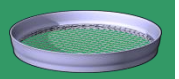
LinuxCampus.net
trainings from the experts

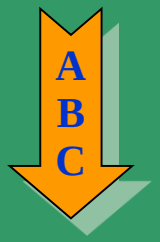
1100 Linux 1 Grundlagen

Einfache Textverarbeitungsfilter

Peter Jahn, MSc
System Engineer
training@LinuxCampus.net

Kurs Agenda



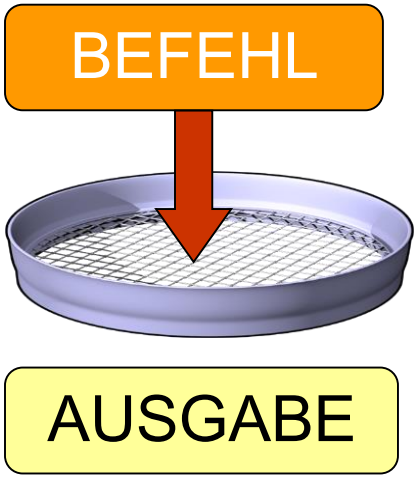


Linux: Einfache Textverarbeitungsfilter

- Filtern von Dateiinhalten
- Übersetzen von Dateiinhalten
- Sortieren von Ausgaben
- Vergleichen von Inhalten

Textfilter

- Was ist ein Textverarbeitungsfilter?
  - Ein Programm, das Datenströme einliest, bearbeitet und wieder ausgibt
  - Die Ausgabe kann auf dem Bildschirm ausgegeben oder in eine neue Datei geschrieben werden



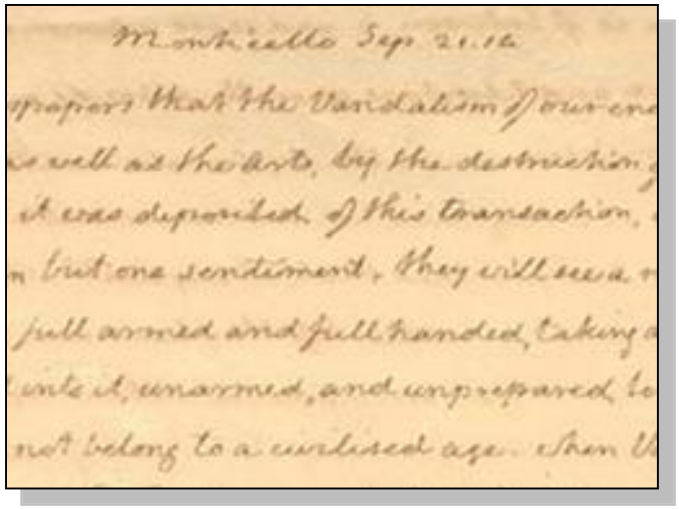
Auszug Filterprogramme

- grep ... Sucht nach Textmustern
- sed ... Suchen & Ersetzen
- wc ... Zählen von Zeichen, Zeilen, Wörtern
- sort ... Sortierten der Ausgabe
- tr ... Übersetzen von Zeichen
- diff ... Vergleichen von Dateiinhalten
- fmt ... Textausgabenbreite festlegen

grep

GREP

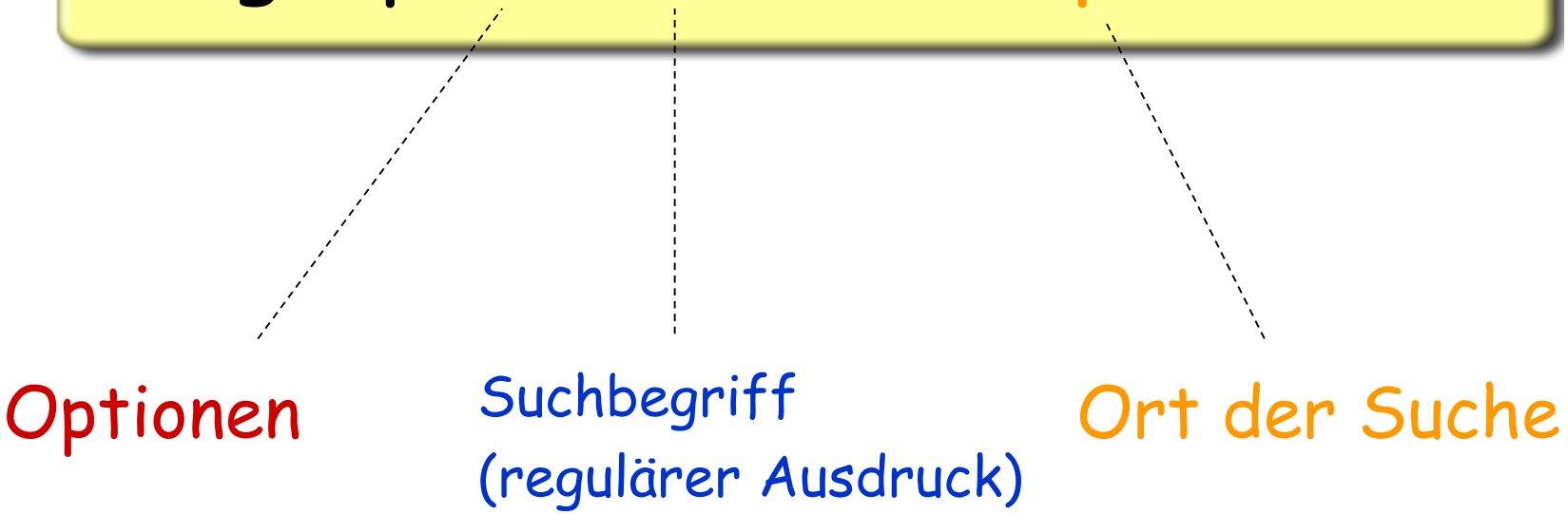
- GREP - "Get Regular Expression"
  - sucht nach speziellen Textmustern in Dateien
  - Sucht nach Textmustern in Shellausgaben
  - Textmuster sind meist Reguläre Ausdrücke



- Anzahl des Wortes "Linux"
- Alle Zeilen die mit "Linux" beginnen
- Alle Zeilen die kein "Linux" enthalten
- Alle Wörter die ...Linux... enthalten

Grep Syntax

```
grep -i root /etc/passwd
```

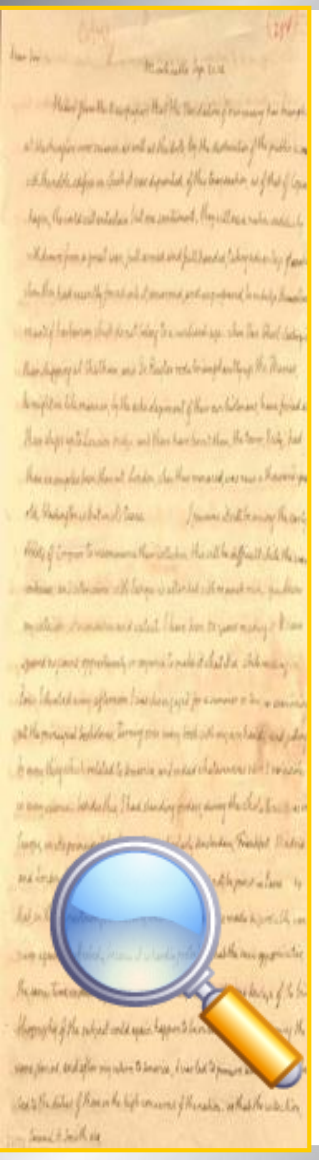


Grep Optionen

```
grep -i root /etc/passwd
```

Option	Bedeutung
-c	Ausgabe der Zeilenanzahl
-i	Groß- und Kleinschreibung ignorieren
-l	nur den Dateinamen anzeigen
-n	zusätzlich die Zeilennummer ausgeben
-s	keine Fehlermeldungen ausgeben (supress)
-v	Nur Zeilen die nicht das Suchmuster enthalten
-r	Rekursiv suchen
-w	Das Suchmuster muss ein eigenes Wort sein
-E/F/P	In welchem Modi soll grep gestartet werden

Beispiele



```
#Gibt es einen Benutzer ROOT?
peter@tux> grep root  /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

```
# Wieviele Benutzer verwenden die Shell /bin/bash
peter@tux> grep -c /bin/bash  /etc/passwd
14
```

```
#besteht eine FTP- Verbindung?
peter@tux> netstat -n | grep :21
tcp  0  172.16.3.33:21   172.16.3.10:43956  ESTABLISHED
```

```
# Rekursives Suchen?
peter@tux> grep  -r  nameserver  /etc
/etc/services: nameserver   42/udp  # Host Name Server
/etc/init.d/named: echo "Checking nameserver BIND "
/etc/resolv.conf: nameserver 172.16.0.1
```

Grep & Pipes

- Suchen nach dem init-Prozess
  - Resultat zeigt immer auch den Prozess für die Suche

```
#Suche nach dem Prozess init
peter@tux> ps ax | grep init
1        S    init
20456    S+   grep init
```

- Verbesserte Suche
  - ps ax | grep init | grep -v grep

Ausgabe der Dateinamen

- Nur die Dateinamen anzeigen
  - Anstatt dem Inhalt können auch nur die Dateien aufgelistet werden in denen das Muster gefunden wurde
  - grep -l (kleines L)

```
#Ausgabe der Dateien
peter@tux#  grep -l root /etc/*
/etc/aliases
/etc/aliases.db
/etc/auditd.conf
...
```

Ausgabe des Strings

- Nur den gefunden String anzeigen
  - Anstatt der kompletten Zeile kann auch nur der gefunden String ausgegeben werden
  - Hilfreich beim testen von regulären Ausdrücken
  - grep -o

```
#Ausgabe der Strings
peter@tux#  grep -o root /etc/fstab
root
root
root
```

Ausgabe der Zeilen Position

- Zusätzlich die Zeilen Position anzeigen
  - Es kann auch vor jedes Ergebnis die Zeilennummer ausgegeben werden
  - grep -n

```
#Ausgabe der Zeilennummer
peter@tux#  grep -n false /etc/passwd
5:mail:x:8:12:Mailer daemon:/var/spool/clientmqueue:/bin/false
9:messagebus:x:100:101:User for D-BUS:/var/run/dbus:/bin/false
10:haldaemon:x:101:102:User for haldaemon:/var/run/hal:/bin/false
```

Ausgabenumfang

- Umfang der Ausgabe
  - Normalerweise wird nur die Zeile mit dem gefunden Muster ausgegeben
  - -A # ...Display # lines after match
  - -B # ... Display # lines before match
  - -C # ... Display # lines before & after match

Reguläre Ausdrücke

Reguläre Ausdrücke

- Was sind Reguläre Ausdrücke
  - Ein regulärer Ausdruck ist ein Suchbegriff
  - Der Suchbegriff ist ein spezielles Textmuster
    - Zeile beinhaltet das Wort hello
    - Textmuster beginnt mit der Zahl 1 gefolgt von 4 Zeichen (10000, 12345, 1abcd)
  - Das Textmuster steht an einer bestimmten Stelle
    - Zeile beginnt/endet mit dem Wort hello

Reguläre Ausdrücke

- Wichtiges zu Regulären Ausdrücken
  - * und ? haben eine ganz andere Bedeutung
  - anders als Wildcards in Dateinamen, muss ein regulärer Ausdruck immer nur auf einen Teil des Textes passen
- Anwendungsgebiete
  - Editoren wie vi, emacs
  - Programme wie grep, egrep, awk, sed
  - Programmiersprachen wie Perl, Java, VB.NET,...

Auszug aus den Metazeichen

BEDEUTUNG		BEISPIEL
.	Ein beliebiges Zeichen, außer Zeilenumbruch	Lin . x
[ ]	Ein Zeichen aus der Menge [Klammerninhalt]	[0-9] [adgo]
*	Beliebige Wiederholung des vorstehenden Musters	[0-9]* a*
?	Null oder ein Vorkommen des vorherigen Musters	[0-9]\? a\?
^	Das Muster steht am Anfang der Zeile/Zeichenkette	^TripleS
$	Das Muster steht am Ende der Zeile/Zeichenkette	TripleS$
[^]	Ein Zeichen außer das in der Klammer	[^0-9]
re1|re2	ODER; Das eine oder das andere Muster	abc|123
(ab)+	Mindestens ein Auftreten der Menge "ab"	(ab)+
+	Eines oder mehrere Vorkommen	[0-9]+ a+

Meistens müssen die regulären Ausdrücke in '...' geschrieben werden

Syntax Highlighting

- Grep Syntax Highlighting
  - Auf manchen Distributionen werden die gefunden Textstellen automatisch farblich hervorgehoben

```
# Grep mit Syntax Highlighting
peter@tux #  grep error /var/log/messages
Apr  2 22:12:46 linux kernel: end_request: I/O error, dev fd0, sector 0
Apr  2 22:12:46 linux kernel: end_request: I/O error, dev fd0, sector 0
```

  - Dieses Verhalten kann auch manuell aktiviert werden
    - export GREP_OPTIONS='--color=auto'

Automatisches Syntax Highlighting

- Variante 1: Alias
  - Definieren eines Alias für grep

```
# ~/.bashrc   oder ~/.alias
alias grep="grep  --color=auto"
```

- Variante 2: Grep Optionen
  - Definieren von Standardoptionen für grep

```
# ~/.bashrc
export  GREP_OPTIONS='--color=auto'
```

Das Metazeichen .

```
 # Beispiel
 peter@tux # grep  'r..t'  /etc/passwd
root:x:0:0:root:/root:/bin/bash
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
```

- 'r..t'
  - Jeder Punkt steht für 1 beliebiges Zeichen
    - es muss aber auch 1 Zeichen an der Stelle vorkommen

Das Metazeichen ^

```
# Beispiel
peter@tux # grep  '^r..t'  /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

- '^r..t'
  - Die Zeile beginnt mit dem Muster 'r..t'
    - Wenn ein Abstand vor dem 'r' währe, würde das Muster nicht mehr angezeigt werden

Das Metazeichen [ ]

```
# Beispiel
peter@tux # grep -c '5[0-9]' /etc/passwd
4
peter@tux # grep -c  '5[0-9]*' /etc/passwd
7
```

- 5[0-9]
  - Die Zahl 5 gefolgt von 1er Zahl aus dem Bereich 0-9
- 5[0-9]*
  - wie oben jedoch darf die Zahl 0-9
    - 0-mal, 1-mal oder X-mal vorkommen

Kombination von Metazeichen

```
 # Zeile beginnt mit   1: oder  2:  oder  3:
 peter@tux# grep  ^[123]:  /etc/inittab
1:2345:respawn:/sbin/mingetty --noclear tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
```

- ^[123]:
  - Zeile beginnt mit der Zahl 1,2 oder 3
  - direkt gefolgt von einem Doppelpunkt
    - 1: ...ist gültig
    - 12: ...ist nicht gültig

Pattern1 ODER Pattern2

```
# Suchen von 2 unterschiedlichen Pattern
peter@tux> grep 'proc\|floppy' /etc/fstab
proc            /proc           proc    defaults        0 0
usbfs           /proc/bus/usb   usbfs   noauto          0 0
/dev/fd0        /media/floppy   auto    noauto,user,sync  0 0
```

- grep 'pattern1\|pattern2' filename
  - Bei der Verwendung von basic regexp verlieren die Meta characters ?, +, {, |, (,) ihre Bedeutung und müssen daher durch einen Backslash maskiert werden

Das Metazeichen ?

```
# Beispiel
peter@tux # grep 'r\?.t' /etc/passwd
root:x:0:0:root:/root:/bin/bash
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
```

- 'r\?.t'
  - Das Zeichen ? steht für 0 oder X Zeichen
    - Es muss jedoch mit einem \ vor der Bash geschützt werden damit es bei basic regex funktioniert

Sucher nach der Position

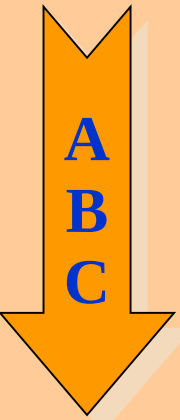
- Suche mit Positionsangaben
  - grep '^hello' … Zeile beginnt mit hello
  - grep '^[Hh]ello' … Zeile beginnt mit Hello/hello
  - grep '^\.' … Zeile beginnt mit einem Punkt
  - grep '^[^.]' … Zeile beginnt nicht mit einem Punkt
  - grep 'hello$' … Zeile endet mit hello
  - grep '^hello$' … einziges Wort in der Zeile
  - grep 'hello\>' … Wort endet mit …hello
  - grep '\<hello' … Wort beginnt mit hello…

GREP Beispiele

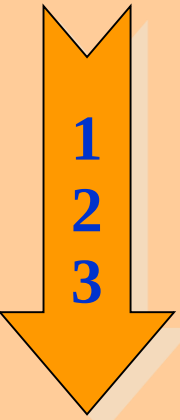
- Details zu Benutzer Peter anzeigen
  - grep '^peter:' /etc/passwd
- Alle Zeilen die nicht mit # beginnen anzeigen
  - grep -v '^#' /etc/hosts
    - Alles anzeigen das nicht dem Suchmuster entspricht
- Hat es beim booten Probleme gegeben
  - grep -i 'error' /var/log/boot.msg
    - Gross und Kleinschreibung ignorieren

Ausgabe sortieren

Sortierte Ausgabe



- Textausgabe alphabetisch sortiert ausgeben
  - sort textdatei …Inhalt sortieren
  - befehl | sort …Befehlsausgabe sortieren




```
# Inhalt sortiert ausgeben
peter@tux>  sort  Liste.txt
Apfel
Banane
Karotten
Milch
Zitronen
...
```

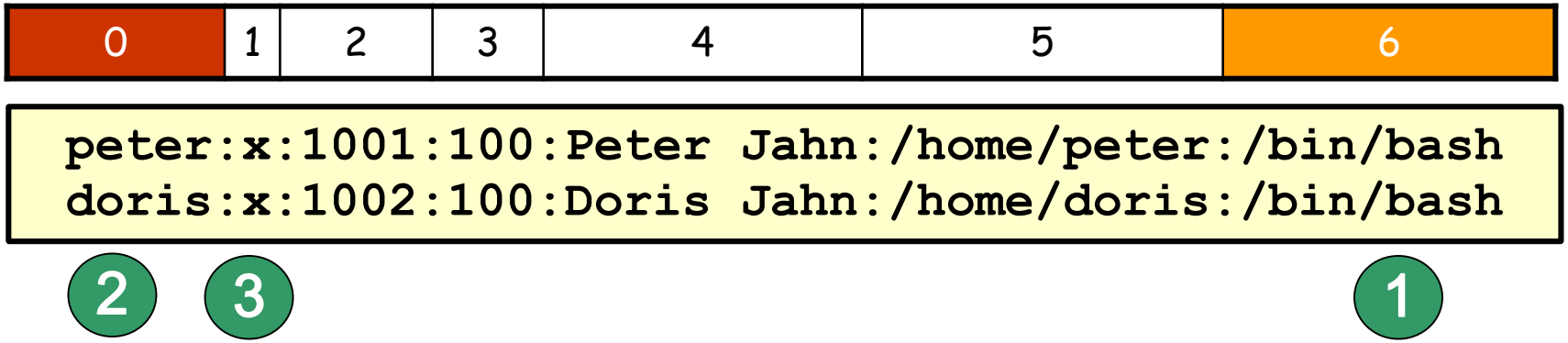
```
# Ausgabe sortiert ausgeben
peter@tux>  ls /  | sort
bin
boot
dev
etc
home
...
```

Sortieren

```
sort -r /etc/passwd
```

SORT OPTIONEN	
-f	Ignoriert Groß- & Kleinschreibung
-r	Sortiert absteigend
-t x	x ist das Trennzeichen
-n	anhand des numerischen Wertes sortieren
-b	ignoriert führende Leerzeichen

Sortieren



- Sortierreihenfolge bestimmen

  sort -t: +6 +0 -1 /etc/passwd

  1. Zuerst nach der Login Shell sortieren (+6)
  2. Danach nach dem Benutzernamen (+0) sortieren
  3. Ab Feld 1 nicht mehr (-1) sortieren

Numerisch sortieren

- ls -i | sort
  - sortiert nach den einzelnen Zahlen
- ls -i | sort -n
  - sortiert nach dem kompletten Zahlenwert

```
# Nach dem Zahlenwert (Inode) sortieren
peter@tux>  ls -i | sort -n
     5    .xinitrc.template
    10    bin
    56    public_html
    ...
```

Sortieren

```
# Normale Ausgabe eines Verzeichnisses
peter@tux>  ls -l
-rwxrwxrwx   1   jahn   users    234000  2005-02-20 09:34  video.txt
-rwxrwxrwx   1   jahn   users     12000  2005-02-20 09:38  liste.txt
```

A
B
C

1
2
3

- Sortierte Ausgabe eines Verzeichnisses
  - ls -l | sort
  - ls -l | sort -n +4 -5

+4 ...Startfeld für Sortierung

-5 ...Wo soll die Sortierung aufhören

Sort

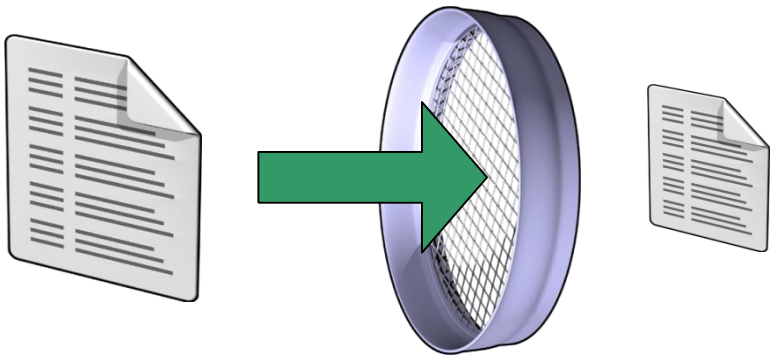
- Speicherverbrauch ermitteln
  - Welche 10 Benutzer haben die meisten Daten in ihrem Heimatverzeichnis?
  - du -cks /home/* | sort -rn | head

```
# Wer verbraucht am meisten Platz im Heimatverzeichnis
peter@tux>  du -cks /home/* | sort -rn | head
42923462    /home/peter
348982      /home/franz
789         /home/karl
...
```

Suchen und Ersetzen mit "sed"

Suchen & Ersetzen

- Was ist sed?
  - ein Stream Editor der als Filter arbeitet
  - hilft beim automatischen Verändern von Dateiinhalten
  - sucht nach Zeichen und ersetzt diese
  - wird sehr oft in Shell Scripts verwendet



Eingabe

- Datenstrom
  - sed empfängt auf seinem Eingabekanal die Daten

```
sed 'Kommando' 0< Datei
```

```
sed 'Kommando' Datei
```

```
befehl | sed 'Kommando'
```

  - Jede Zeile wird in einem Buffer abgelegt

sed Ablauf

- Vorgehensweise
  - Zeile für Zeile liest sed von der Standardeingabe (Schleife)
    - kopiert die einzelne Zeile in einem Buffer (Patternspace)
    - bearbeitet den Buffer (Aktionen)
    - gibt den überarbeiteten Buffer auf der Standardausgabe wieder aus

SED-Skripte

```
sed 'xxx' eingabe
```

Scripte	Bedeutung
p	Adressierte Bereiche ausgeben (print)
d	Adressierte Bereiche löschen (delete)
s	Ersetzt gefundenes Suchmuster (substitute)
a	Text nach einer Position anhängen (append)
i	Text vor einer Position einfügen (insert)
c	Ersetzt adressierte Bereiche (change)
r	Liest eine Datei ein und setzt den Inhalt hinter einen Bereich
w	Speichert den adressierten Bereich in einer Datei (write)
g	Aktion auf die ganze Zeile anwenden (global)

Löschen mit sed

```
sed 'xxx' eingabe
```

- Zeile X aus dem Datenstrom löschen
  - '3d' ...dritte Zeile
  - '$d' ...letzte Zeile

- Zeilen von,bis aus dem Datenstrom löschen
  - '1,3d' ...Zeile 1-3
  - '2,$d' ...Zeile 2 bis letzte Zeile
  - '1,/STOP/d' ...1 Zeile bis zur Zeile mit dem Wort STOP

Löschen mit sed

- Löschoptionen
  - d ...Zeile löschen auf die Muster passt
  - !d ...Alles löschen außer die Zeilen auf die das Muster passt

```
# Alles löschen außer dem definierten Muster
root@suse # at /etc/passwd | sed  '2!d'
avahi:x:494:492:User for Avahi:/var/run/avahi-daemon:/bin/false
```

Zeilen löschen mit sed

- alles außer den ersten 10 Zeilen löschen
  - sed '11,$d' /etc/passwd 11-Ende löschen
  - sed '1,10!d' /etc/passwd 1-10 NICHT löschen
- Zeile 5-15 aus dem Dokument löschen
  - sed '5,15d' eingabe.txt > ausgabe.txt
- alles ab der 2ten Zeile aus dem Dokument löschen
  - sed '2,$d' eingabe.txt > ausgabe.txt

Zeilen löschen mit sed

- sed '1,/www/d' /etc/passwd
  - In Zeile 1 beginnen und alle Zeilen markieren bis in einer Zeile der Suchbegriff 'www' gefunden wird
    - 1 , /suchbegriff/
  - Die Aktion für die Auswahl ist "delete"

Erklärung: Zeile 1-~25 wird gelöscht,
Zeile ~26-Ende wird ausgegeben

```
# In der Zeile 25 steht der String www
root@tux # cat -n /etc/passwd | sed '1,/www/d'
    26    user:x:1000:100:user:/home/user:/bin/bash
    27    donald:x:1001:100::/home/donald:/bin/bash
```

Delete & reguläre Ausdrücke

- sed '/:1[0-9][0-9][0-9]:/!d ' /etc/passwd
  - ID 1000-1999 nicht löschen
    - Alle Benutzer mit einer ID 1000-1999 anzeigen

```
# Suche nach einer speziellen UID
root@tux # sed  '/:1[0-9][0-9][0-9]:/!d' /etc/passwd
user:x:1000:100:user:/home/user:/bin/bash
donald:x:1001:100::/home/donald:/bin/bash
```

  - Varianten
    - sed -n '/:1[0-9]\{3\}:/p' /etc/passwd
    - grep ':1[0-9]\{3\}:' /etc/passwd
    - egrep ':1[0-9]{3}:' /etc/passwd

Startzeile~Aktion

```
# In der 1 Zeile beginnen und jede 2te Zeile löschen
peter@tux~>  cat passwd | sed  '1~2d'
2  avahi:x:494:492:User for Avahi:/var/run/avahi-daemon:/bin/false
4  daemon:x:2:2:Daemon:/sbin:/bin/bash
6  ftp:x:40:49:FTP account:/srv/ftp:/bin/bash
8  lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
10  man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash
```

```
# In der 2 Zeile beginnen und jede 2te Zeile löschen
peter@tux~> cat passwd | sed  '2~2d'
1  at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
3  bin:x:1:1:bin:/bin:/bin/bash
5  dnsmasq:x:495:65534:dnsmasq:/var/lib/empty:/bin/false
7  games:x:12:100:Games account:/var/games:/bin/bash
9  mail:x:8:12:Mailer daemon:/var/spool/clientmqueue:/bin/false
```

Speichern unter

- Schreiben
  - Mit 'w' kann ein markierter Bereich in eine neue Datei geschrieben werden

```
# Ausgabe unterdrücken und Zeile 2-4 in eine neue Datei schreiben
peter@tux~>  sed -n '2,4w newfile.txt'   /etc/passwd
peter@tux~>  cat newfile.txt
2  avahi:x:494:492:User for Avahi:/var/run/avahi-daemon:/bin/false
3  bin:x:1:1:bin:/bin:/bin/bash
4  daemon:x:2:2:Daemon:/sbin:/bin/bash
```

suchen & ersetzen

```
sed  's /SuchMuster/ErsatzMuster/'  [Datei(en)]
```

- sed 's/suse/linux/' option
  - s ...Suchen und Ersetzen eines Musters
  - suse ...Suche nach Muster (Case Sensitiv)
  - linux ...Ersetzen durch Muster
  - / ...beliebiges Trennzeichen (3x)

Suchen und Ersetzen

- Doppelpunkte durch Tabulatoren ersetzen
  - sed 's/:/\t /g' passwd
  - sed 's/:/STRG-V TAB/g' passwd

```
# Normale Ausgabe
peter@tux~> cat /etc/passwd
at:x:25:25:Batch jobs d.:/var/spool/atjobs:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
```

```
# Suchen & Ersetzen
peter@tux~> sed -e 's/:/      /g' /etc/passwd
at      x       25      25      Batch jobs d.     /var/spool/atjobs      /bin/bash
bin     x       1       1       bin               /bin                   /bin/bash
ftp     x       40      49      FTP account       /srv/ftp               /bin/bash
```

SED

```
sed [option] 'xxx' eingabe
```

- SED Optionen
  - Bei komplexeren Aufgaben können zusätzliche sed-Optionen hilfreich sein

Option	Bedeutung
-n	Ausgabe unterdrücken
-e	Definition einer weiteren sed-Aktion
-f	Sed Aktionen aus einer Datei lesen
-i	Originaldatei bearbeiten

Ausgabe von Bereichen mit Print

- sed '<bereich>p'
  - Adressierten Bereich ausgeben (print)

- sed -n '<bereich>p'
  - Ausgabe unterdrücken (-n) und nur den mit (p) markierten Bereich anzeigen
    - sed -n '2p' ...2te Zeile
    - sed -n '$p' ...letzte Zeile
    - sed -n '3,6p' ...3-6te Zeile
    - sed -n '5,$p' ...5-letzte Zeile
    - sed -n 'p' ...alles ausgeben

2d vs. 2p vs. -n 2p

```
# Alles ausgeben und die 2te Zeile löschen
peter@tux~> cat -n /etc/passwd | sed '2d'
1  at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
3  bin:x:1:1:bin:/bin:/bin/bash
4  daemon:x:2:2:Daemon:/sbin:/bin/bash
```

```
# Alles ausgeben und die 2te Zeile anzeigen
peter@tux~> cat -n /etc/passwd | sed '2p'
1  at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
2  avahi:x:494:492:User for Avahi:/var/run/avahi-daemon:/bin/false
2  avahi:x:494:492:User for Avahi:/var/run/avahi-daemon:/bin/false
3  bin:x:1:1:bin:/bin:/bin/bash
```

```
# Nichts ausgeben und nur die 2te Zeile anzeigen
peter@tux~> cat -n /etc/passwd | sed -n '2p'
2  avahi:x:494:492:User for Avahi:/var/run/avahi-daemon:/bin/false
```

mehrere Aktionen durchführen

- sed -e 's/:/\t /g' -e '5,10d' /etc/passwd
  - Wenn mehrere Aktionen durchgeführt werden sollen muss jede Aktion mit einem -e eingeleitet werden

- sed -e '{s/:/\t /g ; 5,10d}' /etc/passwd
  - Optional können die einzelnen Aktionen in einer geschwungen Klammer mit Strichpunkt getrennt werden

SED

sed [-n] [-e script] [-f scriptdatei] [eingabe]

- sed -n
  - Alle Ausgaben unterdrücken, außer sie werden mit p explizit angefordert
- sed -e
  - Wenn sed mehrere Befehle ausführen soll muss vor jedem Befehl -e angeführt werden
- sed -f
  - sed liest seine Editing-Befehle aus einer Datei

sed

```
text=abcdefghijk
```

- echo $text | sed 's/cde/-----/g'
  - löscht die Zeichenfolge cde

www.LinuxCampus.net

anhängen

- cat /etc/passwd | sed '/false/a\
  ----------'
  - Nach jeder Zeile die das Wort false enthält den Text aus der 2 Zeile anhängen
  - a\ ...nach der Zeile anhängen
  - i\ ...vor der Zeile anhängen

```
news:x:9:13:News system:/etc/news:/bin/bash
polkituser:x:104:106:PolicyKit:/var/run/PolicyKit:/bin/false
----------
postfix:x:51:51:Postfix Daemon:/var/spool/postfix:/bin/false
----------
root:x:0:0:root:/root:/bin/bash
```

Syntax

- Wichtiges zum Syntax
  - ältere sed-Versionen können nur die Variante a\ [ENTER]
    - der einzufügende Text musst in einer Fortsetzungszeile stehen

```
sed '/false/a\
----------------------'
```

  - neuere sed-Versionen unterstützen auch die Variante a\
    - der einzufügende Text kann in der selben Zeile stehen

```
sed '/false/a\  ----------------------'
```

anhängen & einfügen

- cat /etc/passwd | sed -e '/root/a\
  ----------' -e '/root/i\
  ----------'

```
news:x:9:13:News system:/etc/news:/bin/bash
polkituser:x:104:106:PolicyKit:/var/run/PolicyKit:/bin/false
postfix:x:51:51:Postfix Daemon:/var/spool/postfix:/bin/false
----------
root:x:0:0:root:/root:/bin/bash
----------
squid:x:31:65534:WWW-proxy squid:/var/cache/squid:/bin/false
sshd:x:71:65:SSH daemon:/var/lib/sshd:/bin/false
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
```

anhängen & einfügen

- sed ' /Sysadmin/ a\ ...'
  - Nach der Zeile mit dem Suchmuster etwas anhängen
- sed '3 a\ ... '
  - Nach der 3ten Zeile etwas anhängen
- sed '$ a\ ...'
  - Nach der letzten Zeile etwas anhängen

Change

- sed '3 c\ ----------'
  - Mit c\ für Change können wir auch eine Zeile durch einen neue ersetzten
    - Befehl funktioniert gleich wie a\ und i\

Originaldatei bearbeiten

- Originaldatei bearbeiten
  - Die Option -i führt dazu, daß die Änderungen nicht am Bildschirm angezeigt werden sondern direkt in der Originaldatei
    - ACHTUNG kann zu Datenverlust führen

```
# Zu Übungszwecken eine Sicherungskopie erstellen
peter@tux~>  cp  /etc/passwd  passwd
```

```
# Zeile 2-3 in der Sicherungskopie  löschen
peter@tux~>  sed -i  '2,3d'  passwd
```

Hilfreiche Links

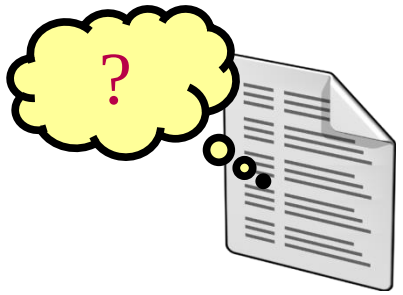
- sed and Multi-Line Search and Replace
  - http://austinmatzko.com/2008/04/26/sed-multi-line-search-and-replace/
- Sed: Texte automatisch verarbeiten
  - http://www.linux-fuer-alle.de/doc_show.php?docid=181&catid=8

Zählen

Inhalte zählen

- Wieviele Zeilen, Wörter, Zeichen hat eine Datei
  - wc Preisliste.txt
    - -l oder --lines ...zählt nur die Zeilen
    - -w oder --words ...zählt nur die Wörter
    - -c oder --chars ...zählt nur die Zeichen

```
# Zählen & ausgeben der  Zeilen, Wörter und  Zeichen
peter@tux~>  wc Preisliste.txt
 180  409  5784   Preisliste.txt
```



wc - Beispiele

- Zählen der *.conf Dateien
  - ls /etc/*.conf | wc -l
- Begriff "wait" in Datei inittab zählen
  - grep wait /etc/inittab | wc -l
- Liste der Dateien speichern + Summe ausgeben
  - ls -l | sed 1d | wc -l

    sed 1d ...löscht die 1 Zeile (Summe) von ls -l

wc Beispiele

- Alle Dateien in diesem Verzeichnis zählen
  - Wenn der find Befehl ohne Suchbegriff aufgerufen wird zeigt er alle Objekte an die sich im Suchpfad befinden. Ideal um die Objektanzahl zu zählen

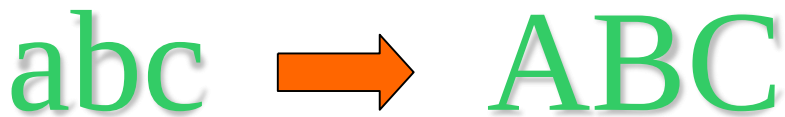
```
# Objekte zählen
peter@tux>  find . | wc -l
4182
```

```
# Objekte zählen
peter@tux>  find /etc | wc -l
3280
```

Dateiinhalte übersetzen

Übersetzen von Dateiinhalten

- Was ist tr?
  - tr ist ein "translation" Filter
  - tr sucht nach Zeichenmengen und ersetzt sie
- Anwendungsgebiete
  - Kleinbuchstaben durch Großbuchstaben ersetzen
  - Leerzeilen entfernen
  - Wörter ersetzen



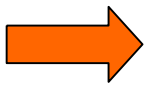
Translation Filter

```
tr -ds Menge1 Menge2
```

TR OPTIONEN	
-s	Reduziert mehrere Zeichen aus Menge 1 auf ein solches Zeichen (squeeze)
-d	Enfernt alle Zeichen aus Menge1 (delete)
-ds	delete Menge1 DANACH reduziert Menge2
-c	Ersetzt Menge1 durch alle Zeichen die nicht in der Menge sind

Übersetzen von Dateiinhalten

- alle vorkommenden a,b durch A,B ersetzen
  - cat Test.txt | tr ab AB > TestNeu.txt
- Klein durch Grossbuchstaben ersetzen
  - cat klein.txt | tr a-z A-Z > gross.txt

abc  ABC

Übersetzen von Dateiinhalten

- ersetzen mit vordefinierte Mengen
  - cat klein.txt | tr [:lower:] [:upper:] > gross.txt

VORDEFINIERTE MENGEN	
[:lower:]	Kleinbuchstaben
[:upper:]	Grossbuchstaben
[:alnum:]	a-zA-Z0-9
[:space:]	Leerzeichen (horizontal & vertikal)
weitere Optionen: man tr oder man grep	

tr - Beispiele

- tr abc xyz
  - Ersetzen von a durch x, b durch y, c durch z
- tr -s '\n'
  - alle Leerzeilen löschen (Newline-Zeichen)
- tr -cs 'a-zA-Z0-9' '\n'
  - Jedes Wort in eine einzelne Zeile umbrechen

tr & löschen

- Definieren einer Variable
  - text=abcdefghijk
- Löschen der einzelnen Zeichen a c e
  - echo $text | tr -d 'ace'

tr & löschen

- Definieren einer Variable
  - VAR=%triples
- Zeichen % löschen und wieder speichern
  - VAR=`echo $VAR | tr -d %`

Dateien umbennen (A->a)

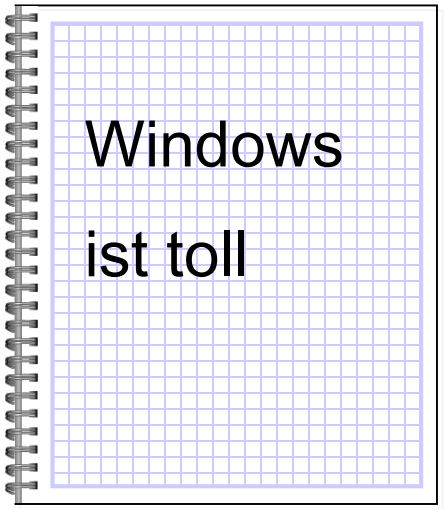
- Dateien umbenennen
  - alle Dateien im lokalen Verzeichnis bearbeiten
  - alle Großbuchstaben durch Kleinbuchstaben ersetzen

```
for i in *
 do
    lower=`echo $i | tr  [:upper:] [:lower:]`
    mv $i $lower
 done
```

Dateiinhalte vergleichen

Vergleichen von Dateiinhalten

```
diff /etc/hosts /etc/hosts.old
```



DIFF

```
diff -i /etc/hosts /etc/hosts.old
```

OPTIONEN	
-w	Leerzeichen ignorieren
-c , -c #	zusätzlich 3 (#) Folgezeilen ausgeben
-r	Rekursiv alle Unterverzeichnisse vergleichen
-i	Gross- & Kleinschreibung ignorieren
man diff	

Dateien vergleichen mit "diff"

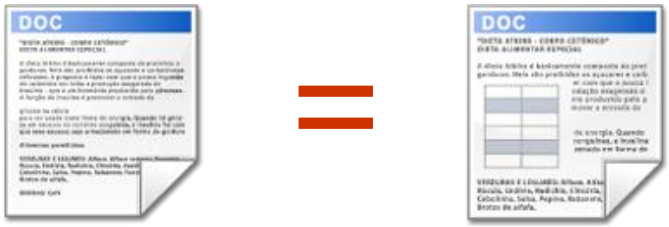
```
#etc/hosts
127.0.0.1       localhost
10.5.1.1        pc1
10.5.1.2        pc2
10.5.1.3        pc3
```

```
#etc/hosts.backup
127.0.0.1       localhost
10.5.1.1        pc1
10.5.1.2        pc2
10.5.1.4        pc4
```

```
1c1
< #etc/hosts
> #etc/hosts.backup
5c5
< 10.5.1.3      pc3
>10.5.1.4       pc4
```

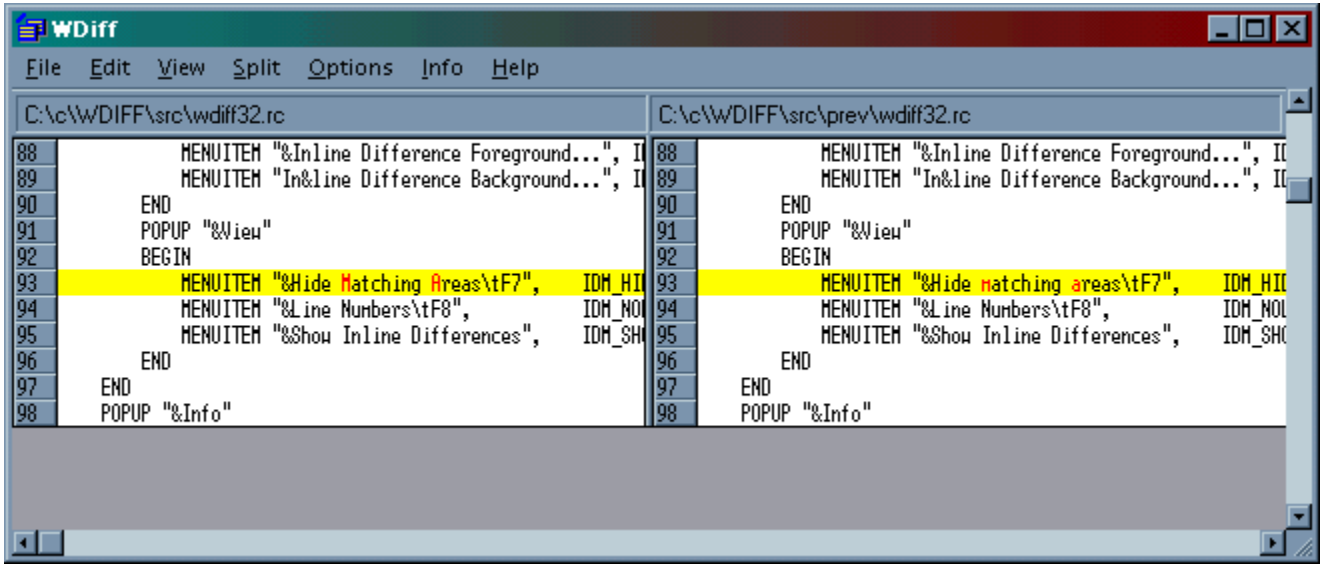
Dateien vergleichen

- Vergleichen von 2 Dateiinhalten
  - diff optionen datei1 datei2
- Vergleichen von 2 grossen Dateiinhalten
  - diff optionen datei1 datei2 > differenz.txt &



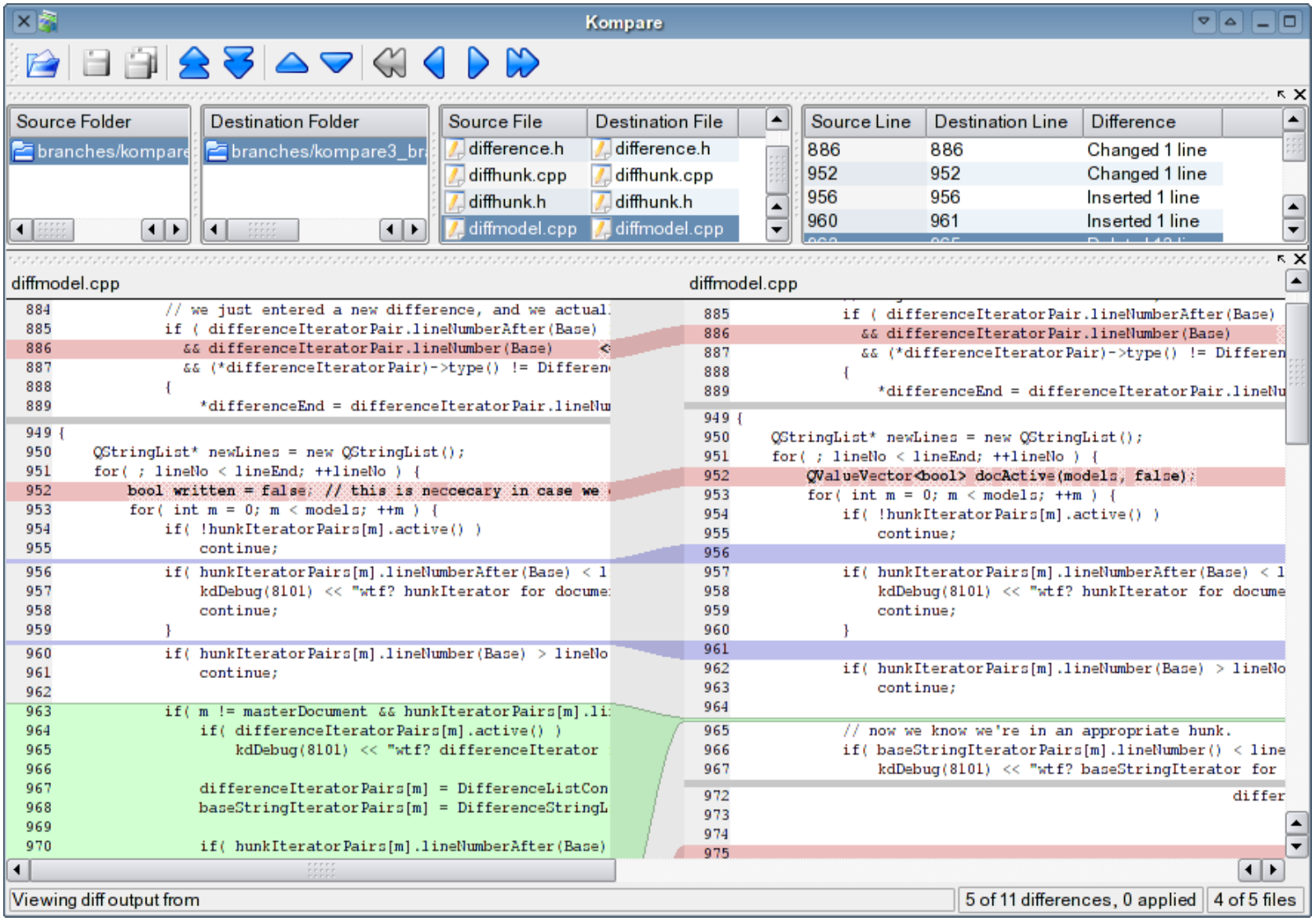
Varianten von diff

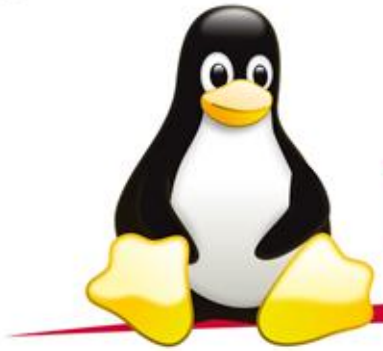
- wdiff
  - ist eine Frontend für diff und vergleicht wort-weise
  - ideal für Texte wo sich eher Wörter anstatt von Zeilen verändert haben



www.LinuxCampus.net

Grafisches KDE-DIFF: kompare





LinuxCampus.net

trainings from the experts