R	Bildschirm neu aufbauen und gepufferte Eingabe löschen.
Befehle zum Suchen	
/*suchbegriff*	Vorwärts nach Suchbegriff suchen.
?*suchbegriff*	Rückwärts nach Suchbegriff suchen.
n	Vorwärtssuche wiederholen.
N	Rückwärtssuche wiederholen.
ESC n	Vorwärtssuche über alle Dateien wiederholen.
ESC N	Rückwärtssuche über alle Dateien wiederholen.
ESC u	Markierung für gefundene Suchmuster aktivieren und deaktivieren.

Tabelle 3.9: Interne Befehle des Befehls `less`

Eingabe	Funktion
	Das Modifizieren von Suchmustern ist möglich durch:
	[ESC] [N] oder [!] — Nach nicht übereinstimmenden Zeilen suchen.
	[Strg] [E] oder [*] — Mehrere Dateien durchsuchen.
	[Strg] [F] oder [@] — Die Suche bei Verwendung von / bei der ersten und bei Verwendung von ? bei der letzten Datei beginnen.
	[Strg] [K] — Übereinstimmungen markieren, aber Position beibehalten.
	[Strg] [R] — Keine regulären Ausdrücke verwenden.
Befehle zum Springen	
zahl[g] oder [ESC] [<]	Springe auf die erste Zeile oder Zeile *zahl*.
zahl[G] oder [ESC] [<]	Springe auf die letzte Zeile oder Zeile *zahl*.
[p] **%**	Springe auf den Dateianfang oder auf *zahl* Prozent der Datei.
[t]	Springe zur nächsten Markierung.
[T]	Springe zur vorherigen Markierung.
[{] [(] [[]	Suche schließendes Klammerelement })].
[}] [)] []]	Suche öffnendes Klammerelement { ([.
[ESC] [Strg] [F]	Entspricht der Suche nach }, fordert aber zwei Zeichen, auf die sich die Suche beziehen soll.
[ESC] [Strg] [B]	Entspricht der Suche nach {, fordert aber zwei Zeichen, auf die sich die Suche beziehen soll.
[M] *buchstabe*	Markiert die aktuelle Position mit dem angegebenen Kleinbuchstaben.
['] *buchstabe*	Springt an die Position, die mit dem angegebenen Kleinbuchstaben markiert wurde.
[)]	Springt auf die vorherige Position oder die letzte Datei.
[']	Entspricht der Eingabe von [m], aber behält Position bei.
Befehle, um Dateien zu bearbeiten	
[:][e] *dateiname* oder [Strg] [X] [Strg] [V]	Datei *dateiname* in die Liste der zu bearbeitenden Dateien laden.
[:][n]	Nächste Datei in der Dateiliste lesen.
[:][p]	Vorherige Datei in der Dateiliste lesen.
[:][x]	Erste Datei in der Dateiliste lesen.
[:][d]	Aktuelle Datei aus der Dateiliste entfernen.
[:][f] oder [=] oder [Strg] [G]	Ausgabe von Dateiname, Position in Dateiliste, Zeilennummer der ersten Bildschirmzeile, Zeilengesamtzahl, Zeichennummer des ersten Bildschirmzeichens und Zeichengesamtzahl.
Weitere Befehle	
[!] *befehl*	Aufruf eines Shellbefehls, ohne die Anzeige zu beenden.
[+] *befehl*	Befehl wird jedes Mal ausgeführt, wenn eine neue Datei geladen wird.
[|] *markierung befehl*	Übergabe eines Dateiausschnitts von der ersten Bildschirmzeile bis zur Markierung an einen Befehl.

Tabelle 3.9: Interne Befehle des Befehls `less` *(Forts.)*

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-?` oder `--help`	Zeigt den Hilfebildschirm an.
`-a` oder `--search-skip-screen`	Aktiviert die Suche bei Suchbefehlen ab der letzten angezeigten und nicht ab der zweiten angezeigten Zeile.
`-A` oder `--mouse-support`	Funktioniert im Moment nur in einem `xterm`-Fenster und bewirkt, dass der Befehl Mausaktionen durchführt.
`-b`*anzahl* oder `--buffers=`*anzahl*	Definiert die Anzahl des Pufferspeichers für jede Datei in Kilobyte. Standardmäßig werden 64 Kilobyte pro Datei verwendet.
`-B` oder `--auto-buffers`	Das automatische Belegen von Puffern für Daten aus einer Pipe wird dadurch verhindert.
`-c` oder `--clear-screen`	Baut beim Blättern den Bildschirm vollständig neu von oben nach unten auf.
`-C` oder `--CLEAR-SCREEN`	Löscht beim Blättern zuerst den Bildschirm und baut ihn dann vollständig neu von oben nach unten auf.
`-d` oder `--dumb`	Unterdrückt die Ausgabe von Fehlermeldungen des Terminals.
`-D`*zeichen*`color` oder `--color=`*zeichen*`color`	Definiert die Farbe des angezeigten Textes, wobei *zeichen* folgende Werte annehmen kann: `n` = normal `s` = standout `d` = bold `u` = underlined `k` = blink
`-e` oder `--quit-at-eof`	Der Befehl wird beendet, wenn zweimal EOF erreicht wurde.
`-E` oder `--QUIT-AT-EOF`	Der Befehl wird beendet, wenn EOF erreicht wurde.
`-f` oder `--force`	Erzwingt das Öffnen von Verzeichnissen und Gerätedateien und unterdrückt die Warnmeldung beim Öffnen einer Binärdatei.
`-F` oder `--quit-if-one-screen`	Beendet den Befehl automatisch, wenn die vollständige Datei auf dem ersten Bildschirm angezeigt werden kann.
`-g` oder `--hilite-search`	Nur die vom letzten Suchbefehl gefundene Übereinstimmung wird markiert und nicht alle passenden Textstellen.
`-G` oder `--HILITE-SEARCH`	Das Markieren aller gefundenen Übereinstimmungen wird unterdrückt.
`-h`*anzahl* oder `--max-back-scroll=`*anzahl*	Nicht mehr als *anzahl* Zeilen werden auf einmal zurückgeblättert.
`-i` oder `--ignore-case`	Wenn das Suchmuster keine Großbuchstaben enthält, wird unabhängig von der Groß- und Kleinschreibung gesucht.
`-I` oder `--IGNORE-CASE`	Es wird generell unabhängig von der Groß- und Kleinschreibung gesucht.
`-j=` *anzahl* oder `--jump-target=`*anzahl*	Die Zielzeile einer Suche oder eines Sprungs wird auf der Bildschirmzeile mit der Nummer *anzahl* vom oberen Bildschirmrand aus positioniert.
`-J` oder `--status-column`	Gibt eine Statusspalte in der linken Bildschirmecke aus, die Zeilen anzeigt, die mit der aktuellen Suche übereinstimmen.
`-k`*datei* oder `--lesskey-file=`*datei*	Eine Datei wird eingelesen, die eine spezielle Tastenbelegung spezifiziert.

Tabelle 3.10: Optionen des Befehls `less`

Option	Bedeutung
-K*zeichensatz*	Zwingt den Befehl diesen Zeichensatz zu verwenden.
-m oder --long-prompt	Anzeige eines dem Befehl more entsprechenden Prompts.
-M oder --LONG-PROMPT	Anzeige eines ausführlicheren Prompts mit Prozentsatz, aktuellen Zeilennummern und Gesamtzeilenzahl.
-n oder --line-numbers	Unterdrückt Zeilennummern.
-N oder --LINE-NUMBERS	Ausgabe der aktuellen Zeilennummer vor jeder Zeile.
-o*datei* oder --log-file=*datei*	Ausgabe von Informationen aus einer Pipe werden nach *datei* und an den Bildschirm geschrieben.
-O*datei* oder --LOGFILE=*datei*	Entspricht der Option -o, überschreibt aber bereits vorhandenen Dateien ohne Rückfrage.
-p*suchmuster* oder --pattern=*suchmuster*	Beim Starten des Befehls wird nach der ersten Übereinstimmung mit *suchmuster* gesucht.
-P*prompt* oder --prompt=*prompt*	Definiert den Prompt: -m (kurzer Prompt), -M (langer Prompt) oder = (setzt den Prompt gleich der Angabe).
-q oder --quiet oder --silent	Deaktiviert den Warnton, der ertönt, wenn versucht wird, vor den Dateianfang oder hinter das Dateiende zu blättern.
-Q oder --QUIET oder --SILENT	Der Warnton wird generell unterdrückt.
-r oder --raw-control-chars	Steuerzeichen werden direkt angezeigt; dies kann zu Anzeigeproblemen führen.
-R oder --RAW-CONTROL-CHARs	Entspricht der Option -r, es wird aber versucht, die Anzeige so gut wie möglich darzustellen.
-s oder --squeeze-blank-lines	Aufeinanderfolgende Leerzeilen werden als eine Zeile ausgegeben.
-S oder --chop-long-lines	Lange Zeilen werden abgeschnitten, anstatt diese umzubrechen.
-t*tag* oder --tag=*tag*	Die Datei, die *tag* enthält, wird editiert; dazu wird die Datei ./tags verwendet.
-T*datei* oder --tag-file=*datei*	Anstatt die Datei ./tags zu verwenden, wird die angegebene Datei verwendet.
-u oder --underline-special	Backspaces und Zeilenrückläufe (carriage returns) werden als druckbare Zeichen interpretiert.
-U oder --UNDERLINE-SPECIAL	Backspaces und Zeilenrückläufe (carriage returns) werden als Steuerzeichen interpretiert.
-w oder --hilite-unread	Zeilen nach dem Dateiende werden als Leerzeilen anstatt mit einer Tilde ~ angezeigt.
-W oder --HILITE-UNREAD	Entspricht -w, aber markiert zeitweise die erste neue Zeile nach jedem Befehl, der mehr als eine Zeile weit springt.
-XXX	Markiert Zeichen, die dazu verwendet werden, falsche Zeichen darzustellen. Standardmäßig werden solche Zeichen binär dargestellt.
-x*anzahl* oder --tabs=*anzahl*	Definiert die Tabulatorstopps mit *anzahl* Zeichen; Standardwert ist 8.
-X oder --no-init	Die Initialisierungs- und Deinitialisierungssequenzen von Termcap an das Terminal werden unterdrückt.
--no-keypad	Die Initialisierungs- und Deinitialisierungssequenzen der Tastatur an das Terminal werden unterdrückt

Tabelle 3.10: Optionen des Befehls less *(Forts.)*

Option	Bedeutung
-y*anzahl* oder --max-forw-scroll=*anzahl*	Es wird nie mehr als *anzahl* Zeilen vorwärts geblättert.
-[z]*anzahl* oder --window=*anzahl*	Ändert die standardmäßige Scrollinggröße des Fensters auf *anzahl* Zeilen. Standard ist eine Bildschirmseite.

Tabelle 3.10: Optionen des Befehls `less` *(Forts.)*

Der Befehl `head`

Dieser Befehl gibt standardmäßig die ersten 10 Zeilen einer Datei aus.
`$ head [-option(en)] datei(en)`
Der Befehl kennt folgende Optionen:

Option	Bedeutung
-c*anzahl* oder --bytes=*anzahl*	Ausgabe der angegebenen Anzahl der ersten Bytes; nach Anzahl kann ein `b` für Blöcke à 512 Byte, `k` für 1 Kilobyte- oder `m` für 1 Megabyte-Blöcke verwendet werden.
-*anzahl* oder n*anzahl* oder --lines=*anzahl*	Ausgabe der ersten *anzahl* Zeilen; Vorgabewert ist 10.
-q oder --quiet oder --silent	Keine Ausgabe von Dateinamen als Kopfzeile.
-v oder --verbose	Immer Ausgabe einer Kopfzeile mit Dateinamen.

Tabelle 3.11: Optionen des Befehls `head`

Sie können zum Beispiel die Anzahl der auszugebenden Zeilen steuern, indem Sie diese als Option eingeben. Hier werden die ersten 12 Zeilen ausgegeben:
`$ head -12 prot1000`

Der Befehl tail

Dieser Befehl gibt standardmäßig die letzten 10 Zeilen einer Datei aus.
`$ tail [-option(en)] datei`
Der Befehl kennt folgende Optionen:

Option	Bedeutung
-c*anzahl* oder --bytes=*anzahl*	Ausgabe der angegebenen Anzahl der letzten Bytes; nach Anzahl kann ein `b` für Blöcke à 512 Byte, `k` für 1 Kilobyte- oder `m` für 1 Megabyte-Blöcke verwendet werden.
-f oder -F oder --follow	Das Programm bricht die Ausgabe nicht am Dateiende ab, sondern zeigt ständig neue Einträge einer wachsenden Datei an, zum Beispiel einer Protokolldatei. Die Anzeige kann mit Strg C beendet werden.
--retry	Ständiger Versuch eine Datei zu öffnen, die beim Starten noch nicht verfügbar ist, zum Beispiel um in einem Terminalfenster eine Protokolldatei auszugeben; nur mit der Option -f sinnvoll.
-*anzahl* oder n*anzahl* oder--lines=*anzahl*	Ausgabe der letzten *anzahl* Zeilen; Vorgabewert ist 10.

Tabelle 3.12: Optionen des Befehls `tail`

Option	Bedeutung
+*anzahl* oder +n*anzahl*	Ausgabe beginnt von der Zeile mit der Nummer *anzahl* vom Beginn der Datei an.
`--max-unchanged-stats=`*anzahl*	Wenn eine Datei nach *anzahl* Iterationen (Vorgabewert ist 5) unverändert ist, kann sie mit dieser Option und `--follow=name` nochmals geöffnet werden, um festzustellen, ob sie gelöscht oder umbenannt wurde. Dies kann zum Beispiel der Fall bei rotierenden Protokolldateien sein, die im laufenden Betrieb umkopiert und umbenannt werden, ohne dass der Filehandle, unter dem sie mit dem Befehl `tail` geöffnet wurden, aktualisiert wird. In dem Fall würde der Befehl ständig auf die veraltete Protokolldatei zugreifen.
`--pid=`*pid*	Wird gemeinsam mit -f verwendet: Das Programm soll beendet werden, wenn der Prozess *pid* beendet wird.
`-q` oder `--quiet` oder `--silent`	Keine Ausgabe von Dateinamen als Kopfzeile.
`-s` oder `--sleep-intervall=`*sek*	Wird gemeinsam mit -f verwendet: Das Programm soll zwischen den Versuchen, die Datei neu auszulesen, eine Pause von *sek* Sekunden machen (Vorgabe ist 1).
`-v` oder `--verbose`	Immer Ausgabe einer Kopfzeile mit Dateinamen.

Tabelle 3.12: Optionen des Befehls `tail` *(Forts.)*

Sie können die Anzahl der auszugebenden Zeilen steuern, indem Sie diese als Option eingeben. Hier werden die letzten 7 Zeilen ausgegeben:

```
$ tail -7 prot1000
```

►► **Tipp:** *Mithilfe der Option* `-f` *lassen sich Protokolldateien, deren neue Einträge am Ende eingefügt werden, sehr gut überwachen. Der Befehl endet mit dieser Option nicht am Dateiende, sondern wartet auf weitere, später angehängte Zeilen, die er dann anzeigt.* ◄◄

Der Befehl echo

Mithilfe dieses Befehls kann ein in der Befehlszeile angegebener Text auf dem Bildschirm bzw. der Inhalt von Verzeichnissen ausgegeben werden. Die Syntax des Befehls lautet:

```
$ echo [-option(en)] argumente
```

Wenn der Text Leerzeichen enthält, die für die Formatierung wichtig sind, ist es notwendig, den Text in Anführungszeichen (") oder Apostrophe (') zu fassen. Der Befehl wird häufig in Shellskripten angewandt, um mit einer Meldung den Stand der Bearbeitung anzuzeigen.

Der Befehl kennt folgende Optionen:

Option	Bedeutung	
-e	Aktiviert die Interpretation der durch einen Backslash eingeleiteten Escape-Sequenzen:	
	nnn	Ausgabe des ASCII-Zeichens mit dem Oktalwert *nnn*.
	\\	Backslash
	\a	Signalton
	\b	Backspace
	\c	Unterdrückt nachfolgende Zeilenschaltung
	\f	Seitenvorschub
	\n	Zeilenschaltung (Newline)
	\r	Zeilenende (Carriage return)
	\t	Horizontaler Tabulator
	\v	Vertikaler Tabulator
-E	Die Interpretation der Escape-Sequenzen wird deaktiviert.	
-n	Die abschließende Zeilenschaltung wird nicht ausgegeben.	

Tabelle 3.13: Optionen des Befehls `echo`

Zum Beispiel:

```
$ echo "Dies ist ein     Text\n" > dat.neu
```

Der Befehl tac

Dieser Befehl ist die umgekehrte Schreibweise des Befehls `cat` und listet die Zeilen einer Datei rückwärts auf. Der Befehl ist feldorientiert, wobei das standardmäßige Feldtrennzeichen das Zeilenende ist. Die Syntax des Befehls lautet:

```
$ tac [-option(en)] datei(en)
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
-b oder --before	Definiert, dass das Feldtrennzeichen vor anstelle nach dem Feld steht.
-r oder --regex	Das Feldtrennzeichen wird als regulärer Ausdruck interpretiert.
-s *trenn* oder --separator=*trenn*	Das Feldtrennzeichen ist *trenn* anstelle des Zeilenendes.

Tabelle 3.14: Optionen des Befehls `tac`

Hier wird die Datei `/etc/passwd` zeilenweise rückwärts ausgegeben:

```
$ tac /etc/passwd
privoxy:x:102:503:Daemon user for privoxy:/var/lib/privoxy:/bin/false
radiusd:x:101:502:Radius daemon:/var/lib/radiusd:/bin/false
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
news:x:9:13:News system:/etc/news:/bin/bash
man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash
ohauptmann:x:507:100::/home/ohauptmann:/bin/bash
hpiffke:x:505:100::/home/hpiffke:/bin/bash
...
```

Der Befehl tee

Dieser Befehl kann die Ausgabe eines Befehls zusätzlich zur Datenumlenkung speichern oder anzeigen. Er liest von der Standardeingabe ein, verdoppelt den Datenstrom und gibt jeweils auf die Standardausgabe und in die angegebene Datei aus. Es kann also der in einer Pipeline verarbeitete Datenstrom abgegriffen werden, ohne dass die Pipeline unterbrochen wird. Die Syntax des Befehls lautet:
`$ tee [-option(en)] datei(en)`
Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-a` oder `--append`	Die Dateien werden an die angegebene(n) Datei(en) angehängt, anstatt diese zu überschreiben.
`-i` oder `--ignore-interrupts`	Interrupt-Signale werden ignoriert.

Tabelle 3.15: Optionen des Befehls `tee`

In diesem Beispiel wird die Ausgabe des Befehls `ls` auf ihre Zeilenanzahl überprüft und gleichzeitig in die Datei `ls-liste` geleitet:
`$ ls | tee ls-liste | wc-l`
Wird die Datei verändert, ändert sich auch die Prüfsumme und gegebenenfalls die Größe in Blöcken.

3.1.6 Dateien und Verzeichnisse löschen

Der Befehl rmdir

Mit diesem Befehl können Sie ein leeres Verzeichnis löschen:
`$ rmdir [-option(en)] verzeichnis(se)`
Zum Beispiel:
`$ rmdir /projekt`
Ist das Verzeichnis nicht leer, wird der Befehl nicht ausgeführt und Sie erhalten eine entsprechende Fehlermeldung:
`rmdir: "/projekt": Directory not empty`

►► **Tipp:** *Verzeichnisse, die nicht leer sind, können Sie mit dem Befehl* `rm` *löschen.* ◄◄

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-ignore-fail-on-non-empty`	Wenn das Verzeichnis nicht leer ist, wird die Fehlermeldung unterdrückt. Nützlich für Stapelverarbeitungen.
`-P` oder `--parents`	Rekursives Entfernen von ineinander geschachtelten, leeren Verzeichnissen: Zuerst wird das unterste Verzeichnis entfernt, dann das darüber liegende usw. Der Befehl `rmdir -p x/y/z` bewirkt dasselbe wie `rmdir x/y/z x/y x`.
`-v` oder `--verbose`	Ausführliche Ausgabe für jedes bearbeitete Verzeichnis.

Tabelle 3.16: Optionen des Befehls `rmdir`

Der Befehl rm

Mit diesem Befehl löschen Sie Dateien oder Verzeichnisse.

```
$ rm [-option(en)] datei(en) | verzeichnis(se)
```

Im folgenden Beispiel werden mehrere Dateien mit diesem Befehl gelöscht:

```
$ rm dat1 dat2 dat3
```

►► **Vorsicht:** *Der Befehl löscht die Dateien normalerweise sofort ohne Rückfrage. Unter Linux ist es standardmäßig nicht möglich, einen Löschvorgang wieder rückgängig zu machen. In diesem Fall könnten Sie nur noch auf die letzte Datensicherung zurückgreifen (vgl. Kapitel 5).* ◄◄

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-d` oder `--directory`	Entfernen eines Verzeichnisses, auch wenn es nicht leer ist (nur für den Benutzer *root*)
`-f` oder `--force`	Schreibgeschützte Dateien werden gelöscht, ohne eine weitere Bestätigung anzufordern.
`-i` oder `--interactive`	Vor jedem Löschen wird eine Bestätigung angefordert.
`-r` oder `-R` oder `--recursive`	Rekursives Löschen: Die Inhalte von Verzeichnissen und die Verzeichnisse selbst werden rekursiv entfernt.
`-v` oder `--verbose`	Ausführliche Ausgabe.

Tabelle 3.17: Optionen des Befehls `rm`

Wenn Sie den Befehl interaktiv starten, werden Sie vor dem Löschen jeder Datei und jedes Verzeichnisses noch einmal gefragt, ob Sie sicher sind:

```
$ rm -i dat2000
rm: remove regular empty file dat2000? y
```

Der Befehl wird auch dazu verwendet, Verzeichnisse zu löschen, die nicht leer sind. Dazu verwenden Sie die Option `-r`:

```
$ rm -r bericht04
```

►► **Vorsicht:** *Der gefährlichste Linux-Befehl überhaupt ist der Befehl* `rm -r *` *oder gar* `rm -rf *`*, denn er löscht rekursiv alles ab der Stelle im Verzeichnisbaum, an der Sie sich gerade befinden. Die Option* `-f` *bedeutet, dass auch schreibgeschützte Dateien ohne zusätzliche Bestätigung gelöscht werden sollen. Bevor Sie einen solchen Befehl absetzen, sollten Sie zumindest zuvor mit dem Befehl* `pwd` *überprüfen, in welchem Verzeichnis Sie sich befinden.* ◄◄

3.1.7 Dateien und Verzeichnisse kopieren

Der Befehl cp

Mit diesem Befehl können Sie Dateien und Verzeichnisse kopieren.

```
$ cp [-option(en)] quelle(n) ziel
```

Es ist entweder möglich, eine Datei auf eine andere Datei zu kopieren, oder ein Verzeichnis auf ein anderes Verzeichnis, oder eine oder mehrere Dateien in ein Verzeichnis.

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-a` oder `--archive`	Eigentümer, Rechte und Verzeichnisstruktur der Quelldateien werden beibehalten, wenn möglich. Die Option entspricht `-dpR`.
`-b` oder `--backup`	Erzeugen einer Sicherung der eventuell existierenden Zieldateien.
`-d` oder `--no-dereference` oder `--preserve=link` oder `-P`	Symbolischen Links wird nicht gefolgt, das heißt, es wird nicht die Originaldatei, auf die der Link verweist, sondern nur der Link kopiert.
`-f` oder `--force`	Vorhandene Zieldateien werden ohne Warnung überschrieben.
`-i` oder `--interactive`	Vor jedem Überschreiben einer Datei wird eine Bestätigung angefordert.
`-H`	Symbolische Links werden aufgelöst, das heißt, es wird die Originaldatei und nicht der Link kopiert. Das ist das Standardverhalten, kann aber zur Dokumentation zum Beispiel in Stapelverarbeitungen ausdrücklich als Option angewiesen werden.
`-l` oder `--link`	Statt einer Kopie wird nur ein Hard Link für eine normale Datei erzeugt.
`-L` oder `--dereference`	Symbolischen Links wird immer gefolgt.
`-p` oder `--preserve=mode,ownership,timestamps`	Eigenschaften (Eigentümer, Gruppe, Zugriffsrechte und Datumsangaben) der Quelldatei beibehalten.
`--no-preserve=`*`attribut`*	Angegebene Attribute (`mode`, `ownership`, `timestamps`) nicht beibehalten.
`--parents`	Die Verzeichnisstruktur der Quelldatei bleibt erhalten, indem der vollständige Quellpfad der Dateien an das Zielverzeichnis gehängt bzw. dort angelegt wird.
`-r` oder `-R` oder `--recursive`	Rekursives Kopieren: Die Inhalte von Verzeichnissen werden rekursiv kopiert, also mit allen darunter liegenden Verzeichnissen.
`--remove-destination`	Jede bereits vorhandene Zieldatei wird zuerst gelöscht und erst dann überschrieben.
`--reply=`*`antwort`*	Nachfragen, wenn die Zieldatei schon existiert: `yes` (ja, immer), `no` (nein, nie) oder `query` (fragen).
`-s` oder `--symbolic-link`	Statt Kopien werden symbolische Links erzeugt.
`-S` oder `--suffix=`*`suffix`*	Festlegen der Endung, die beim Anlegen einer Sicherheitskopie dem Dateinamen angefügt wird. Die Vorgabe ist die Tilde `~`.
`--target-directory=`*`dir`*	Alle Quelldateien und -verzeichnisse werden in das Verzeichnis *`dir`* verschoben.
`-u` oder `--update`	Die vorhandene Zieldatei wird nur dann überschrieben, wenn die Quelldatei dasselbe oder ein neueres Änderungsdatum besitzt.
`-v` oder `--verbose`	Ausführliche Ausgabe.
`-x` oder `--one-file-system`	Unterverzeichnisse in anderen Dateisystemen werden ignoriert.

Tabelle 3.18: Optionen des Befehls `cp`

Im nachfolgenden Beispiel wird die Datei `prot1` auf die Datei `prot1000` kopiert:

```
$ cp prot1 prot99
```

►► **Stop:** *Der Befehl überschreibt die Datei* `prot99`*, wenn sie bereits existiert. Bereits existierende Verzeichnisse werden durch den Befehl aber nicht überschrieben.* ◄◄

Normalerweise werden die Informationsdaten beim Kopieren entsprechend dem kopierenden Benutzer gesetzt. Das heißt, er wird der Besitzer, seine primäre Gruppe (vgl. Kapitel 9) erhält Gruppenbesitz und das Erstellungsdatum ist das aktuelle Datum. Mit folgendem Befehl können die originalen Dateiangaben erhalten bleiben:

```
$ cp -p dat1 datneu
```

Sie können den Befehl auch interaktiv starten, dann werden Sie vor dem Überschreiben noch einmal gefragt, ob Sie sicher sind:

```
$ cp -i prot1 prot1000
cp: overwrite prot1000? y
```

Der Befehl wird auch dazu verwendet, ganze Verzeichnisbäume zu kopieren, zum Beispiel den des Verzeichnisses `bericht`. Dazu verwenden Sie die Option `-r`:

```
$ cp -r bericht bericht2001
```

Ist das Verzeichnis `bericht2001` bereits vorhanden, wird das Verzeichnis `bericht` unterhalb des Verzeichnisses `bericht2001` angelegt. Wenn nicht, wird das Verzeichnis `bericht2001` mit dem Inhalt des Verzeichnisses `bericht` erzeugt.

Der folgende Befehl:

```
$ cp --parents a/b/c/text1 dirneu
```

kopiert die Datei `text1` mitsamt ihrem Verzeichnisbaum `a/b/c` in das Verzeichnis `dirneu`.

Es ist auch möglich, mehrere Dateien in ein Verzeichnis zu kopieren:

```
$ cp prot1 prot1000 bericht2001
```

In diesem Fall muss das Verzeichnis `bericht2001` bereits existieren.

►► **Info:** *Es ist allerdings nicht möglich, mehrere Dateien in eine Datei zu kopieren. Dazu müssen Sie den Befehl* `cat` *verwenden.* ◄◄

3.1.8 Dateien und Verzeichnisse verschieben und umbenennen

Der Befehl mv

Mit diesem Befehl können Sie Dateien und Verzeichnisse umbenennen und/oder verschieben.

```
$ mv [-option(en)] quelle(n) ziel
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-b` oder `--backup`	Erzeugen einer Sicherung der existierenden Zieldateien.
`-f` oder `--force` oder `--reply=yes`	Vorhandene Zieldateien werden ohne Warnung überschrieben.
`-i` oder `--interactive` oder `--reply=query`	Vor jedem Überschreiben einer Datei wird eine Bestätigung angefordert.

Tabelle 3.19: Optionen des Befehls `mv`

Option	Bedeutung
`--reply=`*antwort*	Nachfragen, wenn die Zieldatei schon existiert: `yes` (ja, immer), `no` (nein, nie) oder `query` (fragen).
`-S` oder `--suffix=`*suffix*	Festlegen der Endung, die beim Anlegen einer Sicherheitskopie dem Dateinamen angefügt wird. Die Vorgabe ist die Tilde ~.
`--target-directory=`*dir*	Alle Quellargumente werden in das Verzeichnis *dir* verschoben.
`-u` oder `--update`	Die vorhandene Zieldatei wird nur dann überschrieben, wenn die Quelldatei dasselbe oder ein neueres Änderungsdatum besitzt.
`-v` oder `--verbose`	Ausführliche Ausgabe.
`-x` oder `--one-file-system`	Unterverzeichnisse in anderen Dateisystemen werden ignoriert.

Tabelle 3.19: Optionen des Befehls `mv (Forts.)`

Im nachfolgenden Beispiel wird eine Datei umbenannt:

```
$ mv prot1 prot1111
```

►► **Stop:** *Der Befehl überschreibt die Datei* `prot1111`*, wenn sie bereits existiert.* ◄◄

Sie können den Befehl auch interaktiv starten, dann werden Sie vor dem Überschreiben noch einmal gefragt, ob Sie sicher sind:

```
$ mv -i prot1 prot1111
rm: overwrite prot1111? y
```

Sie können den Befehl auch verwenden, um eine Datei oder ein Verzeichnis zu verschieben:

```
$ mv prot1111 bericht
```

Hier wurde die Datei `prot1111` in das Verzeichnis `bericht` verschoben. Der Befehl wird auch dazu verwendet, ganze Verzeichnisbäume umzubenennen:

```
$ mv bericht protokolle
```

Es ist auch möglich, gleichzeitig zu verschieben und umzubenennen:

```
$ mv protokolle/prot1111 prot1
```

Die Datei `prot1111` im Verzeichnis `protokolle` wird ins aktuelle Verzeichnis verschoben und dort `prot1` genannt.

3.1.9 Dateien vergleichen

Der Befehl diff

Dieser Befehl vergleicht den Inhalt von zwei Dateien und gibt alle unterschiedlichen Zeilen aus:

```
$ diff [-option(en)] datei1 datei2
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-a` oder `--text`	Alle Dateien werden als Textdatei behandelt; auf diese Weise können auch Binärdateien mit dem Befehl verglichen werden.
`-b` oder `--ignore-space-change`	Wiederholte Leerzeichen werden wie ein Leerzeichen behandelt.

Tabelle 3.20: Optionen des Befehls `diff`

Option	Bedeutung
-B **oder** --ignore-blank-lines	Leerzeilen werden ignoriert.
-c **oder** --c *zahl* **oder** --context=[*zahl*]	Verändert das Format der Ausgabe der Zeilen: diese werden in einem Zusammenhang von *zahl* Zeilen ausgegeben (Vorgabewert ist 3).
-d **oder** --minimal	Versucht unbedingt so wenig Unterschiede wie möglich zu finden.
-e **oder** --ed	Ausgabe einer Befehlsfolge des Editors ed mit allen Änderungen.
-E **oder** --ignore-tab-expansion	Änderungen aufgrund Tabulator-Erweiterungen werden ignoriert.
-F *re* **oder** --show-function-line=*re*	Zeigt die letzte Zeile mit einer Übereinstimmung mit dem regulären Ausdruck *re*.
--from-file=*datei1*	Vergleicht *datei1* mit allen Operanden, wobei *datei1* ein Verzeichnis sein kann.
--to-file=*datei2*	Vergleicht *datei2* mit allen Operanden, wobei *datei2* ein Verzeichnis sein kann.
-i **oder** --ignore-case	Ignoriert Unterschiede in der Groß- und Kleinschreibung in den Dateien.
-I *re* **oder** --ignore-matching-lines=*re*	Unterschiede in Zeilen mit einer Übereinstimmung mit dem regulären Ausdruck *re* werden ignoriert.
--ignore-file-name-case	Ignoriert die Schreibweise beim Vergleich von Dateinamen.
-l **oder** --paginate	Die Ausgabe wird dem Befehl pr übergeben, der sie in Seiten unterteilt.
--left-column	Wenn die Option -y für das zweispaltige Format gewählt wurde, gibt diese Option bei gleichen Spalten nur die linke Spalte aus.
-n **oder** --rcs	Die Ausgabe erfolgt im RCS-Diff-Format.
-N **oder** --new-file	Behandelt nicht vorhandene Datei als leere Dateien.
--no-ignore-file-name-case	Berücksichtigt die Schreibweise beim Vergleich von Dateinamen.
-normal	Die Ausgabe erfolgt als normale Diff-Datei.
-p **oder** --show-c-function	Zeigt, in welcher C-Funktion sich jede Änderung befindet.
-q **oder** --brief	Kurzausgabe, die nur besagt, ob die Dateien sich unterscheiden.
-r **oder** --recursive	Vergleicht alle gefundenen Unterverzeichnisse rekursiv.
--strip-trailing-cr	Nachfolgende Zeilenschaltung bei der Eingabe ignorieren.
--suppress-common-lines	Wenn die Option -y für das zweispaltige Format gewählt wurde, unterdrückt diese Option die Ausgabe gleicher Zeilen.
-s **oder** --report-identical-files	Gibt eine Nachricht aus, wenn zwei Dateien identisch sind.
-S *datei* **oder** --starting-file=*datei*	Startet mit der angegebenen Datei *datei* beim Vergleich von Verzeichnissen.
--speed-large-files	Es werden große Dateien vorausgesetzt und viele verstreute kleine Änderungen.
-t **oder** --expand-tabs	Wandelt Tabulatoren in Leerzeichen in der Ausgabe um.
-T **oder** --initial-tab	Stellt einen Tabulator an den Zeilenanfang, um vorhandene Tabulatoren passend anzuordnen.

Tabelle 3.20: Optionen des Befehls diff *(Forts.)*

Option	Bedeutung
`-u` **oder** `-U` *zahl* **oder** `--unified=[`*zahl*`]`	Diese Option ist eine einheitlichere Form der Option `-c`: das Format der Ausgabe der Zeilen wird verändert und sie werden in einem einheitlichen Zusammenhang von *zahl* Zeilen ausgegeben (Vorgabewert ist 3).
`--unidirectional-new-file`	Behandelt eine nicht vorhandene erste Datei als leere Datei.
`-w` **oder** `--ignore-all-space`	Ignoriert alle Leerzeichen.
`-W` *zahl* **oder** `--width=`*zahl*	Diese Option gibt zusammen mit der Option -y für die zweispaltige Ausgabe an, wie breit die maximale Ausgabe in *zahl* ist. Vorgabe ist 130.
`-x` *re* **oder** `--exclude=`*re*	Schließt Dateien eines Verzeichnisses aus dem Vergleich aus, wenn es mit dem Ausdruck ***re*** übereinstimmt.
`-X` *datei* **oder** `--exclude-from=`*datei*	Schließt Dateien eines Verzeichnisses aus dem Vergleich aus, wenn diese mit den Ausdrücken aus der Datei *datei* übereinstimmen.
`-y` **oder** `--side-by-side`	Ausgabe in zwei Spalten.

Tabelle 3.20: Optionen des Befehls `diff` *(Forts.)*

Zum Beispiel:

```
$ diff obst obst1
3a4
> pfirsich
7d7
< mandarinen
```

Es werden zuerst die Nummer der abweichenden Zeilen und dann deren Inhalt angezeigt. Die Zeilen aus der ersten Datei werden mit > und die aus der zweiten Datei mit < markiert. In diesem Beispiel weicht die vierte Zeile der zweiten Datei mit dem Eintrag `pfirsich` von der ersten Datei ab und die erste Datei hat in der siebten Zeile den abweichenden Eintrag `mandarinen`.

Der Befehl diff3

Dieser Befehl vergleicht den Inhalt von drei Dateien. Dazu wird die Grundversion einer Datei verwendet und mit den Änderungen der ersten Datei abgeglichen. Anschließend werden die Unterschiede zwischen diesen beiden Dateien mit der dritten Datei abgeglichen und alle unterschiedlichen Zeilen ausgegeben:

```
$ diff3 [-option(en)] datei1 datei2 datei3
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-3` **oder** `--easy-only`	Ausgabe von nicht zusammengeführten und nicht überlappenden Änderungen.
`-a` **oder** `--text`	Alle Dateien werden als Textdatei behandelt; auf diese Weise können auch Binärdateien mit dem Befehl verglichen werden.
`-A` **oder** `--show-all`	Ausgabe von allen Änderungen und Konflikten, die letzteren wrden in Klammern angezeigt.

Tabelle 3.21: Optionen des Befehls `diff3`

Option	Bedeutung
`--diff-program=`*programm*	Verwenden eines Programms zum Vergleichen der Dateien.
`-e` oder `--ed`	Ausgabe einer Befehlsfolge des Editors `ed` mit allen Änderungen.
`-E` oder `--show-overlap`	Ausgabe einer Befehlsfolge des Editors `ed` mit allen Änderungen, die nicht zusammengeführt wurden, und Anzeige von überlappenden Änderungen in Klammern.
`-i` oder `--ignore-case`	Hängt an die Ausgabe der Befehlsfolge des Editors `ed` zusätzlich die Befehle `w` (`write`) und `q` (`quit`) an, damit `ed` beim Verlassen gespeichert wird.
`-L` *datei* oder `--label=` *datei*	Für die Ausgabe wird die Datei *datei* verwendet.
`-m` oder `--merge`	Das Ergebnis aller Änderungen wird in eine Datei ausgegeben.
`-T` oder `--initial-tab`	Stellt einen Tabulator an den Zeilenanfang, um vorhandene Tabulatoren passend anzuordnen.
`-x` oder `--overlap-only`	Ausgabe einer Befehlsfolge des Editors `ed` mit allen überlappenden Änderungen.
`-X` *datei* oder `--exclude-from=`*datei*	Entspricht der Option `-x`, aber Ausgabe von überlappenden Änderungen in Klammern.

Tabelle 3.21: Optionen des Befehls `diff3` *(Forts.)*

Zum Beispiel:

```
$ diff eins zwei drei
====
1:4,5c
  vier
  fünf
2:4,5c
  vier
  5
3:4,5c
  4
  fünf
```

Es werden zuerst die Nummer der Dateien, dann die abweichenden Zeilen und schließlich deren Inhalt angezeigt. In diesem Beispiel weicht die fünfte Zeile der ersten Datei mit dem Eintrag `fünf` von der zweiten Datei ab und die vierte Zeile der ersten Datei von der dritten Datei mit dem Eintrag `vier`.

►► **Info:** *Mit* `diff3` *könnten verglichen werden: ein Ursprungprogramm in einer Version X, eine kundenangepasste Kopie des Ursprungsprogramms und das neue Ursprungsprogramm in der Version X+1. Ergebnis wäre eine Liste aller Änderungen, die notwendig sind, um die kundenangepasste Kopie auf den Versionsstand X+1 anzuheben (soweit technisch ermittelbar).* ◄◄

Der Befehl cmp

Dieser Befehl vergleicht zwei Dateien und bricht an der ersten Stelle ab, an der diese sich unterscheiden.

```
$ cmp [-option(en)] datei1 datei2
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-b` oder `--print-bytes`	Ausgabe der unterschiedlichen Bytes.
`-c` oder `--print-chars`	Die sich unterscheidenden Bytes werden als Buchstaben ausgegeben.
`-i` *`zahl`* oder `--ignore-initial=`*`zahl`*	Die ersten *`zahl`* Bytes der Eingabe werden übersprungen.
`-i` *`zahl:zahl`* oder `--ignore-initial=`*`zahl:zahl`*	Die ersten *`zahl`* Bytes der Eingabe von Datei1 (*`zahl`* vor dem Doppelpunkt) und von Datei2 (*`zahl`* nach dem Doppelpunkt) werden übersprungen.
`-l` oder `--verbose`	Ausgabe der Bytenummern (Position) und Werte (Inhalt) aller sich unterscheidenden Bytes.
`-n` *`zahl`* oder `--bytes=`*`zahl`*	Es werden höchstens *`zahl`* Bytes verglichen.
`-s` oder `--quiet` oder `--silent`	Es erfolgt keine Ausgabe, sondern es wird nur ein Rückgabewert an das aufrufende Programm übergeben: 0 bedeutet, die Dateien sind gleich. 1 bedeutet, die Dateien unterscheiden sich. 2 bedeutet, ein Zugriff auf die Dateien ist nicht möglich.

Tabelle 3.22: Optionen des Befehls `cmp`

```
$ cmp obst obst1
obst obst1 differ: char 21, line 4
```

Die Dateien unterscheiden sich ab dem 21. Byte in der vierten Zeile.

```
$ cmp -c eins zwei
eins zwei differ: byte 21, line 5 is 146 f  65 5
```

Die Dateien unterscheiden sich ab dem 21. Byte in der 5. Zeile. Bei der Datei `eins` lautet das Byte `f` und bei Datei `zwei` lautet es `5`.

Der Befehl comm

Dieser Befehl vergleicht zwei sortierte Dateien und zeigt gemeinsame Zeilen an sowie Zeilen, an denen diese sich unterscheiden.

```
$ comm [-option] datei1 datei2
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-1`	Zeilen unterdrücken, die nur in *`datei1`* enthalten sind.
`-2`	Zeilen unterdrücken, die nur in *`datei2`* enthalten sind.
`-3`	Zeilen unterdrücken, die in beiden Dateien enthalten sind.

Tabelle 3.23: Optionen des Befehls `comm`

Zum Beispiel:

```
$ comm eins drei
                eins
                zwei
                drei
4
fünf
        vier
        5
```

Die Dateien unterscheiden sich ab der dritten Zeile. Die Übereinstimmungen werden rechts angedruckt und die Unterscheidungen linksbündig bzw. mittig.

3.1.10 Dateiausgabe formatieren

Der Befehl groff

Dieser Befehl ist ein so genanntes Frontend für das Dokumentenformatiersystem `groff`. Dieses ruft in der Regel troff mit einem Postprozessor für das angegebene Gerät auf. Die Syntax des Befehls lautet:

```
$ groff [-option(en)] datei(en)
```

►► **Info:** *Die Befehle* `troff` *und* `nroff` *werden in ähnlicher Weise verwendet.* ◄◄

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-a`	Erzeugt eine ASCII-Annäherung bei der Ausgabe.
`-b`	Druckt mit jeder Warn- oder Fehlermeldung eine Rückverfolgung aus, um die Fehlerursache zu finden.
`-C`	Aktiviert den Kompatibilitätsmodus.
`-d`*cs* `-d`*name*`=`*s*	Weist `c` oder einem Namen die Zeichenkette *s* zu. Dabei muss `c` ein aus einem Buchstaben bestehender Name sein.
`-e`	Vorbearbeitung mit dem Befehl `eqn`.
`-E`	Unterdrückt alle Fehlermeldungen von `troff`.
`-f`*fam*	*fam* wird als standardmäßige Font-Familie verwendet.
`-F`*dir*	Sucht im Verzeichnis oder den Verzeichnispfad *dir* nach Unterverzeichnissen *devname* (der Name ist der Name des Geräts) und dort nach einer Datei DESC und Font-Dateien.
`-g`	Vorbearbeitung mit dem Befehl `grn`.
`-G`	Vorbearbeitung mit dem Befehl `grap`.
`-h`	Hilfetext ausgeben.
`-i`	Liest die Standardeingabe, nachdem alle angegebenen Eingabedateien bearbeitet wurden.
`-I`*dir*	Diese Option kann verwendet werden, um ein Verzeichnis anzugeben, das nach Dateien durchsucht wird. Die Option setzt die Option `-s` voraus.
`-l`	Die Ausgabe wird an ein Spoolprogramm zum Druck versandt.
`-L`*arg*	Übergibt *arg* an das Spoolprogramm. Jedes Argument sollte mit einer eigenen Option `-L` übergeben werden.

Tabelle 3.24: Optionen des Befehls `groff`

Option	Bedeutung
-m*name*	Liest die Datei `name.tmac` ein und wenn diese nicht gefunden wird, stattdessen `tmac.name`.
-M*dir*	Sucht im Verzeichnis oder den Verzeichnispfad *dir* für Makro-Dateien.
-N	Lässt keine Zeilenschaltungen bei Begrenzungszeichen von `eqn` zu.
-o*list*	Gibt nur Seiten in einer Liste aus, die eine durch Komma getrennte Liste von Seitenbereichen ist.
-p	Vorbearbeitung mit dem Befehl pic.
-P*arg*	Übergibt arg dem Postprozessor. Jedes Argument sollte mit einer eigenen Option -L übergeben werden.
-R	Vorbearbeitung mit dem Befehl refer. Es wird dabei kein Mechanismus für die Übergabe von Argumenten zur Verfügung gestellt.
-r*cn* -r*name=n*	Definiert das Zahlenregister *c* oder den Namen mit *n*, wobei *c* ein aus einem Zeichen bestehender Name sein muss. *n* kann jeder beliebiger numerischer Ausdruck von troff sein.
-s	Vorbearbeitung mit dem Befehl soelim.
-S	Sicherer Modus. Übergibt die Option -S an das Programm pic und deaktiviert die folgenden troff-Anforderungen: `.open`, `.opena`, `.pso`, `.sy` und `.pi`. Dieser Modus ist aus Sicherheitsgründen Standard.
-t	Vorbearbeitung mit dem Befehl `tbl`.
-T*dev*	Bereitet die Ausgabe für das Gerät `dev` vor. Standardgerät ist `ps`.
-U	Unsicherer Modus. Kehrt zum ehemaligen unsicheren Verhalten des Befehls zurück.
-v	Veranlasst, das von `groff` aufgerufene Programme ihre Versionsnummer ausgeben.
-V	Druckt die Pipeline an der Standardausgabe aus anstatt sie auszuführen.
-wname	Aktiviert einen Namen für eine Warnung
-Wname	Unterdrückt Namen für Warnungen.
-X	Vorschau mit dem Programm `gxditview` anstatt den normalen Postprozessor zu verwenden.
-z	Unterdrückt die (formatierte) Ausgabe von `troff`. Nur Fehlermeldungen werden ausgegeben.
-Z	Die Ausgabe von `troff` wird nicht nachbearbeitet. Normalerweise startet `troff` automatisch den geeigneten Postprozessor.

Tabelle 3.24: Optionen des Befehls `groff` *(Forts.)*

Verfügbare Geräte sind:

Gerät	Bedeutung
ps	Für Postscript-Drucker und -Vorschau.
dvi	Für TeX dvi-Format.
X75	Für eine 75dpi X11-Vorschau.
X100	Für eine 100dpi X11-Vorschau.
ascii	Für Schreibmaschinen-ähnliche Geräte.
ascii8	Für Schreibmaschinen-ähnliche Geräte. Im Gegensatz zu `ascii` ist dieses Gerät 8 Bit-fähig. Dieses Gerät ist dafür vorgesehen, für andere Zeichensätze als ASCII oder ISO 8859-1 verwendet zu werden.
latin-1	Für Schreibmaschinen-ähnliche Geräte, die den Zeichensatz ISO 8859-1 verwenden.

Tabelle 3.25: Geräteoptionen des Befehls `groff`

Gerät	Bedeutung
utf8	Für Schreibmaschinen-ähnliche Geräte, die den Zeichensatz ISO 10646 (Unicode) verwenden.
cp1047	Für Schreibmaschinen-ähnliche Geräte, die den Zeichensatz EBCDIC IBM cp1047 verwenden.
nippon	Für Schreibmaschinen-ähnliche Geräte, die den japanischen Zeichensatz EUC verwenden.
lj4	Für HP LaserJet4-kompatible (PCL5-kompatible) Drucker.
lbp	Für Canon CAPSL-Drucker (LBP-4 und LBP-8-Serie Laserdrucker).
html	Erzeugung von HTML-Ausgabe.

Tabelle 3.25: Geräteoptionen des Befehls `groff` *(Forts.)*

Um zum Beispiel die Manpage `ls.1` auf der Standardausgabe mit dem Ausgabegerät `ascii` und dem Befehl `less` seitenweise auszugeben, kann der folgenden Befehl verwendet werden:

```
$ groff -m mandoc -Tascii ls.1 | less
```

Der Befehl col

Dieser Befehl ist ein Filter, um Escape-Sequenzen und Zeilenschaltungen (LF) nachzubearbeiten, um die Ausgabe eines Befehls, zum Beispiel `troff`, leserlich darzustellen. Die Syntax des Befehls lautet:

```
$ col [-option(en)]
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
-b	Keine Ausgabe des Zeichens Backspace. Eine nützliche Option beim Drucken von Manpages.
-f	Halbzeilenvorschübe sind zugelassen.
-l*num*	Mindestens *num* Zeilen werden gepuffert; standardmäßig werden 128 Zeilen gepuffert.
-p	Erzwingt, dass unbekannte Steuersequenzen unverändert durchgegeben werden. Standardmäßig filtert `col` alle ihm bekannten Steuersequenzen von der Eingabe.
-x	Standardmäßig wandelt `col` Leerzeichen in Tabulatoren um; diese Option unterdrückt die Umwandlung.

Tabelle 3.26: Optionen des Befehls `col`

In diesem Beispiel wird die Ausgabe einer Manpage in der Datei `ls.rtf` gespeichert, wobei die Backspaces entfernt werden:

```
$ man ls | col -b > ls.rtf
```

Der Befehl colcrt

Dieser Befehl ist ein weiterer Filter, um Escape-Sequenzen und Zeilenschaltungen (LF) nachzubearbeiten, um die Ausgabe eines Befehls leserlich darzustellen. Die Syntax des Befehls lautet:

```
$ colcrt [-option(en)] datei(en)
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
-	Unterdrückt das Unterstreichen von Text.
-2	Bei halben Zeilen wird die Ausgabe mit doppeltem Zeilenabstand formatiert.

Tabelle 3.27: Optionen des Befehls `colcrt`

Zum Beispiel:
```
$ colcrt -2 liste.rtf
```


Der Befehl colrm

Dieser Befehl löscht die angegebenen Spalten einer Datei. Eine Spalte entspricht einem einzelnen Zeichen einer Zeile. Die Syntax des Befehls lautet:
```
$ colrm [startspalte [endespalte]]
```
In diesem Beispiel wird die 5. bis 17. Spalte aus der Datei `liste.txt` entfernt:
```
$ colrm 5 17 < liste.txt
```
Das Programm liest normalerweise von der Standardeingabe.

Der Befehl column

Dieser Befehl formatiert die Zeilen der Eingabe in Spalten. Die Elemente füllen dabei zuerst die Ausgabezeilen auf. Die Syntax des Befehls lautet:
```
$ column [-option(en)] datei
```
Der Befehl kennt folgende Optionen:

Option	Bedeutung
-c *columns*	Die Gesamtbreite der Ausgabe wird auf die Breite der angegebenen Spalten formatiert.
-s *char*	Definiert ein Zeichen, das verwendet wird, um die Spalten für die Option `-t` zu trennen.
-t	Legt die Anzahl der Spalten fest, die die Eingabe enthält und erzeugt eine Tabelle. Die Spalten werden standardmäßig durch Leerzeichen getrennt bzw. durch das mit der Option `-s` angegebene Zeichen.
-x	Füllt die Spalten vor den Reihen.

Tabelle 3.28: Optionen des Befehls `column`

Zum Beispiel wird die Datei `liste` so ausgegeben, dass der Buchstabe ; als Trennzeichen verwendet wird:
```
$ column -t -s ";" < liste
```

Der Befehl cut

Mithilfe dieses Befehls können Sie die Ausgabe einer Datei oder eines Datenstroms verändern, indem bestimmte Spalten oder Felder daraus extrahiert werden. Die Syntax des Befehls lautet:
```
$ cut [-option(en)] datei
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-b` oder `--bytes=`*liste*	Nur die angegebenen Bytes werden ausgegeben.
`-c` oder `--characters=`*liste*	Nur die angegebenen Spalten werden ausgegeben.
`-d`*x* oder `--delimiter=`*x*	Das Trennzeichen der Felder ist *x*.
`-f` oder `--fields=`*liste*	Nur die angegebenen Felder werden ausgegeben und der Tabulator ist das Feldtrennzeichen.
`--output-delimiter=`*string*	Verwendet *string* als Ausgabetrennzeichen; Standard ist, das Eingabetrennzeichen zu verwenden.
`-s` oder `--only-delimited`	Zeilen, die keine Trennzeichen enthalten, werden nicht ausgegeben.

Tabelle 3.29: Optionen des Befehls `cut`

Es kann nur eine der Optionen `-b`, `-c` oder `-f` verwendet werden. Jede Listenangabe besteht aus einem Bereich oder mehreren durch Kommas getrennten Bereichen. Mögliche Bereiche sind:

Bereich	Bedeutung
`N`	Das n-te Byte, Zeichen oder Feld; gezählt wird ab 1.
`N-`	Vom n-ten Byte, Zeichen oder Feld an bis zum Ende der Zeile.
`N-M`	Vom n-ten bis einschließlich zum m-ten Byte, Zeichen oder Feld.
`-M`	Vom ersten bis einschließlich zum m-ten Byte, Zeichen oder Feld.

Tabelle 3.30: Bereiche des Befehls `cut`

Im folgenden Beispiel wird aus der Zeile der Zeitausgabe der Wochentag (Feld 1) und die Uhrzeit (Feld 4) und extrahiert. Als Trennzeichen wird das Leerzeichen verwendet:

```
$ date | cut -d' ' -f1,4
Fri 14:20:38
```

Im nächsten Beispiel werden die Zugriffsrechte und der dazu gehörende Dateiname im Verzeichnis `/home/her/briefe` ausgegeben:

```
$ ls -l /home/her/briefe
total 8
drwxr-xr-x  2   her  users  512  Nov 29 12:55 mai03
drwxr-xr-x  2   her  users  512  Nov 28 10:45 jun03
-rw-r--r--  1   her  users  267  Oct  4 09:33 vorl
$ ls -l /home/her/briefe | cut -c2-11,55-
otal 8
rwxr-xr-x  briefe
rwxr-xr-x  memos
rw-r--r--  vorlage1
```

Der Befehl expand

Dieser Befehl ersetzt die Tabulatoren der angegebenen Dateien mit Leerzeichen und gibt das Ergebnis an der Standardausgabe aus. Die Syntax des Befehls lautet:

```
$ expand [-option(en)] datei(en)
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-i` oder `--initial`	Konvertiert keine Tabulatoren, die nicht nach Leerzeichen kommen.
`-t` oder `--tabs=`*`nr`*	Standardmäßig entspricht ein Tabulator 8 Leerzeichen. Mit dieser Option kann die Anzahl der Leerzeichen, die der Tabulator einnimmt, angegeben werden.
`-t` oder `--tabs=`*`list`*	Verwendet eine durch Komma getrennte Liste für die genauen Tabulatorpositionen.

Tabelle 3.31: Optionen des Befehls `expand`

In diesem Beispiel werden die Tabulatoren der Datei `liste.txt` in Leerzeichen umgewandelt:

```
$ expand liste.txt
```

Der Befehl fmt

Dieser Befehl formatiert die Zeilen einer beliebigen Textdatei im Blocksatz in einer angegebenen Breite. Die Syntax des Befehls lautet:

```
$ fmt [-option(en)] datei
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-c` oder `--crown-margin`	Erhält die Einrückung der ersten beiden Zeilen eines Absatzes.
`-p` oder `--prefix=`*`string`*	Formatiert nur Zeilen, die mit dem Präfix *`string`* beginnen.
`-s` oder `--split-only`	Teilt lange Zeilen, aber entfernt keine Zeilenumbrüche.
`-t` oder `--tagged-paragraph`	Entspricht -c, wenn die Einrückungen der ersten beiden Zeilen nicht gleich sind. Andernfalls wird die erste Zeile unterschiedlich von der zweiten eingerückt.
`-u` oder `--uniform-spacing`	Fügt ein Leerzeichen zwischen Worten und zwei Leerzeichen zwischen Sätzen ein.
`-w` oder `--width=`*`nr`*	Angabe der maximalen Zeilenbreite. Standard sind 75 Spalten.

Tabelle 3.32: Optionen des Befehls `fmt`

Im Beispiel wird die Datei `liste` so ausgegeben, dass die Spaltenbreite 65 Zeichen beträgt:

```
$ fmt -w 65 liste
```

Der Befehl fold

Dieser Befehl fügt Zeilenumbrüche einer Datei hinzu, sodass die angegebene Breite eingehalten wird, auch mitten in einem Wort. Die Syntax des Befehls lautet:

```
$ fold [-option(en)] datei
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-b` oder `--bytes`	Zählt Bytes anstelle von Spalten.
`-s` oder `--spaces`	Führt den Umbruch nur bei Leerzeichen durch.
`-w` oder `--width=`*nr*	Verwendet die angegebene Spaltenbreite anstelle des Standards von 80 Spalten.

Tabelle 3.33: Optionen des Befehls `fold`

Im Beispiel wird die Datei `liste` so ausgegeben, dass die Spaltenbreite 55 Zeichen beträgt:

```
$ fold -w 55 liste
```

Der Befehl pr

Dieser Befehl kann verwendet werden, um Dateien zum Drucken aufzubereiten. Er kann unter anderem eine Ausgabe mit Seitennummern, Datum- und Zeitangabe einschließlich einer Anzeige des Dateinamens erzeugen. Die Syntax des Befehls lautet:

```
$ pr [-option(en)] datei(en)
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-`*n* oder `--columns=`*n*	Die Ausgabe erfolgt n-spaltig nach unten, wenn nicht die Option -a verwendet wird.
`+`*n*`[:`*m*`]` oder `--pages=`*n*`[:`*m*`]`	Der Ausdruck beginnt auf Seite n und endet gegebenenfalls bei *m*.
`-a` oder `--across`	Die Spalten werden horizontal anstelle von vertikal aufgefüllt.
`-c` oder `--show-control-chars`	Steuerzeichen werden in die so genannte Caret-Notation (zum Beispiel ^D) und andere nicht druckbare Zeichen in die Oktalzahlnotation mit Backslash umgewandelt.
`-d` oder `--double-space`	Die Ausgabe erfolgt mit doppeltem Zeilenabstand.
`-D` oder `--date-format=`*format*	Das Format *format* wird für das Datum der Kopfzeile verwendet.
`-e[`*char*`[:`*breite*`]]` oder `--expand-tabs[=`*char*`[:`*breite*`]]`	Tabulatoren oder angegebene Tabulatorzeichen *char* werden in Leerzeichen mit der angegebenen Breite *breite* umgewandelt. Standardbreite ist 8.
`-f` oder `-F` oder `--form-feed`	Seiten werden durch Seitenvorschub anstelle von Zeilenschaltungen getrennt (bei der Option `-F` durch eine 3-zeilige Kopfzeile und ohne diese Option durch eine 5-zeilige Kopf- und Fußzeile).
`-h "`*header*`"` oder `--header=`*header*	Verwendet eine zentrierte Kopfzeile *header* anstelle des Dateinamens.
`-i[`*char*`[:`*breite*`]]` oder `--output-tabs[=`*char*`[:`*breite*`]]`	Leerzeichen werden in Tabulatoren oder alternative Tabulatorzeichen *char* mit der angegebenen Breite *breite* umgewandelt. Standardbreite ist 8.
`-J` oder `--join-lines`	Verbindet Zeilen und deaktiviert das Abschneiden von Zeilen mit der Option `-W`. Es gibt keine Anordnung in Spalten.
`-l`*n* oder `--length=`*n*	Die Seitenlänge beträgt n Zeilen (Standard sind 66).

Tabelle 3.34: Optionen des Befehls `pr`

Option	Bedeutung
`-m` oder `--merge`	Druckt alle Dateien parallel, jede in einer Spalte.
`-n[`*trenn*`[:`*n*`]]` oder `--number-lines [= `*trenn*`[:`*n*`]]`	Fügt Zeilennummern hinzu und verwendet dabei entweder 5-stellige Ziffern oder die mit *n* angegebenen Stellen. Wird ein Trennzeichen *trenn* angegeben, dann wird dieses anstelle eines Tabulators an die Zeilennummer angehängt.
`-N`*n* oder `--first-line-number=`*n*	Startet das Zählen mit der angegebenen Nummer *n* bei der ersten Zeile der ersten gedruckten Seite.
`-o`*n* oder `--indent=`*n*	Den linken Rand um ***n*** Leerzeichen einrücken.
`-r` oder `--no-file-warnings`	Unterdrückt eine Warnmeldung, wenn eine Datei nicht geöffnet werden kann.
`-s[`*char*`]` oder `--separator=[`*char*`]`	Trennt Spalten mithilfe eines einzigen Zeichens, Standard ist das Tabulatorzeichen.
`-s`*string* oder `--sep-string[=`*string*`]`	Trennt Zeilen mithilfe einer Zeichenkette *string*, Standardtrennzeichen ist der Tabulator.
`-t` oder `--omit-header`	Unterdrückt Kopf- und Fußzeilen.
`-T` oder `--omit-pagination`	Unterdrückt Kopf- und Fußzeilen und Seitenvorschub.
`-V` oder `--show-nonprinting`	Verwendet die Oktalzeichennotation mit Backslash zur Darstellung von nicht druckbaren Zeichen.
`-w`*n* oder `--width=`*n*	Die Zeilenbreite beträgt *n* Zeichen (Standard sind 72).
`-W`*n* oder `--page-width=`*n*	Die Zeilenbreite beträgt *n* Zeichen (Standard sind 72). Es werden Zeilen abgeschnitten, wenn nicht die Option `-J` verwendet wird.

Tabelle 3.34: Optionen des Befehls `pr` *(Forts.)*

In diesem Beispiel wird der Inhalt der Datei `liste` ab Seite 3 aufgelistet:

```
$ pr +3 liste
```

Im nächsten Beispiel wird der Inhalt der Datei `liste` mit einer Seitenlänge von 30 Zeilen und dem Titel »Bücherliste« ausgegeben:

```
$ pr -h "Bücherliste" -l 30 liste
```

Im letzten Beispiel wird der Inhalt des aktuellen Verzeichnisses seitenweise, ohne Kopfzeile und in 3 Spalten aufgelistet:

```
$ ls | pr -t -4 | more
```

Der Befehl printf

Dieser Befehl gibt Texte in formatierter Form aus. Die Syntax des Befehls lautet:

```
$ printf format [strings]
```

Die angegebenen Zeichenketten `strings` werden in den angegebenen Formaten ausgegeben, für die so genannte *Escape-Sequenzen* verwendet werden. Die zu formatierende Zeichenkette sollte in Anführungszeichen stehen.
Die Formate haben folgende Bedeutung:

Format	Bedeutung
`\"`	Doppelte Hochkomma (Anführungszeichen).
`\`*nnn*	Ausgabe des ASCII-Zeichens mit dem Oktalwert *nnn*.
`\\`	Backslash

Tabelle 3.35: Formate des Befehls `printf`

Format	Bedeutung
\a	Signalton
\b	Backspace
\c	Unterdrückt nachfolgende Zeilenschaltung
\f	Seitenvorschub
\n	Zeilenschaltung (Newline)
\r	Zeilenende (Carriage return)
\t	Horizontaler Tabulator
\v	Vertikaler Tabulator

Tabelle 3.35: Formate des Befehls `printf` *(Forts.)*

Neben normalem Text können im Formatierungsstring auch Platzhalter angegeben werden, an deren Stelle nachfolgende Argumente (Zeichenketten, Zahlen) eingefügt werden. Die Platzhalter beginnen dabei mit einem Prozentzeichen % und können folgende Formatierungsoptionen enthalten.

Platzhalter	Bedeutung
text	Ausgabe von *text*.
%s	Platzhalter für eine Zeichenkette oder eine Zahl.
%*m*s	Platzhalter für eine rechtsbündig ausgegebene Zeichenkette mit einer Breite von *m* Zeichen.
%-*m*s	Platzhalter für eine linksbündig ausgegebene Zeichenkette mit einer Breite von *m* Zeichen.
%*m*.*n*s	Platzhalter für eine Zeichenkette mit einer Breite von *m* Zeichen. Von der Zeichenkette selbst werden aber nur *n* Zeichen ausgegeben.
%*m*.*n*f	Platzhalter für eine Gleitkommazahl mit einer Breite von *m* Zeichen und *n* Nachkommastellen.

Tabelle 3.36: Formate des Befehls `printf`

In diesem Beispiel wird Text mithilfe von Escape-Sequenzen mehrzeilig und mit Tabulator ausgegeben:

```
$ printf "Das ist eine Ausgabe\nauf 3\n\vZeilen\n"
Das ist eine Ausgabe
auf 3
       Zeilen
```

Im nächsten Beispiel wird eine rechtsbündige Ausgabe mit einer Breite von 25 Zeichen erzeugt:

```
$ printf "|%25s|\n" Bücherliste
|               Bücherliste|
```

Der Befehl sort

Dieser Befehl sortiert alle Eingabezeilen gemäß dem ASCII-Code aufsteigend. Standardmäßig wird die ganze Zeile in den Sortiervorgang mit eingeschlossen, es kann aber auch nach bestimmten Spalten innerhalb der Datei sortiert werden. Die Syntax des Befehls lautet:

```
$ sort [-option(en)] datei(en)
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-b` oder `--ignore-leading-blanks`	Führende Leer- und Tabulatorzeichen werden ignoriert
`-c` oder `--check`	Prüft, ob die Dateien bereits sortiert sind. In diesem Fall wird keine Ausgabe durchgeführt.
`-d` oder `--dictionary-order`	Die Sortierung berücksichtigt nur Zeichen, Ziffern und Leerzeichen (Sortierung in lexikalischer Ordnung).
`-f` oder `--ignore-case`	Groß-/Kleinschreibung wird ignoriert.
`-g` oder `--general-numeric-sort`	Vergleich erfolgt auf der Basis des allgemeinen numerischen Werts.
`-i` oder `--ignore-nonprinting`	Es werden nur druckbare Zeichen berücksichtigt.
`-k` oder `--key=`*`pos1`*`[,`*`pos2`*`]`	Schlüssel beginnt bei *pos1* und endet bei *pos2*. Die Nummerierung beginnt bei 1.
`-m` oder `--merge`	Fügt mehrere sortierte Dateien zusammen.
`-M` oder `--month-sort`	Die Sortierung erfolgt nach Monatskürzel in der Reihenfolge JAN < FEB < … < unbekannter Monatsname.
`-n` oder `--numeric-sort`	Numerisches Sortieren, das heißt Zahlen werden nach ihrem Wert sortiert, nicht nach der Reihenfolge der Ziffern (vgl. 34 und 120).
`-o` oder `--output=`*`datei`*	Das Ergebnis der Sortierung wird in die Datei *datei* geschrieben, die mit der Eingabedatei identisch sein kann.
`-r` oder `--reverse`	Absteigendes Sortieren
`-s` oder `--stable`	Stabilisiert die Sortierung, indem es den letzten Vergleich deaktiviert.
`-S` oder `--buffer-size=`*`n`*	Verwendet die Größe *n* als Hauptspeicherpuffer.
`-t`*`trenn`* oder `--field-separator=`*`trenn`*	Trennzeichen zwischen Feldern ist *trenn*.
`-T` oder `--temporary-directory=`*`dir`*	Das Verzeichnis *dir* wird für temporäre Dateien verwendet anstelle von `$TMPDIR` oder `/tmp`.
`-u` oder `--unique`	Gleiche Eingabezeilen werden nur einmal (unique) ausgegeben.
`-z` oder `--zero-terminated`	Zeilen werden mit dem Bytewert 0 anstelle mit einer Zeilenschaltung beendet.
`+`*`n`*`[-`*`m`*`]`	Der Sortierschlüssel beginnt bei Feld *n*+1 und endet bei einschließlich *m*; wenn *m* nicht vorhanden ist, werden alle Zeichen bis zum Zeilenende zum Sortierschlüssel.

Tabelle 3.37: Optionen des Befehls `sort`

►► **Info:** *Wenn der Befehl* `sort` *eine Zeile liest, wird diese in einzelne Felder aufgeteilt. Die Sortierschlüssel legen fest, nach welchem Feld sortiert wird. Der Standardwert ist das erste Feld und die Feldtrennzeichen sind standardmäßig Leerzeichen und Tabulatoren.* ◄◄

Im ersten Beispiel werden alle Zeilen der Datei `/etc/passwd` nach dem Zeilenanfang sortiert und in die Datei `passwd_sort` geschrieben. Dabei wird die Groß- und Kleinschreibung ignoriert:

```
$ sort -f /etc/passwd > passwd_neu; more passwd_neu
adm:x:4:4:Admin:/var/adm:
anna:x:107:1::/export/home/anna:/bin/sh
berta:x:1000:1::/export/home/berta:/bin/sh
bin:x:2:2::/usr/bin:
```

```
daemon:x:1:1::/:
her:x:101:1:Ute Hertzog:/export/home/her:/bin/bash
lp:x:71:8:Line Printer Admin:/usr/spool/lp:
...
```
Im zweiten Beispiel wird der Inhalt der Datei `/etc/passwd` numerisch mithilfe des dritten Feldes, der Benutzer-ID, sortiert. Das Feldtrennzeichen ist der Doppelpunkt:
```
$ sort +2n -t: /etc/passwd
```

Der Befehl tr

Dieser Befehl kann Zeichen umwandeln oder löschen. Dies ist beim Austauschen von Dateien zwischen unterschiedlichen Betriebs- oder Programmsystemen oft nützlich, wenn die verwendeten Zeichensätze voneinander abweichen. Die Syntax des Befehls lautet:
`$ tr [-`*option(en)*`] [`*string1* `[`*string2*`]]`
Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-c` oder `--complement`	Verwendung des ersten Komplements von Zeichenkette *string1*.
`-d` oder `--delete`	Alle Zeichen aus der Zeichenkette *string1* werden gelöscht und nicht übersetzt.
`-s` oder `--squeeze-repeats`	Jede Eingabefolge von Zeichen, die in der Zeichenkette *string1* enthalten sind und wiederholt werden, werden durch ein einzelnes Vorkommen dieses Zeichens ersetzt.
`-t` oder `--truncate` *string1*	Als Erstes wird die Zeichenkette *string1* auf die Länge von *string2* abgeschnitten.

Tabelle 3.38: Optionen des Befehls `tr`

Der Befehl `tr` kennt ebenfalls interpretierte Escape-Sequenzen:

Sequenz	Bedeutung
`\`*nnn*	Ausgabe des ASCII-Zeichens mit dem Oktalwert *nnn*.
`\\`	Backslash
`\a`	Signalton
`\b`	Backspace
`\c`	Unterdrückt nachfolgende Zeilenschaltung.
`\f`	Seitenvorschub
`\n`	Zeilenschaltung (Newline)
`\r`	Zeilenende (Carriage return)
`\t`	Horizontaler Tabulator.
`\v`	Vertikaler Tabulator.
char1-char2	Alle Zeichen von *char1* bis *char2* aufsteigend.
`[`*char**`]`	In Zeichenkette *string2* Zeichen *char* bis zur Länge von Zeichenkette *string1* kopieren.

Tabelle 3.39: Sequenzen des Befehls `tr`

Sequenz	Bedeutung
[*char***anz*]	Die Zeichen *char* werden *anz* Mal kopiert; bei der Zahl *anz* handelt es sich um eine Oktalzahl, wenn diese mit 0 beginnt.
[:alnum]	Alle Buchstaben und Zahlen.
[:alpha]	Alle Buchstaben.
[:blank]	Alle Leerzeichen und horizontalen Tabulatoren.
[:cntrl]	Alle Steuerzeichen.
[:digit]	Alle Ziffern.
[:lower]	Alle Kleinbuchstaben.
[:print]	Alle druckbaren Zeichen mit Leerzeichen.
[:punct]	Alle Satzzeichen.
[:space]	Alle Leerzeichen und Tabulatoren, horizontal und vertikal.
[:upper]	Alle Großbuchstaben.
[:xdigit]	Alle hexadezimalen Ziffern.
[=*char*=]	Alle Zeichen, die dem Zeichen *char* entsprechen.

Tabelle 3.39: Sequenzen des Befehls `tr` *(Forts.)*

Im ersten Beispiel werden alle Kleinbuchstaben der Datei `liste` in Großbuchstaben umgewandelt und das Ergebnis wird in der Datei `liste-gross` gespeichert:

```
$ tr "[:lower]" "[:upper]" < liste > liste-gross
```

Im nächsten Beispiel werden alle CR-Zeichen (mit Oktalcode 15) aus der Datei `dos.txt` gelöscht und das Ergebnis in die Datei `unix.txt` gespeichert. Damit wurde ein DOS-Text mit den Zeilenendezeichen CR und LF in einen Unix-Text mit dem Zeilenendezeichen LF umgewandelt:

```
$ tr -d '\015' < dos.txt > unix.txt
```

Der Befehl unexpand

Dieser Befehl kann Zeichenketten, die aus mindestens zwei führenden Leerzeichen oder Tabulatoren bestehen, in Tabulatoren umwandeln. Die Syntax des Befehls lautet:

```
$ unexpand [-option(en)] datei(en)
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-a` oder `--all`	Konvertiert nicht nur die führenden, sondern alle Leerzeichen.
`--first-only`	Konvertiert nur die führenden Leerzeichen; diese Option deaktiviert `-a`.
`-t` *n* oder `--tabs=`*n*	Tabulatoren werden *n* Zeichen anstelle der standardmäßigen 8 zugewiesen.
`-t` *list* oder `--tabs` *list*	Verwendet eine durch Kommas getrennte Liste als Tabulatorpositionen.

Tabelle 3.40: Optionen des Befehls `unexpand`

In diesem Beispiel werden alle führenden Leerzeichen in der Datei ***liste*** durch Tabulatoren ersetzt:

```
$ unexpand liste
```

Der Befehl uniq

Dieser Befehl entfernt doppelte und aufeinanderfolgende Zeilen aus der sortierten Datei `datei1` und schreibt das Ergebnis auf die Standardausgabe oder in die Datei `datei2`. Die Syntax des Befehls lautet:

```
$ uniq [-option(en)] [datei1 [datei2]]
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-c` oder `--count`	Stellt jeder Zeile die Anzahl voran, wie oft sie vorkommt.
`-d` oder `--repeated`	Druckt nur doppelt vorhandene Zeilen aus.
`-D` oder `--all-repeated[=`*methode*`]`	Druckt alle doppelten Zeilen aus. Die Trennmethode *methode* kann die Werte `none` (Standardwert), `prepend` und `separate` annehmen. Die Begrenzung erfolgt durch Leerzeilen.
`-`*n* oder `-f` *n* oder `--skip-fields=`*n*	Verhindert, dass die ersten *n* Felder verglichen werden.
`-i` oder `--ignore-case`	Ignoriert beim Vergleich eine unterschiedliche Schreibweise.
`+`*n* oder `-s` *n* oder `--skip-chars=`*n*	Verhindert, dass die ersten *n* Zeichen verglichen werden.
`-u` oder `--unique`	Gibt nur nicht doppelt vorhandene Zeilen aus.
`-w` *n* oder `--check-chars=`*n*	Vergleicht nicht mehr als *n* Zeichen in jeder Zeile.

Tabelle 3.41: Optionen des Befehls `uniq`

Im folgenden Beispiel werden alle Namen in der Datei `namensliste` ausgegeben, die mehr als einmal vorkommen, und in die Datei `d-namen` eingetragen:

```
$ sort namensliste | uniq -d > d-namen
```

Der Befehl iconv

Viele Linux-Versionen stellten den Zeichensatz in den vergangenen Jahren standardmäßig auf »de_DE.UTF-8«. Wenn durch diese UTF-8-Kodierung Zeichen in den Dateien nicht mehr korrekt dargestellt werden, kann dies mit dem Befehl iconv korrigiert werden. Der Befehl liest die Daten aus der Datei `datei1` und schreibt das konvertierte Ergebnis in die Standardausgabe (oder alternativ mit Ausgabeumlenkung oder Angabe der Ausgabedatei in die Datei `datei2`). Die Syntax des Befehls lautet:

```
$ iconv -f code -t code datei
```

oder:

```
$ iconv -f code -t code datei1 > datei2
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-c`	Lässt fehlerhafte Zeichen in der Ausgabe weg.
`-f` *code* oder `---from-code` *code*	Konvertiert aus dem angegebenen Zeichensatz.
`-l` oder `-list`	Listet alle bekannten Zeichensätze auf.

Tabelle 3.42: Optionen des Befehls `iconv`

Option	Bedeutung
`-o` *datei* oder `--output` *datei*	Gibt die Ausgabedatei an (es kann aber auch Ausgabeumlenkung verwendet werden).
`-s` oder `-silent`	Unterdrückt Warnungen, aber keine Fehlermeldungen.
`-t` *code* oder `--to-code` *code*	Konvertiert in den angegebenen Zeichensatz.

Tabelle 3.42: Optionen des Befehls `iconv` *(Forts.)*

Im folgenden Beispiel wird die mit dem Zeichensatz `latin1` kodierte Datei `bericht.txt` in die Datei `bericht1.txt` in den UTF-8-Zeichensatz konvertiert:

```
$ iconv -f latin1 -t utf-8 bericht.txt > bericht1.txt
```


3.1.11 Mit Dateien arbeiten

Der Befehl which

Dieser Befehl zeigt an, in welchem Pfad sich ein Shellbefehl befindet:

```
$ which [-option(en)] befehl
```

►► **Info:** *Es werden nur die Verzeichnisse durchsucht, die in der Variable* `PATH` *hinterlegt sind. Variablen werden im Kapitel 7 ausführlich behandelt.* ◄◄

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-a` oder `--all`	Ausgabe aller passenden Befehle, die mithilfe der Variablen PATH gefunden werden, und nicht nur des ersten.
`-i` oder `--read-alias`	Liest Aliase von der Standardeingabe und gibt passende auf der Standardausgabe aus.
`-skip-dot`	Überspringt Verzeichnisse in der Variablen `PATH`, die mit einem Punkt beginnen.
`--skip-tilde`	Überspringt Verzeichnisse in der Variablen `PATH`, die mit einer Tilde beginnen bzw. sich im Homeverzeichnis befinden.
`-show-dot`	Wenn ein Verzeichnis in der Variablen `PATH` mit einem Punkt beginnt und mit der Suche übereinstimmt, beginnt die Ausgabe des Befehls namens auch mit einem Punkt, zum Beispiel `./befehl`.
`--show-tilde`	Wenn ein Verzeichnis mit dem Homeverzeichnis übereinstimmt, wird eine Tilde am Anfang des Dateinamens ausgegeben.

Tabelle 3.43: Optionen des Befehls `which`

Zum Beispiel:

```
$ which man
/usr/bin/man
```

Der Befehl file

Dieser Befehl versucht den Typ einer Datei zu ermitteln, indem der oberste Teil einer Datei mithilfe der Datei `/etc/magic` ausgewertet wird.

```
$ file [-option(en)] datei(en)
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-b` **oder** `--brief`	Die Dateinamen werden nicht vor den Ausgabezeilen aufgeführt (Kurzmodus).
`-c` **oder** `--checking-printout`	Das Format der *`magic`*-Datei wird überprüft; die Option kann mit `-m` verwendet werden, um eine neue *`magic`*-Datei zu prüfen, bevor sie angewandt wird.
`-f` **oder** `--files-from` *`datei`*	Der Befehl wird auf die Dateinamen angewandt, die in der Datei *`datei`* stehen.
`-F` *`string`* **oder** `--separator` *`string`*	Anstatt des Separators »:« wird der mit *`string`* angegebene Separator verwendet.
`-i` **oder** `--mime`	Ausgabe von Strings im Mime-Typ.
`-k` **oder** `--keep-going`	Bei der ersten Übereinstimmung wird nicht angehalten.
`-L` **oder** `--dereference`	Symbolischen Links wird gefolgt.
`-m` *`liste`* **oder** `--magic-file` *`liste`*	Verwendet die Datei *`liste`* anstatt der Datei `/etc/magic`.
`--n` **oder** `--no-buffer`	Die Ausgabe wird nicht gepuffert.
`-N` **oder** `--no-pad`	Die Ausgabe wird nicht aufgefüllt.
`-s` **oder** `--special-files`	Behandelt spezielle Dateien (block- und zeichenorientierte Gerätedateien) als normale Dateien.
`-Z` **oder** `--uncompress`	Versucht, aus komprimierten Dateien weitere Informationen zu erhalten.

Tabelle 3.44: Optionen des Befehls `file`

Zum Beispiel:

```
$ file *
poem:          ascii text
save:          Korn shell script text
sammlung:      directory
testdatei:     empty
prot10:        symbolic link
prog99:       ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for
GNU/Linux 2.2.5, dynamically linked (uses shared libs), not stripped
```

►► **Stop:** *Das Ergebnis des Befehls stimmt nicht immer, da nur ein Teil der Datei ausgewertet wird.* ◄◄

Der Befehl wc

Dieser Befehl zählt Zeilen, Worte, Zeichen und Bytes in einer Datei:

`$ wc [-`*`option(en)`*`] `*`datei(en)`*

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-c` **oder** `--bytes`	Anzahl der Bytes ausgeben.
`-l` **oder** `--lines`	Anzahl der Zeilen ausgeben.
`-m` **oder** `--chars`	Anzahl der Zeichen ausgeben.

Tabelle 3.45: Optionen des Befehls `wc`

Option	Bedeutung
`-w` **oder** `--words`	Anzahl der Worte ausgeben.
`-L` **oder** `--max-line-length`	Länge der längsten Zeile ausgeben.

Tabelle 3.45: Optionen des Befehls `wc (Forts.)`

Zum Beispiel:

```
$ wc prot1000
      17      39      108          prot1000
```

In dieser Ausgabe werden die Zeilen (17), Worte (39) und Bytes (108) für die Datei `prot1000` angezeigt.

Der Befehl od

Der Befehl konvertiert Daten in andere Darstellungsformen, zum Beispiel in die hexadezimale Darstellung:

```
$ od [-option(en)] datei
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-A radix` **oder** `--address-radix=radix`	Steuerung der Ausgabe von Datei-Offsets: `d` **für dezimal,** `o` **für oktal,** `x` **für hexadezimal oder** `n` **für nichts.**
`-j bytes` **oder** `--skip-bytes= bytes`	Anzahl `bytes` **der am Dateianfang zu übergehenden Eingabebytes.**
`-N` **oder** `--read-bytes=bytes`	Begrenzen der Ausgabe auf `bytes` **Eingabebytes pro Datei.**
`-s` **oder** `--strings[=bytes]`	Ausgabe von Zeichenketten mit mindestens `bytes` alphanumerischen Zeichen.
`-t typ` **oder** `--format= typ`	Wählen des Ausgabeformats `typ`: ■ `a` = benanntes Zeichen ■ `c` = normales ASCII-Zeichen ■ `d`[*anzahl*] = Dezimalzahl mit Vorzeichen, *anzahl* Bytes pro Zahl ■ `f`[*anzahl*] = Fließkommazahl, *anzahl* Bytes pro Zahl ■ `o`[*anzahl*] = Oktalzahl, *anzahl* Bytes pro Zahl ■ `u`[*anzahl*] = Dezimalzahl ohne Vorzeichen, *anzahl* Bytes pro Zahl ■ `x`[*anzahl*] = Hexadezimalzahl, *anzahl* Bytes pro Zahl
`-v` **oder** `--output-duplicates`	Anzeige der Zeilenunterdrückung nicht mit dem Zeichen *.
`-w` **oder** `--width[=bytes]`	Ausgabe von Anzahl `bytes` **pro ausgegebener Zeile.**
`--traditional`	Argumente in traditioneller Form werden angenommen.

Tabelle 3.46: Optionen des Befehls `od`

Es kann auch eine ältere Syntax verwendet werden:

```
$ od --traditional [datei] [[+]offset [[+]marke]]
```

Dabei ist `marke` die Pseudoadresse des ersten Bytes, das ausgegeben wird, und wird bei fortschreitender Ausgabe entsprechend erhöht. Das Präfix ox oder oX ist hexadezimal für `offset` und `marke`, das Suffix ».« bedeutet oktal und »b« (block) multipliziert mit 512.

Werden traditionelle Formatangaben verwendet, so werden diese beim Mischen akkumuliert:

Option	Bedeutung
`-a` **oder** `-t a`	Wählen von benannten Zeichen.
`-b` **oder** `-t oC`	Wählen von Oktalbytes.
`-c` **oder** `-t c`	Wählen von ASCII-Zeichen.
`-d` **oder** `-t u2`	Wählen von dezimalen kurzen Zahlen ohne Vorzeichen.
`-f` **oder** `-t fF`	Wählen von Fließkommazahlen.
`-h` **oder** `-t x2` **oder** `-x`	Wählen von hexadezimalen kurzen Zahlen.
`-i` **oder** `-t d2`	Wählen von dezimalen kurzen Zahlen.
`-l` **oder** `-t d4`	Wählen von dezimalen langen Zahlen.
`-o` **oder** `-t 02`	Wählen von oktalen kurzen Zahlen.

Tabelle 3.47: Traditionelle Optionen des Befehls `od`

Zum Beispiel:

```
$ od -t cx1 obst
0000000    f   e   i   g   e  \n   z   i   t   r   o   n   e  \n   b   i
  66  65  69  67  65  0a  7a  69  74  72  6f  6e  65  0a  62  69
0000020    r   n   e  \n   m   e   l   o   n   e  \n   a   p   f   e   l
  72  6e  65  0a  6d  65  6c  6f  6e  65  0a  61  70  66  65  6c
0000040   \n   t   r   a   u   b   e   n  \n   m   a   n   d   a   r   i
...
```

Der Befehl basename

Der Befehl gibt Dateinamen ohne die führenden Pfadangaben aus:

```
$ basename datei [suffix]
```

Wird zusätzlich ein Suffix angegeben, wird dieses auch entfernt.
Im folgenden Beispiel wird der Pfadanteil eines Dateinamens entfernt:

```
$ basename /usr/lib/libjpeg.la
libjpeg.la
```

In diesem Beispiel wird zusätzlich noch das Suffix entfernt:

```
$ basename /usr/lib/libjpeg.la .la
libjpeg
```

Der Befehl dirname

Der Befehl gibt den Pfadanteil eines Dateinamens aus:

```
$ dirname datei
```

Im folgenden Beispiel wird der Pfadanteil eines Dateinamens extrahiert:

```
$ dirname /usr/lib/libjpeg.la
/usr/lib
```

Der Befehl sum

Dieser Befehl berechnet zu einer Datei eine Prüfsumme und die Anzahl von Blöcken und gibt diese Informationen aus. Diese Funktion ist nützlich, wenn Dateien übertragen wurden, um Übertragungsfehler festzustellen oder größere Dateien auf Gleichheit zu überprüfen. Die Syntax des Befehls lautet:

```
$ sum [-option(en)] datei
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-r`	Deaktiviert die Option -s. und verwendet den Prüfsummenalgorithmus von BSD-Unix mit 1 KByte großen Blöcken.
`-s` oder `--sysv`	Verwendet den Prüfsummenalgorithmus von System V-Unix und 512 Byte große Blöcke.

Tabelle 3.48: Optionen des Befehls `sum`

Hier wird die Prüfsumme und Blockgröße der Datei `liste` ausgegeben:

```
$ sum liste
47821  4
```

Wird die Datei verändert, dann ändert sich auch die Prüfsumme und gegebenenfalls die Größe in Blöcken.

Der Befehl whereis

Dieser Befehl sucht die Binär-, Quell- oder Hilfe-Dateien für einen Befehl oder eine Datei. Die Syntax des Befehls lautet:

```
$ whereis [-option(en)] datei
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-b`	Suche nur nach Binärdateien.
`-m`	Suche nur nach Einträgen in den Manpages.
`-s`	Suche nur nach Quelldateien.
`-u`	Suche nach ungewöhnlichen Einträgen. Eine Datei wird als ungewöhnlich bezeichnet, wenn sie keinen Eintrag des gewünschten Typs hat. Der Befehl `whereis -m -u *` sucht im aktuellen Verzeichnis nach allen Dateien, die nicht dokumentiert sind.
`-B` *dir*	Ändert oder begrenzt den Platz, an dem nach Binärdateien gesucht wird.
`-M` *dir*	Ändert oder begrenzt den Platz, an dem nach Einträgen in den Manpages gesucht wird.
`-S` *dir*	Ändert oder begrenzt den Platz, an dem nach Quelldateien gesucht wird.
`-f`	Beendet die letzte Verzeichnisliste und signalisiert den Beginn von Dateinamen. Diese Option muss immer nach einer der Optionen `-B`, `-M` oder `-S` verwendet werden.

Tabelle 3.49: Optionen des Befehls `whereis`

Im folgenden Beispiel wird nach dem Befehl `ls` gesucht:

```
$ whereis ls
ls: /bin/ls /usr/share/man/man1/ls.1.gz
```

Der Befehl xargs

Dieser Befehl führt den angegebenen Befehl mit beliebigen Argumenten aus, wobei er aber die übrigen Argumente nicht direkt angibt, sondern von der Standardeingabe liest. Auf diese Weise werden die Argumente in mehreren Teilen an den Befehl gegeben, sodass der Befehl mehr Argumente als üblich verarbeiten kann. Die Argumente bestehen in der Regel aus Listen mit Dateinamen, die mithilfe der Befehle `find` oder `ls` über eine Pipe an den Befehl `xargs` übergeben werden. Die Syntax des Befehls lautet:

```
$ xargs [-option(en)] befehl
```

Der Befehl kennt folgende Optionen:

Option	Bedeutung
`-0` oder `--null`	Eingabe-Dateinamen werden durch Null-Bytes anstelle von Leerzeichen beendet. Hochkommata und Backslash werden nicht besonders behandelt.
`-e[`*`string`*`]` oder `--eof[=`*`string`*`]`	Definiert EOF mit der angegebenen Zeichenkette *`string`* oder standardmäßig mit _.
`-i[`*`string`*`]` oder `--replace[=`*`string`*`]`	Ersetzt die angegebene Zeichenkette *`string`* oder standardmäßig »{}« mit von der Standardeingabe gelesenen Namen.
`-l[`*`zeilen`*`]` oder `--max-lines [=`*`zeilen`*`]`	Höchstens *`zeilen`* nicht leere Eingabezeilen dürfen pro Befehlszeile auftreten. Standardmäßig ist dieser Wert 1.
`-n` *`args`* oder `--max-args=`*`args`*	Höchstens *`args`* Argumente pro Befehlszeile dürfen verwendet werden.
`-p` oder `--interactive`	Bei jeder Befehlsausführung wird der Benutzer nach einer Bestätigung mit `y` gefragt.
`-P` *`max`* oder `--max-procs=`*`max`*	Höchstens *`max`* Prozesse dürfen zu einem Zeitpunkt laufen, der Standardwert ist 1. 0 bedeutet, dass so viel wie möglich Prozesse gleichzeitig laufen.
`-r` oder `--no-run-if-empty`	Wenn die Standardeingabe nur aus Leerzeichen besteht, wird der Befehl nicht gestartet.
`-s` *`max`* oder `--max-chars=`*`max`*	Höchstens *`max`* Zeichen pro Befehlszeile dürfen verwendet werden.
`-v` oder `--verbose`	Die Befehlszeile wird an der Standardfehlerausgabe vor der Ausführung ausgegeben.
`-x` oder `--exit`	Beendet den Befehl, wenn die maximale Länge der Befehlszeile, wie von der Option `-s` festgelegt, überschritten wird.

Tabelle 3.50: Optionen des Befehls `xargs`

Im folgenden Beispiel werden alle Dateien des Verzeichnisses `/home` nach dem Muster `Quote` durchsucht und die Übereinstimmungen werden in die Datei `ausgabe` geschrieben:

```
$ find /home -print | xargs grep Quote > ausgabe &
```