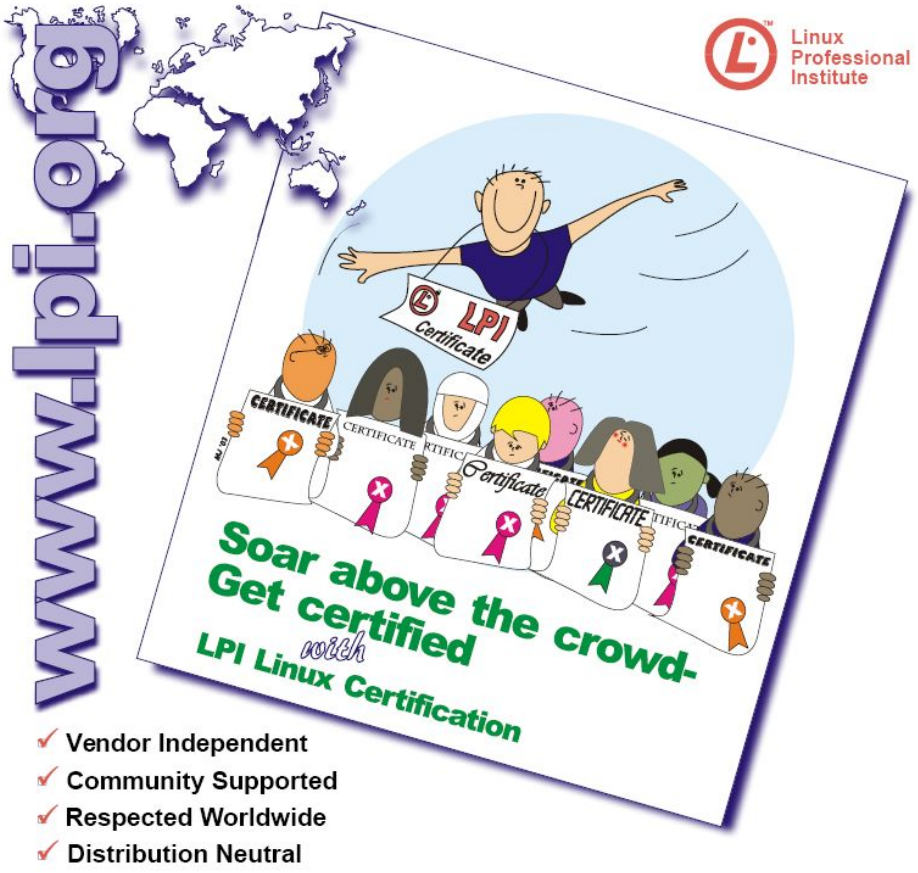
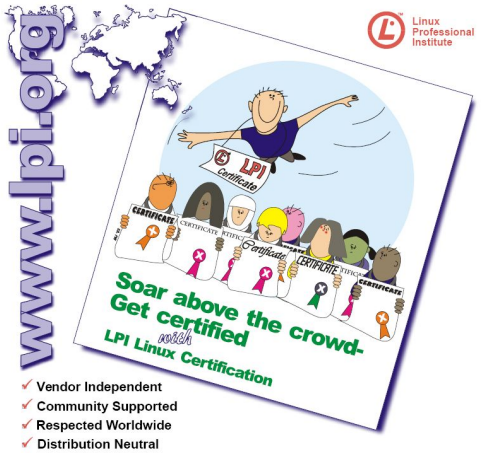
1.103.2 Texte mit Filterprogrammen bearbeiten



1.103.2 Texte mit Filterprogrammen bearbeiten



Quellen:

http://www.lpi.org

http://www.lpi-test.de

Dean, Jeffrey: LPI Linux Certification (LPI Level1), O'Reilly

1.103.2 Texte filtern

Beschreibung: Prüfungskandidaten sollten in der Lage sein, Filter auf Textströme anzuwenden. Dieses Lernziel beinhaltet das Senden von Textdateien und -ausgaben durch Textfilterprogramme, um die Ausgabe zu modifizieren, und die Verwendung von Standard-Unix-Kommandos, die im GNU textutils Paket enthalten sind.
Die wichtigsten Dateien, Bezeichnungen und Anwendungen:

cat
cut
expand
fmt
head
hexdump*)
join
nl
paste
pr
sed*)
sort
split
tac
tail
tr
unexpand
uniq
wc

*) Eigentlich nicht aus GNU Ex-Package "*Textutils*"

Weitere Kommandos aus dem ehemaligen GNU Package '*textutils*':

chksum
comm
csplit
fold
md5sum
od
ptx
sum
tsort

Bei den meisten Befehlen gilt:

befehl liest die angegebenen Dateien oder die Standardeingabe (wenn keine Datei oder anstelle einer Datei „-“ angegeben ist), und gibt das Ergebnis auf der Standardausgabe aus.

Beispiele:

befehl datei	Lesen aus Datei
befehl -	Lesen von Standardeingabe (i.d.R. Tastatur)
	Ausgabe in beiden Fällen auf die Standardausgabe (i.d.R. die Konsole)
befehl - > ausdatei	Umleitung der Ausgabe in die Datei "ausdatei"
befehl1 - | befehl2	Die Ausgabe von befehl1 wird weitergeleitet zur Standardeingabe von befehl2 (piping)

cat

concatenate files and print on the standard output

Syntax: cat [OPTION] [FILE]...

Optionen:

-b, --number-nonblank	Nummeriere alle Zeilen, die nicht NULL sind
-n, --number	Nummeriere alle Zeilen
-t	zeige alle TABs als ^I
-s, --squeeze-blank	nie mehr als eine Leerzeile in der Ausgabe

Beispiele:

cat -n datei

cat datei1 datei2 datei3

cat datei1 datei2 datei3 > datei4

tac

... die Umkehrung von cat. Die Ausgabe startet pro Datei mit den untersten Zeilen und endet mit der 1. Zeile. Reihenfolge der Dateien in der Ausgabe ändert sich nicht!

Syntax: tac [OPTION] [FILE]...

Optionen:

-s, --separator=STRING	Benutze STRING als Trenner anstelle von Newline

Beispiele:

tac datei

tac datei1 datei2 datei3

tac datei1 datei2 datei3 > datei4

head

... schreibt den Anfang einer Datei (die ersten 10 Zeilen) auf die Standardausgabe

Syntax: head [OPTION]... [FILE]...

Optionen:

-c, --bytes=[-]N	die ersten N Bytes jeder Datei ausgeben; mit führendem „-“, alle außer den letzten N Bytes jeder Datei
-n, --lines=[-]N	s.o. mit Zeilen

N kann Suffixe (b 512, k 1024, m 1024*1024) enthalten

Beispiele:

head -n -3 datei	Alle Zeilen, außer den 3 letzten.
head datei1 datei2	Wird mehr als eine Datei angegeben, so wird der Dateiname, in ==> und <== eingeschlossen, der Ausgabe vorangestellt.

tail

... schreibt das Ende einer Datei (die letzten 10 Zeilen) auf die Standardausgabe

Syntax: tail [OPTION]... [FILE]...

Optionen wie bei **head**, jedoch zusätzlich:

-f, --follow	aktualisiert ständig den Zuwachs einer Datei
--retry	probiert es immer wieder, auch wenn auf Datei (vorübergehend) nicht zugegriffen werden kann.

Beispiele:

tail -f datei	
tail -c 3k datei	Gibt die letzten 3 Kbyte der Datei aus.

expand

... ersetzt Tabulatorzeichen durch Folgen von Leerzeichen. Dabei werden in der Ausgabe Tabulatorpositionen gesetzt und die Tabulatoren in der Quelldatei mit einer passenden Anzahl von Leerzeichen ersetzt.

Syntax: expand [OPTION]... [FILE]...

Optionen:

-t, --tabs=ZAHL	setzt die Tabellenbreite mittels Tabulatorpositionen für alle Spalten auf ZAHL anstelle von 8 (Standardwert)
-t, --tabs=LISTE	durch Komma getrennte LISTE von Tabulatorpositionen annehmen
-i, --initial	konvertiert nur die Tabulatoren, die am Zeilenanfang sind

Beispiele:

expand -t 20,69 datei	Tabulatorposition bei 20 und bei 69 wählen

unexpand

... ersetzt Leerzeichen durch Tabulatorzeichen (Umkehr von expand)

Syntax: unexpand [OPTION]... [FILE]...

Optionen:

-a, --all	Alle Leerzeichen, nicht nur die am Zeilenanfang!
--first-only	Nur Leerzeichen am Zeilenanfang (überschreibt -a)
-t, --tabs=N	Nimm N Leerzeichen statt 8 für einen TAB (-a ist automatisch aktiviert)
-t, --tabs=LIST	durch Komma getrennte LISTE von Tabulatorpositionen annehmen (-a ist automatisch aktiviert)

Beispiele:

unexpand -t 20 datei	Tabulatoren befinden sich in 20er Schrittweite. Macht dann Sinn, wenn zuvor Datei mit expand -t 20 erzeugt wurde.

Übungen 1

Machen Sie die Übungen in

1.103.2 Filterprogramme Übungen1

nl

... nummeriert die Zeilen in einer Datei

Syntax: nl [OPTION]... [FILE]...

nl gibt die Zeilen einer oder mehrerer Dateien (oder der Standardeingabe) mit Zeilennummern auf die Standardausgabe. Es können dabei die Zeilen einer (logischen) Seite in einen Kopf, einen Körper und einen Fuß unterteilt werden, die jeweils einzeln und in unterschiedlichen Stilen nummeriert werden.

Default-Optionen: -v1 -i1 -l1 -sTAB -w6 -nrn -hn -bt -fn

-v1	1. Zeilennummer pro logischer Seite
-i1	Inkrement pro Zeile
-l1	Jede einzelne Leerzeile wird nummeriert (kein Squeezing)
-sTAB	Trennzeichen zw. Zeilennummer und Text (hier ein Tabulator)
-w6	Breite der Spalte mit den Zeilennummern
-nrn	Zeilennummern rechtsbündig ohne führende Nullen
-hn	Header-Stil: nicht nummerieren
-bt	Body-Stil: die nicht leeren Zeilen nummerieren
-fn	Footer-Stil: nicht nummerieren

nl

Beispiel-Textdatei:

```
\:\:\:
header
\:\:
line1
line2
line3
\:
footer
\:\:\:
header
\:\:
line1
line2
line3
\:
footer
```

Optionen:

-h [a|t|n]	Setze Header-Stil auf a, t oder n.
-b [a|t|n]	Setze Body-Stil auf a, t oder n.
-f [a|t|n]	Setze Footer-Stil auf a, t oder n.

wobei:

a	Nummeriere alle Zeilen
t	Nummeriere nur nicht-leere Zeilen
n	Nummeriere nicht
\:\:\:	Kennung Header
\:\:	Kennung Body
\:	Kennung Footer

Beispiel:

```
nl -h a datei
```

fmt

Jeden Absatz eines Textes neu formatieren und auf STDOUT ausgeben.

Syntax: fmt [OPTION]... [FILE]...

Optionen:

-u	Aufeinanderfolgende Leerzeichen zusammenfassen. Ein Leerzeichen zwischen Worten, zwei zwischen Sätzen.
-w *weite*	Setze Zeilenbreite auf *weite* (Voreingestellt sind 75 Zeichen)

Bei Angabe von mehreren Dateien, werden diese aneinander gehängt.

Beispiele:

fmt -w 40 datei

join

... verknüpft zwei Dateien nach Schlüsselfeldern

Syntax: join [optionen] datei1 datei2

join verknüpft zwei (alphabetisch) sortierte Dateien, indem je zwei Zeilen mit identischen Schlüsselfeldern zu einer Ausgabezeile verbunden werden. Die Schlüsselfelder sind durch Leerzeichen voneinander getrennt. Führende Leerzeichen werden ignoriert. Wenn nicht anders angegeben, ist das erste Feld einer jeden Zeile Schlüsselfeld. Die Ausgabefelder sind ebenfalls durch Leerzeichen voneinander getrennt. Die Ausgabe besteht aus dem Schlüsselfeld, gefolgt von den übrigen Feldern der Datei1 und schließlich aller Felder der passenden Zeilen von Datei2 ohne das Schlüsselfeld.

Optionen:

-j1 *feldnr* Verknüpft die Dateien nach *feldnr* in *datei1*

-j2 *feldnr* Verknüpft die Dateien nach *feldnr* in *datei2*

-j *feldnr* Verknüpft die Dateien nach *feldnr,* in beiden Dateien vorh.

join

Beispiel

file1:

1 eins

2 zwei

3 drei

file2:

1 11 11

2 22 22

2,5 dummy

3 33 33

4 44 44

join -j 1 file1 file2

liefert:

1 eins 11 11

2 zwei 22 22

3 drei 33 33

paste

... fügt die Zeilen von zwei oder mehr Dateien horizontal zusammen

Syntax: paste [optionen] dateien

paste fügt die Zeilen mehrerer Dateien zusammen. Die Zeilen werden standardmäßig durch tab getrennt und die Ausgabe einer kompletten Zeile (das heißt die Ausgabe der entsprechenden Zeilen aller Dateien) wird mit einem Zeilenende abgeschlossen.

Optionen:

-d *'n'*	Spaltentrenner ist das Zeichen *n*, anstelle von tab
-s	(serial) Bringe alle Zeilen einer Datei auf eine einzige Zeile in der Ausgabe. Sind mehrere Dateien angegeben, so gibt es in der Ausgabe je eine Zeile für jede Datei.

paste

Beispiel:

file1:

```
1
2
3
```

file2:

```
A
B
C
```

paste file1 file2

liefert:

```
1   A
2   B
3   C
```

paste -s file1 file2

liefert:

```
1   2   3
A   B   C
```

Eine ungleiche Anzahl von Zeilen in den Dateien wird durch Anfügen von Leerzeilen ausgeglichen.

sort

Syntax: sort [OPTION]... [DATEI]...

... die Aneinanderfügung aller DATEI(en) sortiert nach der Standardausgabe schreiben.

Optionen:

-b, --ignore-leading-blanks	führende Leerzeichen ignorieren
-f, --ignore-case	Klein- als Großbuchstaben behandeln
-i, --ignore-nonprinting	nur druckbare Zeichen beachten
-n, --numeric-sort	anhand des numerischen Werts sortieren
-r, --reverse	das Ergebnis der Sortierung umkehren
-o, --output=DATEI	Ergebnis in DATEI schreiben statt Standardausgabe
-k, --key=POS1[,POS2]	Schlüssel geht von POS1 bis POS 2 (beginnend mit 1)
-t, --field-separator=SEP	SEP als Trennzeichen statt den Übergang von Nichtleerz. zu Leerzeichen benutzen

Beispiele:

```
sort datei
sort -n datei -o ausdatei
```

split

... spaltet eine Datei in mehrere kleinere

Syntax: split [option] infile [outfile]

split teilt eine Datei in mehrere Teile. Wenn keine weiteren Optionen gegeben sind, wird die Datei in Teile zu je 1000 Zeilen aufgeteilt. Die Ausgabe erfolgt in die Dateien outfileaa, outfileab, outfileac, usw, oder xaa, xab, xac, wenn kein Dateiname angegeben wird.

Optionen:

-*Zeilen*	Anzahl der Zeilen in jeder Ausgabedatei
-l *Zeilen*	Anzahl der Zeilen in jeder Ausgabedatei
-d	Numerische Suffixe, statt Buchstaben
-b *Bytes*[bkm]	Anzahl der Bytes in jeder Ausgabedatei b=512, k=1024, m=1024^2

split

Beispiel:

file1:

```
1 eins
2 zwei
3 drei
4 vier
5 fuenf
6 sechs
```

split -2 file1 splitout_

liefert

splitout_aa	1 eins 2 zwei
splitout_ab	3 drei 4 vier
splitout_ac	5 fuenf 6 sechs

split -b650m Datei Datei

Aufteilen einer großen Datei in 650 MB – Stücke Dateiaa, Dateiab, usw.

cut

... schneidet Teile aus den Zeilen einer Datei aus und zeigt sie an.

Syntax:
```
cut -b  [Datei...]
cut -c  [Datei...]
cut -f  [-d Trenner] [-s] [Datei...]
```

Optionen:

-b, --bytes=LISTE	Wähle diese Bytes (Position in Zeile)
-c, --characters=LISTE	Wähle diese Zeichen der aufgelisteten Spalten (Position in Zeile)
-f, --fields=LISTE	Wähle diese Felder
-d, --delimiter=DELIM	Benutze DELIM anstelle von TAB als Feldtrenner

Beispiele:

```
cut -d: -f 1,5 /etc/passwd
cut -c 1,2,3,4,20 /etc/passwd
cut -c 1-5,20-22,30- /etc/passwd
```

tr

... löscht oder ändert (transliteriert) einzelne Zeichen.

Syntax: tr [OPTION]... String1 [String2]

Übersetzt und/oder löscht Zeichen aus der Standard-Eingabe und schreibt auf die Standard-Ausgabe.

Optionen:

-d, --delete	Lösche die Zeichen in String1
-s, --squeeze-repeats	Unterdrücke sich wiederholende Zeichen

Beispiele:

cat datei | tr a-z A-Z oder tr a-z A-Z < datei

cat datei | tr -s a

cat datei | tr -d axuv Löscht alle a,x,u und v

WC

... zählt die Anzahl von Zeilen, Worten, Zeichen. Die Ausgabe erfolgt in dieser Reihenfolge.

Syntax: wc [optionen] [Datei...]

Optionen:

-c, --chars	Anzahl der Zeichen (Bytes)
-l, --lines	Anzahl der Zeilen
-w, --words	Anzahl der Worte

Ohne Optionen gibt wc alle drei Werte aus.

Beispiele:

file1, file2, file3

wc file[123]

wc -l file1

uniq

Alle hintereinander stehenden identischen Zeilen von EINGABE (oder Standardeingabe) bis auf eine löschen, und auf AUSGABE (oder Standardausgabe) schreiben.

Syntax: uniq [OPTION]... [EINGABE [AUSGABE]]

Optionen:

-c, --count	den Zeilen die Anzahl des Vorkommens voranstellen
-d, --repeated	nur die doppelten Zeilen ausgeben
-u, --unique	nur einmal vorkommende Zeilen ausgeben
-f, --skip-fields=N	nicht die ersten N Felder vergleichen
-s, --skip-chars=N	nicht die ersten N Zeichen vergleichen
-w, --check-chars=N	nicht mehr als N Zeichen pro Zeile vergleichen
-i, --ignore-case	Abweichung in Groß/Kleinschreibung ignorieren

Beispiele:

```
uniq -c datei

uniq -d eindatei ausdatei
```

pr

... produziert seitenweise nummerierte und einfach formatierte Ausgabe von Textdateien.

Syntax: pr [OPTIONEN]... [DATEI]

Optionen:

-d	Doppelter Zeilenabstand zwischen Absätzen
-h header	Benutze header anstelle des Dateinamens im Kopf
-l lines	Setze Seitenlänge auf lines (Standard ist 66)
-o width	Setze linken Rand auf width

Beispiele:
pr -o 4 -h "Das Leben von Galileo Galilei" langer-text

xargs

Syntax: xargs [options] [command] [initiale Argumente]

Führe den Befehl *command*, gefolgt von seinen initialen Argumenten aus und füge dabei zusätzliche Argumente aus der Standardeingabe hinzu. Typischerweise sind dies Dateinamen, welche in ihrer Anzahl zu hoch wären für die Befehlszeile. xargs sorgt dafür, dass *command* so oft ausgeführt wird, bis alle Argumente aus der Standardeingabe abgearbeitet sind.

Optionen:

-n *maxargs*	Begrenzt für jeden Programmaufruf die Anzahl der zusätzlichen Argumente auf *maxargs*
-p	Interaktiver Modus

Beispiele:

```
find /etc -type d | xargs ls -l > dummy
```

hexdump, hd

Syntax: hexdump [options] [datei]...

Optionen:

-b One-byte octal display
-c One-byte character display
-C Canonical hex+ASCII display
-d Two-byte decimal display
-o Two-byte octal display
-x Two-byte hexadecimal display

Beispiele:

hexdump -C datei

Übungen 2

Machen Sie die Übungen in

1.103.2 Filterprogramme Übungen2