ERGAENZUNGEN ZUM KOMMANDOMANUAL FUER SINIX 5.24 Stand: August 1991

In diesem README-File sind eine Reihe von Korrekturen und Ergaenzungen gegenueber dem SINIX Kommando-Manual zur Version 5.22 zusammengefasst. Diese Datei enthaelt noch einmal den vollstaendigen Manualtext zu den bearbeiteten Kommandos. Die veraenderten Passagen sind durch ein @-Zeichen in der ersten Spalte markiert. Den Kommandobeschreibungen ist noch ein Kapitel ueber die Signalbehandlung durch die Shell vorangestellt.

Die Korrekturen und Ergaenzungen in der Uebersicht:

Kommando:	Korrektur:	auf Zeile:
ed	Schreibfehler auf Manualseite 6-397	344
env	Kein Shell-Builtin in der Kommandozeile	1341
ex	Unterschied zwischen 'list' und 'print'	2333
		2455
ipcrm	KEY=00000 ist nicht zulaessig	3217
ps	Ausgabewerte koennen miteinander verschmelzen	3373
tar	Keine 'verbose'-Option bei Ausgabe auf stdout	3709
tr	Probleme bei Zeichenbereichsangaben	4260
		4269

****

ED ZEILENORIENTIERTER EDITOR IM DIALOGBETRIEB

'ed' ist ein interaktiver zeilenorientierter Editor. Mit Hilfe von 'ed-Skripts' (siehe 'Arbeiten mit ed-Skripts') koennen Sie bequem mehrere Dateien mit derselben Kommandofolge bearbeiten. 'ed' verarbeitet die Ausgabe des Kommandos 'diff -e' (siehe 'diff').

```
ed[ -][ -p zeichenkette][ datei]
```

_

Die Option - unterdrueckt die folgenden standardmaessigen Ausgaben:

- Anzahl der verarbeiteten Zeichen bei den 'ed'-Kommandos
  'e' (edit - editieren)
  'r' (read - lesen)
  'w' (write - schreiben)

- Fragezeichen ?, das vor versehentlichem Loeschen des Pufferinhalts warnt, bei den 'ed'-Kommandos
  'e' (edit - editieren)
  'q' (quit - verlassen)

- Ausrufezeichen ! als Bereitzeichen von 'ed' nach einem !-Kommando.

-p zeichenkette
Mit 'zeichenkette' koennen Sie die Bereitzeichenkette definieren, die 'ed' im Kommandomodus ausgibt. Fuer 'zeichenkette' koennen Sie ein oder mehrere Zeichen angeben. Shell-Sonderzeichen (siehe A.2 'Sonderzeichen der Shell')
in 'zeichenkette'
muessen Sie entwerten: entweder einzeln, indem Sie vor jedes Sonderzeichen einen Gegenschraegstrich \ setzen oder mehrere zusammen, indem Sie sie in Hochkommata '...' einschliessen.

'-p zeichenkette' nicht angegeben:
'ed' gibt keine Bereitzeichenkette aus.

datei
Name der Datei, die Sie bearbeiten moechten. 'ed' kopiert die Datei in den internen Puffer und speichert 'datei' als aktuellen Dateinamen.

'datei' nicht angegeben:
Sie beginnen in einem leeren Puffer zu arbeiten und bestimmen erst beim Wegschreiben des Pufferinhalts mit dem Kommando 'w' 'datei' (write - schreiben) in eine Datei deren Namen.

ED-PUFFER

Beim Aufruf von 'ed' wird ein Puffer eroeffnet.

Wenn Sie keine Datei angegeben haben, ist der Puffer leer. Sie fuellen ihn waehrend Ihrer Editorsitzung mit Text.

Wenn Sie eine Datei angegeben haben, wird eine Kopie dieser Datei in den Puffer eingelesen. Den Text im Puffer bearbeiten Sie waehrend Ihrer Editorsitzung.

Bevor Sie den Editor wieder verlassen, muessen Sie entscheiden, ob Sie den neu erstellten oder geaenderten Pufferinhalt sichern, d.h. in eine Datei schreiben moechten.

Wenn ja, schreiben Sie den Pufferinhalt mit dem Kommando 'w'['datei'] (write - schreiben) in die angegebene Datei (Standard: die beim Aufruf von 'ed' angegebene) zurueck und verlassen dann den Editor mit einem der Kommandos 'q' (quit - verlassen), 'Q' (Quit - verlassen) oder mit der Taste <END>.

Wenn nein, koennen Sie mit 'Q' oder zweimal 'q' den Editor verlassen,

ohne vorher den Pufferinhalt mit 'w' zurueckzuschreiben. Nach der ersten Eingabe von 'q' gibt 'ed' als Warnung vor versehentlichem Loeschen des Pufferinhalts ein ? aus. Geloescht wird der Pufferinhalt erst, wenn Sie dann ein zweites Mal 'q' eingeben. Statt des zweiten 'q' koennen Sie auch 'Q' oder <END> eingeben.

ARBEITSMODI

Der 'ed' bietet Ihnen zwei Arbeitsmodi: den Kommandomodus und den Eingabemodus. Nach dem Aufruf mit 'ed[datei]' <ENTER> befindet sich 'ed' im Kommandomodus. Im Kommandomodus geben Sie i.a. ein Kommando in einer Zeile an und schliessen es mit der Taste <ENTER> ab.

Den Eingabemodus schalten Sie ein mit einem der Kommandos
'a' (append - anfuegen)
'i' (insert - einfuegen)
'c' (change - veraendern)
(siehe unten, 'ed-Kommandos').

Im Eingabemodus werden alle folgenden Eingabezeichen, auch nicht-druckbare Zeichen (wie z.B. die Tastencodes der Schreibmarkentasten), in die Arbeitskopie im Puffer geschrieben. Im Eingabemodus erkennt 'ed' keine Kommandos. Sie koennen fuer den Kommandomodus ein Bereitzeichen definieren (siehe Option '-p' und 'ed'-Kommando 'P') und erkennen dann sofort, in welchem Modus Sie gerade arbeiten. Verlassen koennen Sie den Eingabemodus mit einem Punkt in der ersten Spalte . <ENTER> oder mit der Taste <DEL>. Die Taste <DEL> bewirkt generell, dass 'ed' alle Eingaben seit dem letzten <ENTER> ignoriert und ein ? als Warnung ausgibt.

KOMMANDOSTRUKTUR

Fuer 'ed' existiert zu jedem Zeitpunkt eine 'aktuelle Zeile'. Die aktuelle Zeile ist in der Regel die zuletzt durch ein Kommando bearbeitete Zeile. Auf diese aktuelle Zeile beziehen sich die Kommandos immer dann, wenn Sie vor den Kommandos keine anderen Adressen angeben.

Die meisten 'ed'-Kommandos haben folgende Struktur:

```
-------------------------------------------------------------------
[bereich]ed-kommando[parameter...] <ENTER>
-------------------------------------------------------------------
```

bereich
   Mit 'bereich' waehlen Sie die Zeilen im
   Puffer aus, auf die das 'ed'-Kommando angewendet werden soll.
   Fuer 'bereich' koennen Sie eine oder zwei Adressen angeben:

   'bereich' = adresse
      Die durch 'adresse' bezeichnete Zeile gilt als ausgewaehlt.

'bereich' = adresse1,adresse2
'adresse1','adresse2' kennzeichnen den Bereich zwischen den angegebenen Intervallgrenzen einschliesslich.
Die Suche nach beiden Adressen beginnt in der aktuellen Zeile.
Veraendert wird die aktuelle Zeile erst bei der Ausfuehrung von Kommandos.
'adresse2' muss sich auf eine Zeile im Puffer beziehen, die hinter der mit 'adresse1' bezeichneten Zeile liegt, sonst meldet 'ed' einen Fehler.

'bereich' = adresse1;adresse2
'adresse1';'adresse2' kennzeichnen den Bereich zwischen den angegebenen Intervallgrenzen einschliesslich.
Die Suche nach 'adresse1' beginnt in der aktuellen Zeile.
Die mit 'adresse1' bezeichnete Zeile wird dann zur neuen aktuellen Zeile, dann erst wird 'adresse2' ermittelt.
Sie koennen so die Zeile festlegen, bei der ein Suchvorgang in Vor- und Rueckwaertsrichtung beginnen soll (siehe 'Adressen, Punkte 5 und 6').
'adresse2' muss sich auf eine Zeile im Puffer beziehen, die hinter der mit 'adresse1' bezeichneten Zeile liegt, sonst meldet 'ed' einen Fehler.

'bereich' nicht angegeben:
'ed' nimmt die zum jeweiligen Kommando gehoerige Standard-Adresse an; diese ist bei jedem 'ed'-Kommando beschrieben.

Benoetigt 'ed' keine Adresse und haben Sie trotzdem eine angegeben, meldet 'ed' einen Fehler.

Haben Sie mehr Adressen angegeben als das Kommando erlaubt, nimmt 'ed' die letzten.

ADRESSEN

Adressen koennen Sie wie folgt konstruieren:

	Adresse	Bedeutung
1.	.	aktuelle Zeile
2.	$	letzte Zeile
3.	n	n-te Zeile
4.	'x	die mit dem Buchstaben 'x' markierte Zeile. 'x' muss ein Kleinbuchstabe sein (siehe Kommando 'k').
5.	/rA/	in einfacher internationalisierter regulaerer Ausdruck 'rA' in /.../ eingeschlossen (siehe A.1 'Regulaere Ausdruecke'),

adressiert die erste Zeile, die
eine zu dem regulaeren Ausdruck passende
Zeichenkette enthaelt.
Es wird dabei von der aktuellen Zeile aus vorwaerts
gesucht. Findet 'ed' keine passende Zeile, setzt
'ed' die Suche am Dateianfang fort und sucht
wieder bis zur aktuellen Zeile.
Enthaelt der regulaere Ausdruck 'rA' selbst
eines der Begrenzungszeichen / oder ?, muss es
mit \ entwertet werden.

Die Zeichen \n in einem regulaeren Ausdruck
erzielen bei Neue-Zeile-Zeichen im durchsuchten
Text 'keinen' Treffer!
Ein regulaerer Ausdruck kann in 'bereich' nur
einmal vorkommen, z.B. adressiert /rA1/,/rA2/
nur die erste zu /rA2/ passende Zeile.

6. // Ein leerer regulaerer Ausdruck // adressiert -
die Zeile, die zu dem zuletzt angegebenen regulaeren
Ausdruck passt.

7. ?rA?
ie /rA/, aber es wird von der aktuellen Zeile -
aus rueckwaerts in Richtung Dateianfang gesucht.
Enthaelt der regulaere Ausdruck 'rA' selbst
eines der Begrenzungszeichen / oder ?, muss es
mit \ entwertet werden.

8. adr[+]n
n' Zeilen nach der durch 'adr' bezeichneten Zeile.
adr-n
n' Zeilen vor der durch 'adr' bezeichneten Zeile.

9. +n 'n' Zeilen nach der aktuellen Zeile.
-n 'n' Zeilen vor der aktuellen Zeile.

10. [adr]+...
[adr]-...
Eine Zeile nach (+) bzw. vor (_) der durch 'adr'
bezeichneten Zeile. Jedes Vorkommen von + erhoeht,
von - erniedrigt die Adressangabe um 1.
Die Adressangabe ++ adressiert
also die zweite Zeile nach der aktuellen Zeile.

11. , Ein Komma , ist gleichbedeutend mit -
dem Adressenpaar 1,$, wenn ein Kommando folgt, sonst
wird die letzte Zeile ausgegeben.

; Ein Strichpunkt ; ist gleichbedeutend -
mit dem Adressenpaar .,$, wenn ein Kommando folgt, sonst
@ wird die letzte Zeile ausgegeben.

Die folgende Liste enthaelt eine systematische Uebersicht aller 'ed'-Kommandos, die Sie im Kommandomodus eingeben koennen. Die daran anschliessende ausfuehrliche Kommandobeschreibung ist alphabetisch geordnet.

Uebersicht der ed-Kommandos

Eingabemodus einschalten

a	append	anfuegen
c	change	loeschen und ersetzen
i	insert	einfuegen

Bereitzeichen im Kommandomodus ausgeben

P	prompt	Bereitzeichen * ausgeben

Kommandos rueckgaengig machen

u	undo	rueckgaengig machen

Kommandos abbrechen

<DEL>	---	abbrechen

Fehler erlaeutern

h	help	letzte Fehlermeldung erlaeutern Modus
H	Help	ein-/ausschalten, in dem Fehlermeldungen ausfuehrlich erlaeutert werden (siehe Fehlermeldungen).

Text aendern

a	append	anfuegen
c	change	loeschen und ersetzen
d	delete	loeschen
i	insert	einfuegen

Zeilen ausgeben

p	print	ausgeben
l	list	ausgeben mit nicht-druckbaren Zeichen als Oktalzahlen
adresse	---	adressierte Zeile ausgeben
adresse +zahl	---	adressierte Zeile ausgeben
n	number	Zeilen numerieren und ausgeben
<DEL>	---	Zeile hinter aktueller Zeile ausgeben

Zeilennummern ausgeben

adresse=	---	Nummer der adressierten Zeile ausgeben
n	number	Zeilen numerieren und ausgeben

Zeilenbereiche verschieben

```
   t        ---        verschieben von kopierten Zeilen
   m        move       verschieben von Zeilen



Textmuster suchen und ersetzen

   s        substitute suchen und ersetzen

Zeilen verbinden

   j        join       aufeinanderfolgende Zeilen verbinden

Zeilen markieren

   k        mark       Zeilen markieren

Ausgewaehlte Zeilen mit Kommandos bearbeiten

   g        global     Kommandoliste global fuer alle Zeilen
                  ausfuehren, die zu /rA/ passen
   G        Global     interaktiv Kommandoliste global fuer alle
                  Zeilen ausfuehren, die zu /rA/ passen
   v        ---        wie g, aber fuer alle Zeilen, die
                  nicht zu /rA/ passen
   V        ---        wie G, aber fuer alle Zeilen, die
                  nicht zu /rA/ passen

Aktuellen Dateinamen aendern

   f        file-name  aktuellen Dateinamen aendern/ausgeben

Shell-Kommandos ausfuehren

   !        ---        Shell-Kommando aufrufen

Dateien in Puffer einlesen

   e        edit       Pufferinhalt loeschen und neu einlesen
   E        Edit       Pufferinhalt ohne Warnung loeschen und
                  neu einlesen
   r        read       Datei in Puffer einlesen

Pufferinhalt sichern

   w        write      Pufferinhalt in Datei schreiben

Editor verlassen

   q        quit       'ed' verlassen
   Q        Quit       'ed' verlassen ohne Warnung
 <END>      ---        ed verlassen
-----------------------------------------------------------------


BESCHREIBUNG DER ED-KOMMANDOS
```

Die eckigen Klammern [] sind nicht einzugeben! Sie zeigen an, dass die dazwischen eingeschlossene Adressangabe fakultativ ist.
In der Regel darf in einer Zeile nur ein Kommando stehen. Jedoch koennen Sie an die Kommandos (mit Ausnahme von 'e, f, r' und 'w') als Suffix 'l, n' oder 'p' anhaengen, wenn die im folgenden unter den Kommandos 'l, n' und 'p' beschriebenen Funktionen ausgefuehrt werden sollen.

[adresse]a
text
.
   (a - append) liest
   den eingegebenen 'text' und fuegt ihn
   hinter der mit 'adresse' adressierten Zeile an.
   Die aktuelle Zeile ist nun entweder die letzte Zeile des
   eingefuegten Textes oder, falls Sie keinen Text eingegeben haben,
   die adressierte Zeile.
   Die Adresse 0 ist bei
   diesem Kommando erlaubt; der Text wird dann vor die erste Zeile
   des Puffers eingefuegt.
   Ueber die Datensichtstation koennen Sie maximal 256 Zeichen
   je Zeile einschliesslich Neue-Zeile-Zeichen eingeben.

   'adresse' nicht angegeben:
   'adresse' = .

[bereich]c
text
.
   (c - change) loescht den
   angegebenen 'bereich' und ersetzt
   ihn durch den eingegebenen 'text'.
   Die aktuelle Zeile ist nun entweder die letzte Zeile des
   eingegebenen Textes oder, falls Sie keinen Text eingegeben haben,
   die Zeile vor den geloeschten Zeilen.

   'bereich' nicht angegeben:
   'bereich' = .,.

[bereich]d
   (d - delete) loescht den angegebenen
   'bereich'. Die Zeile hinter der letzten geloeschten
   Zeile wird zur aktuellen Zeile.
   Standen die geloeschten Zeilen am Ende des Puffers, wird die
   neue letzte Zeile die aktuelle Zeile.

   'bereich' nicht angegeben:
   'bereich' = .,.

e[datei]
   (e - edit) loescht den
   gesamten Puffer und liest eine Kopie
   des Inhalts von 'datei' ein. Ist der Pufferinhalt
   seit dem letzten Sichern oder Ueberschreiben veraendert und nicht
   mit 'w' gesichert worden, loescht 'ed' den Pufferinhalt nicht

sofort, sondern warnt Sie mit einem ? vor versehentlichem Loeschen.
Geben Sie daraufhin ein weiteres Mal
'e' ein, wird der Pufferinhalt ohne weitere Warnung ueberschrieben.
Die Anzahl der eingelesenen Bytes wird ausgegeben, wenn Sie
'ed' nicht mit der Option - aufgerufen haben. Die aktuelle
Zeile ist die letzte Zeile des Puffers.
Der Name 'datei' wird gespeichert, so dass er spaeter bei den
Kommandos 'e, r' und 'w' als aktueller Dateiname verwendet
werden kann.
Wenn Sie fuer 'datei' ein Ausrufezeichen ! angeben, wird
der Rest der Zeile als Shell-Kommando interpretiert und
ausgefuehrt. Ein mit ! eingeleitetes Shell-Kommando wird nicht
Dateiname gespeichert.

'datei' nicht angegeben:
'datei' = aktueller Dateiname

E[datei]
(E - edit) verhaelt sich wie 'edit',
ausser, dass es den Puffer ohne warnendes ? ueberschreibt, auch
wenn der Inhalt des Puffers veraendert und nicht gerettet wurde.

'datei' nicht angegeben:
'datei' = aktueller Dateiname

f[datei]
(f - file) setzt den aktuellen Dateinamen auf 'datei'.
Den aktuellen Dateinamen verwenden die Kommandos 'e, E, r' und 'w'.

'datei' nicht angegeben:
'ed' gibt den aktuellen Dateinamen aus.

[bereich] g/rA/kommandoliste
(g - global) markiert im ersten Schritt
alle Zeilen, die eine Zeichenkette enthalten, die zu
'rA' passt. 'rA' ist ein einfacher internationalisierter
regulaerer Ausdruck (siehe A.1 'Regulaere Ausdruecke').
Dann wird die naechste markierte Zeile zur aktuellen Zeile
und 'kommandoliste' darauf angewandt.

Ein einzelnes Kommando oder das erste einer
'kommandoliste' muessen Sie in dieselbe Zeile schreiben wie
das Kommmando 'g'. Alle Zeilen einer 'kommandoliste'
ausser der letzten muessen Sie mit \ <ENTER>, die letzte
mit <ENTER> abschliessen.

Die Kommandos 'a', 'i' und 'c' mit der
dazugehoerigen Eingabe 'text' sind zugelassen.
Auch Zeilen in 'text' muessen Sie mit \ <ENTER> abschliessen.
Den die Eingabe normalerweise abschliessenden Punkt . koennen Sie
in der letzten Zeile der 'kommandoliste' weglassen.

Eine leere 'kommandoliste' entspricht dem Kommando 'p'.

Die Kommandos 'g,G,v' und 'V' sind
in der 'kommandoliste' nicht zugelassen.

Nicht mit dem Kommando ! zu kombinieren!

'bereich' nicht angegeben:
'bereich' = 1,$

[bereich]G/rA/
(G - global) 'G' ist die interaktive Variante des Kommandos 'g'.
Zuerst wird jede Zeile markiert, die zu
'rA' passt. 'rA' ist ein einfacher internationalisierter
regulaerer Ausdruck (siehe A.1 'Regulaere Ausdruecke').
Die erste der markierten Zeilen
wird ausgegeben. Sie wird gleichzeitig zur aktuellen Zeile.

Sie haben nun die Moeglichkeit, ein Kommando anzugeben, das
ausgefuehrt werden soll. Hierbei darf es sich nicht um
'a, c, i, g, G, v' oder 'V' handeln. Nach der Ausfuehrung dieses
Kommandos wird die naechste markierte Zeile ausgegeben, usw.

Ein Neue-Zeile-Zeichen hat dieselbe Wirkung wie ein leeres
Kommando. Wenn Sie das kommerzielle Und & eingeben, wird das
Kommando wiederholt, das seit dem letzten Aufrufen von 'G'
als letztes ausgefuehrt wurde.

Mit Kommandos, die unter 'G'
aufgerufen werden, koennen beliebige Zeilen im Puffer adressiert
und bearbeitet werden. 'G' koennen Sie mit der Taste <DEL>
abbrechen.

'bereich' nicht angegeben:
'bereich' = 1,$

h
(h - help) gibt eine kurze Fehlermeldung aus,
die den Grund fuer das letzte Fragezeichen ?
auf dem Bildschirm erklaert.
Moegliche Fehlermeldungen siehe 'Fehlermeldungen'.

H
(H - help) bewirkt, dass 'ed' in einen Modus
uebergeht, in dem bei auftretenden Fehlern anstelle von ?
erlaeuternde Fehlermeldungen ausgegeben werden. Auch das letzte
ausgegebene ? wird, falls vorhanden, erlaeutert. Diesen Modus
koennen Sie durch ein zweites Aufrufen von 'H' ausschalten.
Standardmaessig ist der Modus 'H' ausgeschaltet.
Moegliche Fehlermeldungen siehe 'Fehlermeldungen'.

[adresse]i
text
.
i - insert) fuegt 'text' vor der
mit 'adresse' adressierten Zeile ein.
Die aktuelle Zeile ist nun entweder die letzte Zeile des
eingefuegten Textes oder, falls Sie keinen Text eingegeben haben,
die adressierte Zeile.
Dieses Kommando unterscheidet
sich vom Kommando 'append' nur durch die Positionierung des

eingegebenen 'text'es. Adresse 0 ist nicht zulaessig. Ueber eine
Datensichtstation duerfen Sie maximal 256 Zeichen
einschliesslich des Neue-Zeile-Zeichens
je Zeile eingeben.

'adresse' nicht angegeben:
'adresse' = .

[bereich]j
(j - join) verbindet alle in 'bereich'
liegenden Zeilen zu einer einzigen Zeile und loescht dabei die
entsprechenden Neue-Zeile-Zeichen.
Wenn Sie als Adresse nur eine Zeile angeben, geschieht nichts.
Die neue Zeile wird zur aktuellen Zeile.

'bereich' nicht angegeben:
'bereich' = .,.

[adresse]kx
(k - mark) markiert die mit 'adresse'
adressierte Zeile mit dem
fuer 'x' angegebenen Buchstaben, wobei 'x' ein
Kleinbuchstabe sein muss.
Die Markierung wird aber nicht ausgegeben.
Adressieren koennen Sie die markierte Zeile mit Hochkomma x 'x.
Die aktuelle Zeile bleibt unveraendert.

'adresse' nicht angegeben:
'adresse' = .

[bereich]l
(l - list) gibt im Gegensatz zu
'p' den angegebenen 'bereich' wie folgt aus:
Einige nicht-druckbare Zeichen werden
durch Mnemoniks, z.B Tabulatorzeichen durch das Zeichen >,
die uebrigen nicht-druckbaren Zeichen werden als Oktalzahlen
ausgegeben; ueberlange Zeilen werden mehrzeilig ausgegeben, am Ende
mit dem Zeilenfortsetzungszeichen \ versehen.
Die NLS-Variable 'LC_CTYPE' legt fest, welche Zeichen
in der aktuellen internationalisierten Umgebung als
nicht-druckbar gelten.
Das Kommando 'l' koennen Sie an jedes Kommando anhaengen, mit
Ausnahme von 'e, f, r' und 'w'.
Beachten Sie, dass durch die Darstellung des Neue-Zeile-Zeichens
als \012 die optische Zeilenstruktur zerstoert wird.

'bereich' nicht angegeben:
'bereich' = .,.

[bereich]madresse
(m - move) verschiebt
den 'bereich' hinter
die mit 'adresse' adressierte Zeile. Die Letzte der verschobenen
Zeilen wird die aktuelle Zeile.
Wenn Sie fuer 'adresse' den Wert 0 angeben, wird 'bereich' ganz
an den Anfang der Datei gesetzt.

Liegt 'adresse' innerhalb von 'bereich', gibt 'ed' eine
Fehlermeldung aus.

'bereich' nicht angegeben:
'bereich' = .,.

[bereich]n
(n - number) gibt die mit 'bereich'
adressierten Zeilen aus, wobei an den Anfang jeder Zeile die
Zeilennummer und ein Tabulatorzeichen gesetzt werden. Die zuletzt
ausgegebene Zeile wird zur neuen Zeile.
Das Kommando 'n' koennen Sie an jedes Kommando anhaengen, mit
Ausnahme von 'e, f, r' und 'w'.

'bereich' nicht angegeben:
'bereich' = .,.

[bereich]p
(p - print) gibt die mit
'bereich' adressierten Zeilen aus.
Nicht-druckbare Zeichen werden unveraendert ausgegeben.
Ueberlange Zeilen laufen ohne besondere Kennzeichnung auf der
naechsten Zeile weiter, d.h. man kann sie nicht als solche
erkennen. Die zuletzt ausgegebene Zeile wird zur aktuellen Zeile.
Das Kommando 'p' koennen Sie an jedes Kommando anhaengen, mit
Ausnahme von 'e, f, r' und 'w'. Mit 'dp' wird also die aktuelle
Zeile geloescht und die neue aktuelle Zeile ausgegeben.

'bereich' nicht angegeben:
'bereich' = .,.

P
(P - prompt) bewirkt, dass 'ed' im Kommandomodus einen
Stern * als Bereitzeichen ausgibt. Diesen
Modus koennen Sie durch einen zweiten Aufruf von 'P' wieder
ausschalten. Standardmaessig ist der *-Modus ausgeschaltet.

q
(q - quit) beendet den 'ed'.
Ist der Pufferinhalt seit dem letzten Sichern oder Ueberschreiben
veraendert und nicht mit 'w' gesichert worden,
gibt 'ed' als Warnung vor versehentlichem Loeschen
ein ? aus und wartet auf weitere Eingabe.
Wenn Sie dann <END>, 'Q' oder zum zweitenmal 'q' eingeben,
beenden Sie den 'ed' ohne weitere Warnung und
ohne den Pufferinhalt gesichert zu haben.

Vorsicht

Wenn Sie nach dem ersten 'q' noch Aktionen durchfuehren,
die den Pufferinhalt nicht veraendern, beenden Sie mit dem
zweiten 'q' auch den 'ed' ohne weitere Warnung und
ohne Sicherung des Pufferinhalts.

Q
(Q - quit) beendet den 'ed' sofort

ohne Warnung,
auch wenn Sie den Pufferinhalt seit dem letzten Sichern oder
Ueberschreiben veraendert und nicht mit 'w' gesichert haben.

[adresse]r[datei]
(r - read) liest 'datei' und fuegt ihren
Inhalt hinter der mit 'adresse' adressierten Zeile ein.
Die Adresse 0 ist fuer dieses Kommando erlaubt.
Sie bewirkt, dass 'datei' an den Anfang des Puffers
geschrieben wird.
Nach erfolgreichem Lesen wird die Anzahl der gelesenen Bytes
ausgegeben, wenn Sie 'ed' nicht mit der Option - aufgerufen
haben. Die aktuelle Zeile ist die letzte eingelesene Zeile.
Der aktuelle Dateiname wird nicht auf 'datei' gesetzt,
es sei denn, Sie haben 'ed' ohne Dateinamen aufgerufen und
'datei' ist der erste seit dem Aufruf angesprochene Dateiname.
Wenn Sie fuer 'datei' ein Ausrufezeichen ! angeben, dann
wird der Rest der Zeile als Shell-Kommando interpretiert,
ausgefuehrt und die Ausgabe davon eingelesen. Ein solches
Kommando wird nicht als aktueller Dateiname gespeichert.

'adresse' nicht angegeben:
'adresse' = $

'datei' nicht angegeben:
'datei' = aktueller Dateiname

[bereich]s/rA/Ersetzungszeichenkette/[g][n]
(s - substitute) durchsucht jede Zeile
in 'bereich' nach Zeichenketten, die zu
'rA' passen. 'rA' ist ein einfacher internationalisierter
regulaerer Ausdruck (siehe A.1 'Regulaere Ausdruecke').
In 'jeder' so gefundenen Zeile wird eine zu 'rA'
passende Zeichenkette durch 'Ersetzungszeichenkette' ersetzt:
ohne 'g' und 'n' wird das erste, mit 'g' jedes und
mit 'n' das n-te Auftreten einer passenden Zeichenkette in der
Zeile ersetzt. 'n' ist eine positive ganze Zahl von 1 bis 512.
Falls keine passende Zeichenkette gefunden wurde,
meldet 'ed' ein ? als Fehler.
Als Trennzeichen zwischen dem Kommando 's', dem regulaeren
Ausdruck 'rA' und 'Ersetzungszeichenkette'
koennen Sie nicht nur den Schraegstrich /
verwenden, sondern auch jedes andere beliebige Zeichen
ausser: Leerzeichen und Neue-Zeile-Zeichen.
Das Zeichen wird dadurch als Trennzeichen definiert, dass es
unmittelbar auf 's' folgt.
Zur aktuellen Zeile wird die Zeile, auf der die letzte Ersetzung
stattgefunden hat.

Sonderzeichen in der Ersetzungszeichenkette

Sonderzeichen in Ersetzungszeichenkette | Bedeutung>

& | wird durch die Zeichenkette ersetzt, die in der aktuellen Zeile zu dem

regulaeren Ausdruck rA passt.

\m wird durch die Zeichenkette - ersetzt, die zum m-ten in \(...\) eingeschlossenen regulaeren Unterausdruck von rA passt, wobei m eine einstellige Dezimalzahl ist. Bei durch Klammern ineinander geschachtelten Unterausdruecken wird m durch Zaehlen des Auftretens von \ von links nach rechts bestimmt.

% wird durch die Zeichenkette ersetzt, - mit der beim zuletzt abgelaufenen s-Kommando ersetzt wu'rd'e, wenn Ersetzungszeichenkett'e nur aus dem Prozent-Zeichen % besteht.

Die Sonderbedeutung dieser Zeichen koennen Sie aufheben, indem Sie Ihnen jeweils einen Gegenschraegstrich \ voranstellen.

Zeile trennen in Ersetzungszeichenkette

Wenn Sie in 'Ersetzungszeichenkette' ein Neue-Zeile-Zeichen haben moechten, muessen Sie es mit \ davor entwerten. Sie geben also \ <ENTER> ein. Ein solches Ersetzungskommando darf nicht Teil von 'kommandoliste' der Kommandos 'g' und 'v' sein.

'bereich' nicht angegeben:
'bereich' = .,.

[bereich]ta
kopiert 'bereich' hinter die angegebene Zeile 'a'. Als Adresse fuer 'a' ist 0 zugelassen. Zur aktuellen Zeile wird die letzte der kopierten Zeilen.

'bereich' nicht angegeben:
'bereich' = .,.

u
(u - undo) macht das letzte Kommando rueckgaengig, mit dem der Inhalt des Puffers geaendert wurde. Rueckgaengig machen koennen Sie die Kommandos 'a, c, d, g, i, j, m, r, s, t, v, G' und 'V'.

[bereich]v/rA/kommandoliste
(v - vice-versa) bearbeitet alle Zeilen mit 'kommandoliste', die 'keine' Zeichenkette enthalten, die zu 'rA' passt. 'rA' ist ein einfacher internationalisierter regulaerer Ausdruck (siehe A.1 'Regulaere Ausdruecke'). Bis auf diese Zeilenauswahl funktioniert 'v' wie das Kommando 'g'.

Nicht mit dem Kommando ! zu kombinieren!

'bereich' nicht angegeben:
'bereich' = 1,$

[bereich]V/rA/
(V - vice-versa) ist die interaktive Variante des Kommandos 'v'.
Sie koennen mit 'V' alle Zeilen bearbeiten, die 'keine'
Zeichenkette enthalten, die zu
'rA' passt. 'rA' ist ein einfacher internationalisierter
regulaerer Ausdruck (siehe A.1 'Regulaere Ausdruecke').
Bis auf diese Zeilenauswahl funktioniert 'V' wie das Kommando
'G'.

'bereich' nicht angegeben:
'bereich' = 1,$

[bereich]w[datei]
(w - write) schreibt
'bereich' in die
'datei'. Der aktuelle Dateiname aendert sich
nicht, falls er bereits gesetzt ist. Ist er nicht gesetzt, wird
'datei' zum neuen aktuellen Dateinamen.
Der alte Inhalt von 'datei' wird dabei ueberschrieben.
Existiert 'datei' noch nicht, so wird sie angelegt.
Die aktuelle Zeile
bleibt unveraendert. Nach erfolgreichem Schreiben wird die
Anzahl der geschriebenen Bytes ausgegeben, vorausgesetzt, Sie haben
den 'ed' nicht mit der Option - aufgerufen.

Wenn Sie fuer 'datei' statt eines Dateinamens das
Ausrufezeichen ! angeben, wird der
Rest der Zeile als Shell-Kommando interpretiert
und ausgefuehrt. 'bereich' ist dann die Standard-Eingabe fuer das
Shell-Kommando. Ein solches Kommando wird nicht als aktueller
Dateiname gespeichert.

'bereich' nicht angegeben:
'bereich' = 1,$

'datei' nicht angegeben:
'datei' = aktueller Dateiname

adresse
Die mit 'adresse' adressierte Zeile wird ausgegeben.

[adresse]+zahl
Dieses Kommando gibt die Zeile aus, die 'zahl' Zeilen hinter
der mit 'adresse' adressierten Zeile liegt.

'adresse' nicht angegeben:
'adresse' = .

[adresse]=
Die Nummer von 'adresse' wird ausgegeben. Die aktuelle
Zeile wird dadurch nicht veraendert.

'adresse' nicht angegeben:
'adresse' = $

!kommando
Der Rest der Zeile hinter dem ! wird als Shell-Kommando
interpretiert und ausgefuehrt.
Ist in 'kommando' ein nicht entwertetes Prozent-Zeichen %
enthalten, wird es durch
den gespeicherten Dateinamen ersetzt.
!! wiederholt das letzte 'kommando'.
In beiden Faellen wird die
expandierte Kommandozeile ausgegeben.
Nach Beendigung
des 'kommandos' ist 'ed' wieder aktiv. Die aktuelle Zeile
wird nicht veraendert.
Nicht zu kombinieren mit den Kommandos 'g' und 'v'!

<ENTER>
Geben Sie im Kommandomodus die Taste <ENTER> alleine ein,
wird die Zeile hinter der aktuellen Zeile ausgegeben.
Dies ist mit der Eingabe .+1p gleichbedeutend.
Sie koennen so im Puffer
von einer Zeile zur naechsten springen.

Die Taste <ENTER> muessen Sie druecken, um im Kommandomodus die
Eingabe eines Kommandos oder um im Eingabemodus die Eingabe einer
Textzeile abzuschliessen.

<DEL>
Mit der Taste <DEL> koennen Sie laufende 'ed'-Kommados
unterbrechen oder die Eingabe einer Zeile abbrechen.
'ed' meldet sich anschliessend mit einem ? zurueck.

<END>
Die Wirkung der Taste <END> ist dieselbe wie beim Kommando 'q'.

Steht das Zeichen, das einen regulaeren Ausdruck oder eine
Ersetzungszeichenkette abschliesst (z.B. ein /), unmittelbar vor
einem Neue-Zeile-Zeichen, dann koennen Sie dieses Zeichen weglassen.
Die adressierte Zeile wird dann ausgegeben. Die folgenden
Beispielpaare haben die gleiche Funktion:

```
s/s1/s2      s/s1/s2/p
g/s1         g/s1/p
?s1          ?s1?
```

ARBEITEN MIT ED-SCRIPTS

'ed' liest Kommandos und einzufuegenden Text von der Standard-Eingabe.
Deshalb koennen Sie die Eingabe auch umlenken, so dass 'ed' aus einer
Datei liest. Mit

```
$ ed - datei < ed_scriptdatei > ausgabe
```

wird die Datei 'datei' editiert und mit den 'ed-'Kommandos bearbeitet, die in der Datei 'ed_scriptdatei' stehen. Option - unterdrueckt die standardmaessige Ausgabe der Meldungstexte auf den Bildschrim.

Das Arbeiten mit 'ed-scripts' hat den Vorteil, dass Sie bestimmte Kommandofolgen jederzeit reproduzieren und beliebig oft verwenden koennen. Ausserdem koennen Sie so diesen Vorgang auch als Hintergrundprozess ablaufen lassen und selbst ungestoert am Bildschirm weiterarbeiten:

```
$ ed datei < ed_scriptdatei&
```

ENDE-STATUS

```
 0   bei Erfolg
 >0  bei fehlerhaftem Aufruf des 'ed' oder bei Abbruch einer
     Prozedur aufgrund fehlerhafter 'ed-'Kommandos.
```

FEHLERMELDUNGEN

Wenn Sie beim Aufruf von 'ed' zwischen Option 'p' und 'zeichenkette' kein Leerzeichen eingeben oder 'zeichenkette' vergessen:

```
ed: -p arg missing

ed: -p argument fehlt
```

Wenn Ihnen bei 'ed'-Kommandos Fehler unterlaufen:

```
?

Syntaxfehler im Kommando
```

```
?datei

Datei 'datei' ist nicht vorhanden oder kann nicht gelesen werden.
```

Naehere Informationen bekommen Sie mit den Kommandos 'h' und 'H'. Die haeufigsten Fehlermeldungen sind:

```
line out of range

Zeile ausserhalb des gueltigen Zeilenbereichs
```

```
warning: expecting 'w'

Warnung: 'w' wird erwartet
```

```
no space after command
```

kein Leerzeichen hinter dem Kommando

unknown command

unbekanntes Kommando

bad range

ungueltige Bereichsangabe

cannot open input file

Eingabedatei kann nicht geoeffnet werden

illegal or missing delimiter

unzulaessiges oder fehlendes Trennzeichen

illegal suffix

unzulaessiges Suffix

illegal or missing filename

unzulaessiger oder fehlender Dateiname

no match

keine passende Zeichenkette gefunden

DATEIEN

'ed.hup'
   In dieser Datei werden die Daten gesichert, wenn 'ed' das
   Signal SIGHUP empfaengt (siehe 'kill()' [6b]).

'/tmp/edxxxx'
   Temporaerdatei, mit der 'ed' arbeitet.

'/usr/lib/nls/intlinfo'
   Dateiverzeichnis mit den Datenbasen, deren Dateiname den
   NLS-Variablen als Wert zugewiesen werden kann (siehe 3
   'Internationale' 'Umgebung - NLS (Native Language System)').

UMGEBUNGSVARIABLEN

LANG

beeinflusst den Zeichensatz, die Sprache der Meldungstexte und alles zusammen, was mit den NLS-Variablen einzeln beeinflusst werden kann.

LC_CTYPE
beeinflusst die Zeichenklassifizierung.

LC_COLLATE
beeinflusst die Bedeutung,
die Bereichsangaben, Aequivalenzklassen und Zeicheneinheiten in eckigen Klammern haben (siehe A.1 'Regulaere Ausdruecke').

INTERNATIONALE UMGEBUNG

Wenn Sie nicht in einer englischen, sondern in einer anderssprachigen Umgebung arbeiten, dann gibt 'ed' die Meldungstexte in dieser Sprache aus.Die Sprache wird durch die NLS-Umgebungsvariable 'LANG' definiert.

In einer internationalisierten Umgebung verarbeitet 'ed' einfache internationalisierte regulaere Ausdruecke (siehe A.1 'Regulaere Ausdruecke').

Die NLS-Variable 'LC_COLLATE' legt fest, welche Bedeutung Bereichsangaben, Aequivalenzklassen und Zeicheneinheiten in eckigen Klammern haben. 'LC_CTYPE' legt fest, welche Bedeutung Zeichenklassen in eckigen Klammern haben, und bestimmt ausserdem fuer das 'ed'-Kommando 'l', welche Zeichen als nicht-druckbar gelten.

Ist eine der Variablen 'LC_CTYPE' und 'LC_COLLATE' nicht definiert oder ist ihr die leere Zeichenkette zugewiesen, dann gilt fuer diese Variable als Standardwert der Wert von 'LANG'.

Hat eine der Variablen 'LANG, LC_CTYPE, LC_COLLATE, LC_TIME' 'LC_NUMERIC' und 'LC_MONETARY' einen ungueltigen Wert oder ist die Variable 'LANG' nicht definiert bzw. leer, dann verhaelt sich 'ed', als waere das System nicht internationalisiert.

'ed' ist 8-bit-transparent.

Weitere Informationen zur internationalen Umgebung finden Sie im Kapitel 3 'Internationale Umgebung - NLS' und im Handbuch Internationalisation in SINIX [3].

BEISPIELE

1. Beispiel fuer ein 'ed-script':

   In einer Datei sollen die ersten drei Zeilen
   durch eine Zeile mit dem Text "Adressen" und
   ueberall soll das Wort "Stachus" durch "Karlsplatz" ersetzt werden.

   Inhalt der Datei 'edscript':

   ```
   1,3c
   Adressen
   ```

```
.
1,$s/Stachus/Karlsplatz/g
w
```

Bearbeitung einer Datei mit den Kommandos aus 'edscript':

```
$ ed datei < edscriptZ>
```

Wenn 'ed' seine Kommandos nicht von der Tastatur, sondern
aus einer Datei liest, wird der Editor nach dem ersten
fuer 'ed' unverstaendlichen
Kommando verlassen.

2. Beispiel fuer ein 'here-script' (siehe 'sh'):

   In beliebigen Dateien, die beim Aufruf der Prozedur
   'xy' als Argumente uebergeben werden,
   sollen die ersten drei Zeilen
   durch eine Zeile mit dem Text "Adressen" und
   ueberall soll das Wort "Stachus" durch "Karlsplatz" ersetzt werden.

   Inhalt der Prozedurdatei 'xy':

```
for i in $*
do
ed $i << scrend
1,3c
Adressen
.
1,\$s/Stachus/Karlsplatz/g
w
scrend
done
```

   Bearbeitung der Dateien 'text1, text2' und 'text3'
   mit der Prozedur 'xy':

```
$ sh xy text1 text2 text3
```

   Die Zeichenkette << 'scrend' hinter dem 'ed'-Aufruf bewirkt, dass
   die Shell den Text bis zur Zeichenkette 'scrend' an 'ed'
   als Eingabe uebergibt. Die zweite Zeichenkette 'scrend' muss
   ohne fuehrende Leerzeichen als einziges Wort in einer Zeile stehen.
   Bei der Adressangabe 1,$ muss die Sonderbedeutung von $ mit
   \ aufgehoben werden, da die Shell sonst das nachfolgende 's'
   als Name einer Shell-Variablen interpretieren wuerde.

3. Im folgenden Beispiel werden einige 'ed-'Kommandos vorgefuehrt
   und erlaeutert:

```
$ ed                - Aufruf
P                   - Bereitzeichen * im Kommandomodus ausgeben
```

```
*a                  - Anfuegen an aktuelle Zeile, hier Dateianfang
zeile1              - Eingabetext
zeile2
zeile3
.                   - Eingabemodus beenden
*1,$p               - Zeile 1 bis letzte Zeile ausgeben
zeile1
zeile2
zeile3
*p                  - Aktuelle (= zuletzt bearbeitete) Zeile
                      ausgeben
zeile3
*n                  - Aktuelle Zeile mit Zeilennummer ausgeben
3      zeile3
*1,$n               - Zeile 1 bis letzte Zeile mit Zeilennummern
                      ausgeben
1      zeile1
2      zeile2
3      zeile3
*2c                 - Zeile 2 ersetzen durch nachfolgende Eingabe
                      mit Tabulatorzeichen
zeile2 <TAB> zeile2 <TAB> zeile2
noch eine zeile2
.                   - Eingabemodus beenden
*p                  - Aktuelle Zeile ausgeben
noch eine zeile2
*1,$n               - Zeile 1 bis letzte Zeile mit Zeilennummern
                      ausgeben
1      zeile1
2      zeile2  zeile2  zeile2
3      noch eine zeile2
4      zeile3
*4s/3/4             - In Zeile 4 Zeichen '3' suchen und ersetzen
                      durch '4'
zeile4
*n                  - Aktuelle Zeile mit Zeilennummer ausgeben
4      zeile4
*1,$s/z/Z/g         - Von Zeile 1 bis letzte Zeile alle Zeichen 'z'
                  suchen und durch Z ersetzen
*1,$p               - Zeile 1 bis letzte Zeile ausgeben
Zeile1
Zeile2  Zeile2  Zeile2
noch eine Zeile2
Zeile4
*2l                 - Zeile 2 mit nicht druckbaren Zeichen als
                  Mnemoniks ausgeben, anschl. erscheint *
                  als neues Bereitzeichen.
Zeile2>Zeile2>Zeile2\012*
noch eine Zeile2
*2r datei           - Hinter Zeile 2 Inhalt von 'datei' in Puffer
57                    einlesen und Anzahl der eingelesenen Zeichen
                      ausgeben. datei muss hierfuer existieren!
*1,$n               - Zeile 1 bis letzte Zeile mit Zeilennummern
                      ausgeben
1      Zeile1
2      Zeile2  Zeile2  Zeile2 - Diese und die zwei folgenden Zeilen
3      zeile1 aus datei         kommen aus 'datei'
```

```
   4      zeile2 aus datei
   5      letzte zeile aus datei
   6      noch eine Zeile2
   7      Zeile4


   *6,7c              - Zeile 6 bis 7 ersetzen durch nachfolgende
                        Eingabe
   allerletzte Zeile
   .                  - Eingabemodus beenden
   *1,$n              - Zeile 1 bis letzte Zeile mit Zeilennummern
                        ausgeben
   1      Zeile1
   2      Zeile2  Zeile2  Zeile2
   3      zeile1 aus datei
   4      zeile2 aus datei
   5      letzte zeile aus datei
   6      allerletzte Zeile
   *q                 - Versuch, Editor mit 'q' zu verlassen
   ?                  - Warnung
   *h                 - Fragezeichen erklaeren
                - Erklaerung: erwartetes Kommando w
   warning:expecting 'w'
                  zum Sichern des Pufferinhalts wurde nicht
                  eingegeben
   *w bsp             - Sichern des Pufferinhalts in der Datei 'bsp'
   103                - Ausgabe der Anzahl der gesicherten Zeichen
   *q                 - Editor verlassen
   $                  - Bereitzeichen der Shell
```

SIEHE AUCH

'ced', 'edit', 'ex', 'sed', 'sh', 'vi'

A.1 'Regulaere Ausdruecke'

Handbuch Internationalisation in SINIX [3]

****

ENV   UMGEBUNG BEI AUSFUEHRUNG VON KOMMANDOS AENDERN
       (SET ENVIRONMENT)

Mit 'env' koennen Sie sich die aktuellen Umgebungsvariablen und ihre Werte ausgeben lassen und sie fuer ein Kommando veraendern. 'env' liest die aktuelle Umgebung ein, aendert sie entsprechend der Angabe 'name=wert' und fuehrt das Kommando dann in der veraenderten Umgebung aus. Die schon vorhandenen Angaben fuer 'name' und 'wert' werden durch die neuen Angaben ueberschrieben und vor Ausfuehrung des Kommandos der uspruenglichen Umgebung hinzugefuegt.Die neuen Angaben bilden zusammen mit den unveraenderten Umgebungsvariablen die fuer die Ausfuehrung von 'kommando'gueltige Umgebung.
Wenn kein Kommando angegeben ist, wird die resultierende Umgebung ausgegeben.

```
-------------------------------------------------------------------------
env[ -][ name=wert]...[ kommando[ arg]...]
-------------------------------------------------------------------------


-
   die urspruengliche Umgebung
   wird ignoriert; 'kommando' wird dann exakt in der angegebenen
   Umgebung ausgefuehrt.


name=wert
   'name' ist der Name einer Variablen, die fuer
   'kommando' Gueltigkeit haben soll.

   'wert' ist der Wert von 'name', der fuer
   'kommando' Gueltigkeit haben soll.


kommando
   Name des Kommandos oder der Shell-Prozedur, die Sie unter
@  der definierten Umgebung ausfuehren lassen moechten. 'env' testet,
@  ob es eine Kommandodatei mit dem als Parameter angegebenen
@  Kommandonamen gibt.Wenn man ein Shell-Kommando hinter 'env' angibt,
@  fuehrt das in der Regel zu einer Fehlermeldung.Eine Ausnahme bilden
@  'echo', 'login', 'newgrp' und 'pwd' (siehe Shell, eingebaute
@  Shell-Kommandos). Es ist jedoch moeglich, hinter 'env'
@  eine Shell-Prozedur aufzurufen, die dann alle eingebaute
@  Kommandos der Shell enthalten darf.

arg
   Argument, z.B. Stellungs- oder Kennwortparameter, das Sie an
   'kommando' uebergeben koennen.



BEISPIELE

1. Ausgabe der aktuellen Werte der Umgebungsvariablen:

   <att> env
   ATTPATH=/bin:/usr/bin:.
   ATTSHELL=/bin/sh
   DRU01=-ws=G01
   HOME=/usr1/sysiphus
   LANG=En_US.ASCII
   LOGNAME=sysiphus
   MAIL=/usr/spool/mail/sysiphus
   PATH=/bin:/usr/bin:.
   PS1=<att>
   SHELL=/bin/sh
   SIEPATH=/bin:/usr/bin:.
   SIESHELL=/bin/sh
   TERM=97801
   TZ=MET-1
```

```
UCBPATH=/usr/ucb:/bin:/usr/bin:.
UCBSHELL=/bin/sh
USER=sysiphus
```

2. Ausgabe der geaenderten Werte der Umgebungsvariablen:

```
<att> env PATH=$HOME/proz
ATTPATH=/bin:/usr/bin:.
ATTSHELL=/bin/sh
DRU01=-ws=G01
HOME=/usr1/sysiphus
LANG=En_US.ASCII
LOGNAME=sysiphus
MAIL=/usr/spool/mail/sysiphus
PATH=/usr1/sysiphus/proz
PS1=<att>
SHELL=/bin/sh
SIEPATH=/bin:/usr/bin:.
SIESHELL=/bin/sh
TERM=97801
TZ=MET-1
UCBPATH=/usr/ucb:/bin:/usr/bin:.
UCBSHELL=/bin/sh
USER=sysiphus
```

Die Umgebungsvariable 'PATH' wurde geaendert.

3. Ausgabe der geaenderten Umgebungsvariablen mit der Option -:

```
<att> env - PATH=$HOME/proz
PATH=/usr/sysiphus/proz
```

Die urspruengliche Umgebung wird ignoriert.

4. Aufruf der Datei 'fly', die
sich in '/usr1/sysiphus/SPRUECHE', also in einem
Unterdateiverzeichnis des HOME-Dateiverzeichnisses
befindet.

Inhalt der Datei 'fly':

```
echo "Wenn hinter $1 $1 $2, $2 $1 $1 nach !"
```

'fly' wird nun von einer beliebigen Stelle in Ihrem Dateibaum
aus aufgerufen, hier mit den Argumenten 'Fliegen' und 'fliegen'.

```
$ env PATH=$HOME/SPRUECHE fly Fliegen fliegen
Wenn hinter Fliegen Fliegen fliegen,
fliegen Fliegen Fliegen nach!
```

Mit der neuen Variablendefinition fuer 'PATH' legen
Sie fest, wo das eingegebene Kommando, in unserem Fall die
Datei 'fly', gesucht werden soll: in einem Unterdateiverzeichnis
des 'HOME'-Dateiverzeichnisses, das Sie mit dem Wert der Variablen
'HOME ($HOME)' angeben koennen.

Als Argumente uebergeben Sie an die Stellungsparameter $1 und $2 die Zeichenketten 'Fliegen' und 'fliegen'.

Der Inhalt der Datei 'fly' wird nur deshalb korrekt ausgefuehrt, weil das Kommando 'echo' ein eingebautes Shell-Kommando ist. Alle SINIX-Kommandos, die in /bin oder /usr/bin stehen, koennen durch die Veraenderung der Variablen 'PATH' nicht mehr gefunden werden. Damit die SINIX-Kommandos weiterhin ausgefuehrt werden, muss die Variable 'PATH' wie im folgenden Beispiel geaendert werden.

5. Aufruf der Datei 'loesch', die sich im Dateiverzeichnis '/usr1/sysiphus/proz' befindet. In dieser Datei steht eine Prozedur, die zwei Dateien vergleicht und die eine loescht, wenn die Dateien gleich sind.

   Inhalt der Datei 'loesch':

```
if cmp -s $1 $2
then
rm $2
fi
```

   Aufruf von 'loesch' von einer beliebigen Stelle in Ihrem Dateibaum aus mit den Argumenten 'dat1' und 'dat2':

```
$ env PATH=$PATH:$HOME/proz loesch dat1 dat2
```

   Hier wurde an den urspruenglichen Suchpfad der neue angefuegt, damit sowohl die Prozedur 'loesch' als auch die in der Prozedur enthaltenen SINIX-Kommandos ausgefuehrt werden koennen. Wenn nur der Suchpfad fuer 'loesch' angegeben wird, wird folgender Fehler gemeldet:

```
/usr1/sysiphus/proz/loesch: cmp: not found
```

SIEHE AUCH

'sh, set'

****

EX ZEILENORIENTIERTER EDITOR

'ex' ist ein zeilenorientierter Texteditor.

'ex' bietet Ihnen verschiedene Bearbeitungsmodi.

- Im 'ex'-Eingabe-Modus koennen Sie direkt Text eingeben.
- Im 'ex'-Kommando-Modus koennen Sie Kommandos eingeben, um zum Beispiel

- die Schreibmarke zu positionieren
- Textmuster mit regulaeren Ausdruecken zu suchen (und zu ersetzen)
- in eine andere Datei zu wechseln
- eine Shell aufzurufen.

Zusaetzlich koennen Sie von dem zeilenorientierten Editor 'ex' in den bildschirmorientierten Editor 'vi' wechseln.

Aufbau dieser Beschreibung

Nach der Beschreibung des Aufrufs von 'ex' auf SINIX-Ebene finden Sie folgende Abschnitte:

Arbeitsweise des 'ex'

- Modi des 'ex'
- Editor-Puffer sichern und 'ex' verlassen
- Voreinstellung
- Aktuelle und sekundaere Datei
- Regulaere Ausdruecke
- Ersetzungszeichenketten
- Puffer
- Fehler- und Signalbehandlung

'ex'-Kommandos

- Adressen
- Parameter
- Kommandos

'ex'-Optionen

---

ex[option][datei]...

---

option

-l
(l - lisp) Der Lisp-Modus wird eingeschaltet.
Der Text wird so eingerueckt,
dass er ein fuer Lisp-Programme typisches Aussehen bekommt.
Die folgenden 'vi'-Kommandos erhalten eine Bedeutung fuer Lisp:
oeffnende runde Klammer (, schliessende runde Klammer), oeffnende geschweifte Klammer {, schliessende geschweifte Klammer }, doppelte oeffnende eckige Klammer [[und doppelte schliessende eckige Klammer]].

-r
(r - recover) Stellt Ihre 'ex'-Sitzung wieder her, falls
das System oder der 'ex' waehrend der Sitzung

abgestuerzt ist.
Bei einem Absturz des Systems oder des 'ex'-Editors
werden die Aenderungen, die nur im Editor-Puffer stehen, nicht
in die Datei geschrieben. SINIX versucht jedoch den Inhalt des
Puffers zu retten, indem eine Kopie des
Pufferinhalts angelegt wird - falls moeglich.
Die Wirkung der Option '-r' haengt davon ab,
ob Sie fuer den Parameter 'datei' eine Datei angegeben haben.

'datei' nicht angegeben:
Es wird eine Liste aller geretteten Dateien ausgegeben.

'datei' angegeben:
'datei' wird mit den Aenderungen, die Sie
vor dem Absturz gemacht haben, in den 'vi'-Puffer geholt.
Sie koennen nun die Datei weiter editieren oder
die Aenderungen in eine Datei schreiben.

-R
(R - read-only) 'datei' wird nur zum Lesen geoeffnet. Damit
koennen Sie ein versehentliches Ueberschreiben von 'datei'
verhindern.
Jedoch koennen Sie den Pufferinhalt
im Readonly-Modus in eine Datei mit anderem Namen schreiben.

Vorsicht

Die Datei kann im 'ex'-Kommando-Modus mit 'w!'
dennoch ueberschrieben werden.

-v
(v - vi) Der Editor 'vi' wird aufgerufen. Eine ausfuehrliche
Beschreibung des 'vi' finden Sie in diesem Handbuch; eine
Einfuehrung in den 'vi' im Handbuch Einfuehrung [2], 5 'Editoren',
5.2 'Der Editor vi. Eine Beispielsitzung'.

+[kommando]
Positioniert beim Aufruf des 'ex' auf eine bestimmte Zeile der
zu editierenden Datei oder fuehrt ein 'ex'-Kommando aus.
Wenn Sie auf eine Zeile positionieren, steht die gewuenschte
Zeile danach in der Mitte des Bildschirms.
Wenn Sie ein 'ex'-Kommando ausfuehren lassen,
wird nach Ausfuehrung des 'ex'-Kommandos auf die
letzte Zeile der Datei positioniert - falls das
'ex'-Kommando keine Positionierung bewirkt (z.B. Suchen).

'kommando' nicht angegeben:
'ex' positioniert auf das Ende der Datei.

'kommando' angegeben:
'kommando' kann entweder eine Zeilenangabe ('n') sein,
ein Suchkommando (/'muster') oder ein sonstiges Kommando
des 'ex' ("'ex-kommando'"). Es wird
beim Aufruf des 'ex' ausgefuehrt:

n
'n' ist eine ganze Zahl.
'ex' positioniert auf die 'n'-te Zeile der Datei.

/muster
'ex' positioniert auf die Zeile, die 'muster'
enthaelt. Falls 'muster' Sonderzeichen enthaelt,
muessen Sie die Sonderzeichen
entwerten, damit die Shell diese nicht interpretiert.

'ex-kommando'
"ex-kommando"
'ex-kommando' kann irgendein 'ex'-Kommando sein.
Das 'ex'-Kommando muss in einfachen
Hochkommata oder Anfuehrungszeichen
eingeschlossen werden, damit es von der Shell
nicht interpretiert wird. Falls nicht schon durch
das 'ex'-Kommando positioniert wird,
positioniert der 'ex' auf die letzte Zeile
der Datei.

Beispiel

Nach dem Aufruf des 'ex' mit

ex +/Dienstag termine

wird die Datei 'termine' geoeffnet und die Schreibmarke
auf die erste Zeile der Datei positioniert, die das Wort
'Dienstag' enthaelt (vgl. das Beispiel bei 'vi').

-
Alle interaktiven Ausgaben des Editors werden unterdrueckt.
Die Option Bindestrich
'-' setzen Sie, wenn 'ex' seine Kommandos aus einem
Kommandoskript lesen soll.

datei
Name der Datei, die Sie editieren moechten.
Wenn Sie mehrere Dateien angeben, werden sie in der Reihenfolge
bearbeitet, in der Sie diese angegeben haben.
Mit dem 'ex'-Kommando 'n' wechseln Sie in die naechste Datei.

'datei' nicht angegeben:
'ex' oeffnet einen leeren Editor-Puffer, in den Sie Text schreiben
koennen. Erst beim Zurueckschreiben des
Pufferinhalts mit 'w' (write) in eine Datei oder durch
ein 'f-'Kommando bestimmen Sie deren Namen.

ARBEITSWEISE

'ex' arbeitet immer mit einem Editor-Puffer. Bei Beginn einer
'ex'-Sitzung wird eine Arbeitskopie der Datei, die Sie bearbeiten,
im Editor-Puffer angelegt.
Alle Ihre Aenderungen werden zunaechst
im Editor-Puffer vorgenommen. Sie werden erst gesichert, wenn Sie den

Pufferinhalt mit 'w' (write) in die Originaldatei zurueckschreiben.
Danach koennen Sie 'ex' mit 'q' (quit) verlassen.
Wenn Sie 'ex' verlassen wollen, ohne die Aenderungen in die
Orginaldatei zurueckzuschreiben, muessen Sie 'q!' eingeben,
ohne ein vorheriges 'w'!

Ist in den Eingabedateien das ASCII-Zeichen NUL
(siehe A.4 'Moegliche Zeichensaetze') enthalten, so wird es
geloescht; in den Ausgabedateien kann es daher nicht enthalten sein.

Modi des ex

'ex' bietet Ihnen zwei Arbeitsmodi zur Bearbeitung einer Datei:

- den 'ex'-Kommando-Modus und
- den 'ex'-Eingabe-Modus.

Zusaetzlich koennen Sie von 'ex' in den 'vi' wechseln, in dem
wieder verschiedene Arbeitsmodi zur Verfuegung stehen
(ausfuehrlicher siehe 'vi', 'Modi des vi').

Nach dem Aufruf befindet 'ex' sich im Kommando-Modus, den Sie
am Bereitzeichen Doppelpunkt : am Bildschirm erkennen.
Mit den Kommandos 'a' (append), 'i' (insert) und 'c'
(change) wechseln
Sie in den Eingabe-Modus, in dem Sie Text im Puffer ergaenzen und
aendern koennen (siehe 'ex-Kommandos, Kommandos').

Im Eingabe-Modus werden alle folgenden Eingabezeichen, auch
verschiedene nicht-druckbare Zeichen, in den Puffer geschrieben.
Kommandos werden im Eingabe-Modus nicht als solche interpretiert.
Verlassen koennen Sie den Eingabe-Modus, indem Sie in der
ersten Spalte einer neuen Zeile
einen Punkt . eingeben und danach <ENTER> druecken.

Editor-Puffer sichern und ex verlassen

Um den Editor-Puffer zu sichern oder den 'ex' zu verlassen,
muss sich der 'ex' im 'ex'-Kommandomodus befinden.

Mit folgendem Kommando sichern Sie den Inhalt Ihres Editor-Puffers
in die editierte oder eine angegebene Datei:

w[datei]
   (w - write) Der Inhalt des Editor-Puffers wird in die
   angegebene Datei gesichert.

   'datei' nicht angegeben:
   Der Inhalt des Editor-Puffers wird in
   die editierte Datei geschrieben.

Um den 'ex' zu verlassen, haben Sie folgende Moeglichkeiten:

q
   (q - quit) 'ex' verlassen.

Funktioniert nur, falls noch keine Aenderungen
am Editor-Puffer vorgenommen wurden oder der geaenderte
Editor-Puffer in eine Datei gesichert wurde.

q!
(q - quit) 'ex' verlassen; am Editor-Puffer vorgenommene
Aenderungen gehen verloren.

x
'ex' verlassen und den geaenderten Editor-Puffer
in die editierte Datei schreiben.

wq[datei]
(wq - write and quit) 'ex' verlassen und
den geaenderten Editor-Puffer
in die angegebene Datei 'datei' schreiben.

'datei' nicht angegeben:
Der Inhalt des Editor-Puffers wird in
die editierte Datei geschrieben.

Voreinstellung

Sie koennen den 'ex' in begrenztem Rahmen an Ihre Beduerfnisse
und Gewohnheiten anpassen. Dazu muessen Sie das 'ex'-Kommando
'se' (set) verwenden, mit dem Sie bestimmte Optionen setzen
oder veraendern koennen (siehe 'ex-Optionen').
Diese Aenderungen gelten
in der aktuellen Sitzung. Wie Sie diese Aenderungen dauerhaft
machen koennen, ist beschrieben bei 'vi',
Abschnitt Voreinstellung des 'vi'.

Aktuelle und sekundaere Datei

Die Datei, die, bzw. deren Kopie im Augenblick editiert wird,
wird als aktuelle Datei bezeichnet. Die sekundaere Datei ist
die Datei, die zuletzt bei einem Editier-Kommando angegeben wurde.
Falls die sekundaere Datei zur aktuellen Datei wurde, wird
die vorhergehende aktuelle Datei zur sekundaeren Datei.
Die aktuelle Datei koennen Sie mit dem Prozentzeichen %,
die sekundaere Datei mit dem Nummernzeichen # angeben; in
Dateinamen wird % durch den Namen der aktuellen Datei und #
durch den Namen der sekundaeren Datei ersetzt.

Beispiel

w %.bak
sichert die aktuelle Datei in eine Datei mit dem
gleichen Namen, aber der Endung '.bak'.

Regulaere Ausdruecke

Die Bedeutung von Sonderzeichen in regulaeren Ausdruecken
haengt bei 'ex' davon ab, ob die Option 'magic'

gesetzt ist (vgl. 'ex-Optionen').

'magic' gesetzt:

zeichen
   Ein einfaches Zeichen steht fuer sich selbst. Die folgenden
   Zeichen sind keine einfachen Zeichen, sondern Metazeichen:

   ^ (Dach) am Anfang eines Musters

   $ (Dollar-Zeichen) am Ende eines Musters

   * (Stern) ueberall ausser am Anfang eines Musters

   . (Punkt) an beliebiger Stelle in einem Muster

   [(oeffnende eckige Klammer) an beliebiger Stelle in einem Muster

   ~
      (Tilde) an beliebiger Stelle in einem Muster

   Metazeichen haben eine besondere Bedeutung und muessen durch einen
   Gegenschraegstrich \ entwertet werden, wenn Sie ihre
   Sonderbedeutung verlieren sollen.

^
   Am Anfang eines Musters steht ^ fuer den Zeilenanfang.

$
   Am Ende eines Musters steht $ fuer das Zeilenende.

.
   Ein beliebiges Zeichen.

\<
\>
   Die Zeichen \< passen zum Beginn eines Wortes.
   Das Wort muss dabei mit einem Buchstaben, einer Ziffer oder einem
   Unterstreichungszeichen _ beginnen; vor dem Wort muss entweder der
   Zeilenanfang stehen oder ein Zeichen, das nicht zu den
   oben aufgefuehrten gehoert. Die Zeichen
   \> passen zum Ende eines Wortes.

[zeichenkette]
   Ein beliebiges Zeichen aus 'zeichenkette', wobei 'zeichenkette'
   eine nicht leere Folge von Zeichen ist. Innerhalb von
   'zeichenkette' gibt es folgende Sonderbedeutungen:

   - Ein Bindestrich - zwischen zwei Zeichen definiert einen
      Bereich, zum Beispiel '[a-z]' passt zu 'a', 'z' und
      allen Zeichen, die in der aktuellen Sortierreihenfolge
      dazwischen liegen (das sind z.B. in der amerikanischen
      Umgebung 'LANG=LC_COLLATE=En_US.ASCII' alle Kleinbuchstaben).

   - Ein Dach ^ als erstes Zeichen in 'zeichenkette'
      kehrt die Bedeutung von '[zeichenkette]' um:

Ein beliebiges Zeichen, das nicht in 'zeichenkette'
enthalten ist.

Diese Sonderbedeutungen koennen durch einen
Gegenschraegstrich aufghoben werden.

Null-, ein- oder mehrmaliges Auftreten des vorausgehenden
regulaeren Ausdrucks.

~
Passt zu der Ersetzungszeichenkette, die beim letzten
's'-Kommando (substitute, siehe 'ex-Kommandos, Kommandos')
verwendet wurde.

\(muster\)
Ein regulaerer Ausdruck 'muster' kann
in durch Gegenschraegstrich \
entwertete runde Klammern eingeschlossen werden. Dies dient
nur dazu, den regulaeren Ausdruck zu identifizieren, wenn
er in Ersetzungszeichenketten verwendet werden soll
(siehe 'Ersetzungszeichenketten').

rs
Eine Folge 'rs' von zwei regulaeren Ausdruecken 'r' und
's' ist wieder ein regulaerer Ausdruck.
'rs' passt zu allen Zeichenketten, die aus einer zu 'r'
passenden Zeichenkette, gefolgt von einer zu 's' passenden
Zeichenkette, bestehen.

'nomagic' gesetzt:

Ist 'nomagic' gesetzt, so haben nur die Zeichen
- Dach ^ am Anfang eines Musters
- Dollar $ am Ende eines Musters sowie
- Gegenschraegstrich \
  eine Sonderbedeutung.

Die Zeichen
- Punkt .
- Stern *
- oeffnende eckige Klammer [
- und Tilde ~

haben nur dann eine Sonderbedeutung, wenn
sie mit einem Schraegstrich \ davor entwertet worden sind.

Ersetzungszeichenketten

Das kommerzielle Und & (Gegenschraegstrich kommerzielles Und
\& bei 'nomagic')
steht fuer die Zeichenkette, zu der das Muster passt und die
daher ersetzt werden soll.

Das Zeichen Tilde
~ (\~ bei 'nomagic') wird durch die
Ersetzungszeichenkette des letzten 's'-Kommandos (substitute)

ersetzt.

Die Zeichenkette \'n' ('n' ist eine ganze Zahl) wird durch
die Zeichenkette ersetzt, zu der das im
'n'-ten Klammernpaar \(...\) enthaltene Muster passt.

Ist in einer Ersetzungszeichenkette die Zeichenkette
\'u' oder \'l' enthalten, so wird das erste Zeichen in der
Ersetzungszeichenkette, das unmittelbar auf \'u' (upper)
oder \'l' (lower) folgt,
in einen Grossbuchstaben (bei 'u') bzw. Kleinbuchstaben (bei
'l') umgewandelt, falls es sich bei
diesem Zeichen um einen Buchstaben

handelt. Mit den Zeichen \'U' oder \'L' werden
alle Buchstaben bis zum
Ende der Ersetzungszeichenkette bzw. bis zu den Zeichen
\'e' oder \'E' in Gross- bzw. Kleinbuchstaben umgewandelt.

Puffer

Es gibt einen unbenannten und 26 alphabetische Puffer.

In diese Puffer kann Text kopiert werden ('ya' - yank) oder
wird geloeschter Text gesichert ('d' - delete), der dann
mit 'pu' (put) zurueckgeholt werden kann.

Die Kommandos 'd', 'pu' und 'ya' arbeiten mit dem
unbenannten Puffer, wenn Sie keinen alphabetischen Puffer angeben.
D.h. auch wenn Sie z.B. bei einer Loeschoperation keinen Puffer
angeben, wird das Ergebnis Ihrer Loeschoperation im
unbenannten Puffer gesichert.

Sie koennen in 26 alphabetischen
Puffern Textbloecke sichern. Die Puffer
werden mit den Kleinbuchstaben a-z oder mit den Grossbuchstaben A-Z
bezeichnet.
Verwenden Sie Kleinbuchstaben, wird der alte Pufferinhalt
mit dem neuen ueberschrieben, der alte somit geloescht.
Verwenden Sie Grossbuchstaben, wird der alte Pufferinhalt
nicht ueberschrieben, sondern der neue Text an den alten angehaengt.

Fehler- und Signalbehandlung

Tritt waehrend einer 'ex'-Sitzung ein Fehler auf, so sendet
der 'ex' an die Datensichtstation das
Zeichen BEL (akustisches Signal; siehe A.4 'Moegliche Zeichensaetze')
und gibt eine Fehlermeldung aus.

Bei einem Unterbrechungssignal kehrt 'ex' zusaetzlich in den
'ex'-Kommando-Modus zurueck. Sie koennen nun ein
'ex'-Kommando eingeben.
Liest 'ex' die Eingabe aus einer Datei, so wird 'ex'
verlassen.

EX-KOMMANDOS

Im Unterschied zu den meisten 'vi'-Kommandos, die sofort
vom 'vi' interpretiert und ausgefuehrt werden, muessen
'ex'-Kommandos mit <ENTER> "abgeschickt" werden.

Kommandozeilen, die mit Anfuehrungszeichen " beginnen, werden
ignoriert. Auf diese Weise koennen Sie in ein Kommandoskript
Kommentarzeilen einfuegen.

Adressen

Mit einer Adresse geben Sie eine bestimmte Zeile an.
Einige 'ex'-Kommandos erwarten eine oder mehrere Adressen, um dann
zum Beispiel die
angegebene Zeile oder den Bereich zwischen zwei
angegeben Zeilen einschliesslich zu bearbeiten.

Fuer 'ex' existiert zu jedem Zeitpunkt eine aktuelle Zeile.
Sie wird durch einen Punkt . adressiert.
Die aktuelle Zeile ist in der Regel die Zeile, die
zuletzt durch ein Kommando
bearbeitet wurde oder auf die gezielt (z.B. durch
Suchen) positioniert wurde.

Die Adressen trennen Sie voneinander durch
ein Komma , oder einen Strichpunkt ; .
Diese Adressenliste arbeitet 'ex' von
links nach rechts ab.

- Durch das Trennzeichen Strichpunkt ; wird die zuerst adressierte
  Zeile zur aktuellen Zeile, erst dann wird die naechste
  Adresse ausgewertet.

- Beim Trennzeichen Komma , werden alle Adressen
  von der aktuellen Zeile
  aus ermittelt. Die aktuelle Zeile aendert
  sich erst bei der Durchfuehrung eines Kommandos.

Wenn Sie bei einem Kommando mehr Adressen angegeben haben, als das
Kommando erwartet, werden alle Adressen ausser der letzten
(wenn das Kommando eine Zeile als Adresse erwartet) oder den
letzten beiden (wenn das Kommando einen Zeilenbereich erwartet)
ignoriert. Sind bei einem Kommando zwei Adressen erforderlich, so
muss sich die zuerst adressierte Zeile im Puffer vor der danach
adressierten Zeile befinden. Wenn Sie keine Adresse angeben, verwendet
'ex' standardmaessig die aktuelle Zeile.

adresse

.
  aktuelle Zeile des Puffers.

$

letzte Zeile des Puffers.

n
'n-'te Zeile im Puffer.
Die Zeilen sind sequentiell von 1 ab durchnumeriert.

'x
die mit dem Buchstaben 'x' markierte Zeile.
'x' muss ein Kleinbuchstabe sein (siehe 'ex-Kommandos'
'ma' bzw. 'k'). ''x' ist die Adresse der markierten Zeile.
Bevor 'ex' ein Positionierkommando
durchfuehrt, wird die momentan
aktuelle Zeile markiert. Mit 2 Hochkommata ''
koennen Sie zu ihr zurueckkehren.

/rA/
Ein einfacher regulaerer Ausdruck in /.../ eingeschlossen (siehe
A.1 'Regulaere Ausdruecke') adressiert die
erste Zeile, beginnend mit der
aktuellen Zeile, die eine Zeichenkette
enthaelt, die zu dem regulaeren Ausdruck passt.
Falls notwendig, springt 'ex' vom
Ende des Puffers an seinen Anfang,
um die Suche fortzusetzen (siehe 'ex-Optionen', 'wrapscan').
Soll die zu 'rA' passende Zeile
nur ausgegeben werden, koennen Sie
den zweiten Schraegstrich / weglassen.

'rA' nicht angegeben:
das zuletzt angegebene Suchmuster wird verwendet.

?rA?
Wie '/rA/', nur wird rueckwaerts gesucht.

+n
-n
Beginnt eine Adresse mit Plus- oder Minuszeichen,
gefolgt von einer Dezimalzahl 'n' (optional), ist die Zeile
adressiert, die 'n' Zeilen hinter (+) bzw. vor (-)
in der aktuellen Zeile liegt.
.+3, +3 und +++ haben die gleiche Wirkung.

adr+
adr-
Endet eine Adresse mit einem Plus oder Minus,
ist die Zeile adressiert, die
eine Zeile hinter (+) bzw. vor (-)
in der durch die Adresse bezeichneten
Zeile liegt. Mit - wird die Zeile vor
der aktuellen Zeile adressiert.
Auch am Ende einer Adresse haben + und - kumulativen Effekt;
die Adresse ++ adressiert somit die
zweite Zeile, d.h. die Zeile, die auf die aktuelle Zeile folgt.

%
Das Prozent-Zeichen % ist gleichbedeutend mit dem Adressenpaar 1,$,
d.h. dem gesamten Puffer.

Parameter

Bei den 'ex'-Kommandos werden folgende Parameter benutzt:

zeile = 'adresse'
  eine bestimmte einzelne Zeile, die Sie in einem der im
  Abschnitt Adressen angegebenen Formate angeben koennen.

  'zeile' nicht angegeben:
  der Standard-Wert Punkt . gilt.
  Punkt . ist die aktuelle Zeile.

bereich = 'adresse,adresse'
bereich = 'adresse;adresse'
  Angabe eines Bereiches zwischen zwei Zeilen einschliesslich.
  Die Adressen koennen durch ein Komma , oder einen Strichpunkt ;
  (siehe oben) voneinander getrennt sein.
  'bereich' sollten Sie nicht zusammen mit einer Zahl 'n'
  angeben, da sonst die 'letzte' Adresse des angegebenen Bereiches
  zur 'Anfangs'adresse eines Bereiches wird, der einschliesslich
  dieser Zeile 'n' Zeilen umfasst. Damit ist auf jeden Fall ein
  Bereich adressiert, der hinter dem gemeinten 'bereich' liegt.
  In der Syntax der 'ex'-Kommandos ist dies beruecksichtigt.

  'bereich' nicht angegeben:
  Der Standard-Wert .,. gilt.
  .,. (nur die aktuelle Zeile)

n
  eine positive ganze Zahl.
  Mit 'n' geben Sie die Anzahl der zu bearbeitenden Zeilen an.

  'n' nicht angegeben:
  der Standard-Wert 1 gilt.

zusatz
  eines oder eine Kombination der 'ex'-Kommandos
  # (siehe 'nu'), 'p' und 'l'. Diese Kommandos geben eine Zeile
  in einem bestimmten Format aus und werden im Anschluss
  an das vorangestellte 'ex'-Kommandos ausgefuehrt.

  Die Leerzeichen muessen nicht in jedem Fall eingegeben werden.
  Hier sind sie aus Gruenden der Uebersichtlichkeit nicht als
  optional angegeben.

Diese Parameter koennen mit einer beliebigen Zahl von
Plus- oder Minuszeichen kombiniert werden.

Kommandos

'Uebersicht der ex-Kommandos und ihrer Abkuerzungen'

```
   ^D     | (scroll) | 'n' Zeilen ausgeben, wobei 'n'=$scroll
   ab     | abbrev   | Abkuerzung definieren
   a      | append   | Text-Zeile anfuegen
   ar     | args     | Argumentliste der Kommandozeile ausgeben
   c      | change   | Text-Zeile aendern
   co     | copy     | Text-Bereich kopieren
   d      | delete   | Text-Bereich loeschen
   e      | edit     | editierte Datei wechseln
   f      | file     | aktuellen Dateinamen ausgeben
   g      | global   | fuer alle Zeilen Kommando
          |          | ausfuehren, die zu /rA/ passen
   i      | insert   | Text-Zeile einfuegen
   j      | join     | Zeilen verbinden
   l      | list     | Text mit nicht-druckbaren Zeilen
          |          | ausgeben
   map    | ---      | Makros definieren
   k      | mark     | Zeilen markieren
   m      | move     | Text-Bereich verschieben
   ma     | mark     | Zeilen markieren
   n      | next     | naechste Datei editieren
   nu     | number   | Zeilen numerieren und ausgeben
   #      | number   | Zeilen numerieren und ausgeben
   pre    | preserve | Editor-Puffer retten
   p      | print    | Text-Bereich ausgeben
   pu     | put      | Puffer ausgeben
   q      | quit     | Editor verlassen
   r      | read     | Kopie einer Datei in Puffer einlesen
   rec    | recover  | Datei nach Systemabsturz wiederherstellen
   rew    | rewind   | Argumentliste ruecksetzen (vgl. a'r)'
   se     | set      | Optionen ausgeben und setzen
   sh     | shell    | Shell aufrufen
   so     | source   | Kommandos aus Datei lesen
   s      | substitute suchen und ersetzen
   una    | unabbrev | Definition einer Abkuerzung mit 'ab' aufheben
   u      | undo     | Kommando rueckgaengig machen
   unm    | unmap    | Makrodefinition mit 'map' aufheben
   v      | ---      | wie g, aber fuer alle Zeilen,
          |          | die nicht zu /rA/ passen
   ve     | version  | Versionsnummer des Editors ausgeben
   vi     | visual   | in 'vi' wechseln
   w      | write    | Editor-Puffer in Datei schreiben
   x      | exit     | Editor verlassen mit Sichern des
          |          | Editor-Puffers
   ya     | yank     | Zeilen in Puffer kopieren
   z      | (window) | Text-Bereich ausgeben
   !      | (escape) | Shell-Kommando ausfuehren
   & s    | (resubst)| wiederholen des letzten 'substitute'-
          |          | Kommandos
   <      | (lshift) | Text-Bereich nach links verschieben
   >      | (rshift) | Text-Bereich nach rechts verschieben
   =      | (line no)| Zeilennummer ausgeben
   ------------------------------------------------------------


Kein Kommando angegeben
  Wenn Sie nur 'zeile' oder 'bereich' angeben,
```

so wird automatisch 'p' (print) ausgefuehrt. Eine leere
Eingabe bewirkt, dass die darauffolgende Zeile ausgegeben wird
(wie bei .+1p).

<CTRL>D
(ASCII EOT) gibt die naechsten 'n' Zeilen aus, wobei 'n' den
Wert der 'ex'-Option 'scroll' hat.

ab wort text
(ab - abbreviate) Nur im 'vi' wirksam!
Wenn Sie im 'vi'-Eingabe-Modus 'wort' als komplettes Wort
(d.h. nicht nur
als Teil eines Wortes) eingeben, wird es durch die
Zeichenkette 'text' ersetzt.
Falls 'text' Sonderzeichen, z.B. <ENTER>, enthalten soll, muessen
Sie dem Sonderzeichen <CTRL>V voranstellen.

zeile a
(a - append) 'ex' wechselt in den
Eingabe-Modus; der Text wird hinter der
angegebenen Zeile eingefuegt. Bei 'zeile' gleich 0 wird
der Text ganz an den Anfang des Puffers gesetzt. Die aktuelle
Zeile ist die zuletzt eingegebene bzw., falls keine Eingabe
erfolgte, die adressierte Zeile.
Die Eingabe wird mit einem Punkt . in der ersten Spalte
beendet.

ar
(ar - arguments)
Die Argumentliste aus der Kommandozeile
des 'ex'-Aufrufes wird, eingeschlossen
in eckigen Klammern [...], ausgegeben.

zeile c n bereich c
(c - change) 'ex' wechselt in den Eingabe-Modus;
es werden 'n' Zeilen bzw. die Zeilen
im angegebenen Bereich 'bereich'
durch die eingegebenen Textzeilen ersetzt.
Die letzte der eingegebenen Zeilen wird zur aktuellen Zeile; werden
keine Textzeilen eingegeben, so wird der angegebene Bereich
geloescht, d.h. 'c' wirkt wie 'd' (delete).
Die Eingabe wird mit einem Punkt . in der ersten Spalte beendet.

bereich co zeile zusatz
(co - copy) Eine Kopie des angegebenen
Bereiches 'bereich' wird hinter die
Zeile 'zeile' kopiert; hat 'zeile' die Zeilennummer 0,
wird 'bereich' an den Anfang des Puffers kopiert.

zeile d puffer n bereich d puffer
(d - delete) Die Zeilen im Bereich 'bereich'
bzw. 'n' Zeilen von der
angegebenen Zeile 'zeile' ab werden geloescht. Wird der Name
eines Puffers 'puffer' angegeben, so werden die geloeschten
Zeilen in diesem Puffer gesichert. Mit dem 'ex'-Kommando 'pu'
koennen Sie auf diesen Puffer zugreifen. Die erste auf

die geloeschten Zeilen folgende Zeile wird zur aktuellen Zeile;
befanden sich die geloeschten Zeilen am Ende des Puffers,
so wird die letzte Zeile im Puffer zur aktuellen Zeile.

e[+zeile] datei
(e - edit) Die Datei 'datei' wird editiert.
Wurden am aktuellen Editor-Puffer seit dem letzten
'w'-Kommando (write) Aenderungen vorgenommen, so wird
eine Warnung ausgegeben
und das Kommando nicht ausgefuehrt. Sie koennen jedoch
die Ausfuehrung des 'e'-Kommandos erzwingen, indem
Sie dem 'e' ein Ausrufezeichen nachstellen: 'e! datei'.
Die letzte Zeile des Puffers wird zur
aktuellen Zeile. Wird dieses Kommando jedoch von 'vi' aus
aufgerufen, so wird die erste
Zeile im Puffer zur aktuellen Zeile.

'+zeile'
Die angegebene 'zeile' wird zur aktuellen Zeile. 'zeile'
kann sein:

- eine Zeilennummer 'n'
- das Dollarzeichen $ (Dateiende)
- ein regulaerer Ausdruck in der Form /'rA' oder ?'rA'.

f[name]
(f - file) Ausgabe des aktuellen Dateinamens, sowie
Informationen,
wie z.B. die Zahl der Zeilen und die Position der aktuellen Zeile,

'name' angegeben:
'name' wird zum aktuellen Dateinamen

bereich g/rA/kommandoliste
(g - global) Zunaechst werden alle Zeilen innerhalb
des Bereiches 'bereich'
markiert, zu denen das durch 'rA' angegebene Muster passt.
Dann wird das angegebene Kommando 'kommando' bzw. werden die
in 'kommandoliste' enthaltenen Kommandos ausgefuehrt, wobei
die jeweils markierte Zeile zur aktuellen Zeile wird.

Die Kommandos in 'kommandoliste' koennen in mehreren Zeilen
stehen, vorausgesetzt, die Neue-Zeile-Zeichen sind mit
einem Gegenschraegstrich \ entwertet.
Ist 'kommandoliste' leer, so werden alle Zeilen ausgegeben.
Die Kommandos
'a(ppend)', 'c(hange)', und 'i(nsert)' sind erlaubt;
am Ende von 'kommandoliste' koennen Sie den abschliessenden
Punkt weglassen.
Das 'visual'-Kommando ist ebenfalls erlaubt und liest die
Eingabe von der Datensichtstation.

Folgende Kommandos duerfen in 'kommandoliste'
nicht enthalten sein:
'global' und 'undo'.

```
    Folgende Optionen duerfen in 'kommandoliste'
    nicht enthalten sein:
    'autoprint', 'autoindent' und 'report'.

    Beispiel

       1,$g/^\*/s//+/

       Von Zeile 1 bis Dateiende ($) werden in der ersten Spalte
       ^ alle * durch das Substitute-Kommando ('s') durch
       ein Plus-Zeichen (+) ersetzt.

  zeile i
    (i - insert) 'ex' wechselt in den Eingabe-Modus;
    der eingegebene Text wird
    vor der angegebenen Zeile eingefuegt. Die letzte der eingegebenen
    Zeilen wird zur aktuellen Zeile. Erfolgt keine
    Eingabe, so wird die Zeile vor der adressierten Zeile 'zeile'
    zur aktuellen Zeile.
    Die Eingabe wird mit einem Punkt . in der ersten Spalte
    beendet.

  zeile j n zusatz
  bereich j zusatz
    (j - join) Fasst die mit 'bereich' angegebenen Zeilen bzw. 'n'
    Zeilen von 'zeile' ab zu einer
    Zeile zusammen. Zwischen den ehemaligen Zeilen steht mindestens
    ein Leerzeichen. Wird eine Zeile durch einen Punkt abgeschlossen, so
    werden zwei Leerzeichen eingesetzt.
    Beginnt die anzufuegende Zeile mit einer schliessenden runden
    Klammer ), wird kein Leerzeichen eingefuegt.
    Zusaetzliche Leer- und Tabulatorzeichen am
    Anfang einer Zeile entfallen.

    Wird an das Kommando 'j' ein Ausrufezeichen ! angehaengt,
    werden an die Nahtstellen zwischen zwei verbundenen Zeilen keine
    Leerzeichen gesetzt.


  zeile l n zusatz
  bereich l zusatz
    (l - list) Die angegebenen Zeilen werden ausgegeben;
    Tabulatorzeichen werden durch ^I ersetzt; das Ende jeder Zeile wird
    durch ein Dollarzeichen $ markiert. Der einzige sinnvolle 'zusatz'
    ist hier #, wenn den Zeilen ihre Zeilennummern vorangestellt werden
    sollen. Die letzte der ausgegebenen Zeilen wird zur aktuellen Zeile.
@   'list' unterscheidet sich von dem Kommando 'print' nur dadurch, dass
@   bei 'print' Tabulatorzeichen und Zeilenende-Zeichen nicht mit
@   ausgegeben werden.

  map x kommandos
  map! x text

    Nur im 'vi' wirksam!

    Format 1: Makrodefinition fuer 'vi'-Kommando-Modus
```

Definieren von Makros, die im 'vi'-Kommando-Modus
verwendet werden koennen. Beim
ersten Argument 'x' handelt es sich um ein einzelnes Zeichen
oder die Zeichenkette #'n', wobei
'n' eine Ziffer ist, mit der die
Funktionstaste 'n' angegeben wird.
Wird im 'vi'-Kommando-Modus dieses Zeichen bzw.
diese Funktionstaste eingegeben, so verhaelt
'vi' sich so, als ob 'kommandos' eingegeben wird.
'kommandos' wird dabei als Folge von
'vi'-Kommandos interpretiert.
Sind in 'kommandos' Sonderzeichen,
Leerzeichen oder Neue Zeile-Zeichen enthalten,
so muessen sie mit einem
<CTRL>V entwertet werden.

Beispiel 1

Sie wollen ein Makro definieren, das in der gesamten Datei das
Zeichen Stern * am Zeilenanfang (Spalte 1) durch ein Leerzeichen
ersetzt. Der Name dieses Makros soll "Stern" * sein. Damit
Sie dieses Makro stets zur Verfuegung haben, machen Sie in Ihrer
'.exrc'-Datei (siehe Abschnitt Voreinstellung des 'ex')
folgenden Eintrag:

```
:map * :1,$s/^\*/ /
```

Beispiel 2

Sie moechten die Funktionstaste <F1> belegen.
Es soll die naechste Zeile gesucht werden, an deren
Anfang ein bestimmter regulaerer Ausdruck 'rA'
steht; diese Zeile soll dann automatisch geloescht werden.
Dazu geben Sie (aus dem 'vi'-Kommando-Modus) ein:

```
:map <CTRL>V<F1> /^rA<CTRL>V<ENTER>dd<ENTER>
```

Format 2: Makrodefinition fuer 'vi'-Eingabe-Modus

Definieren von Makros, die im 'vi'-Eingabe-Modus
verwendet werden koennen (vgl. Format 1).
'Jedes' im 'vi'-Eingabe-Modus eingegebene
'x' wird durch 'text' ersetzt.
(Bei dem Kommando 'ab' wird 'x' nur dann ersetzt, wenn
es allein steht.)

Wenn 'x' nicht durch 'text' ersetzt werden soll, muss
'x' ein <CTRL>V vorangestellt werden.

Format 1 und Format 2:

Ein definiertes Makro koennen Sie mit:

unmap x bzw.

unmap! x

wieder aufheben.

zeile ma x
zeile k x
(ma/k - mark) 'zeile' wird mit 'x' markiert; 'x' muss
ein einzelner Kleinbuchstabe sein, dem ein Leer- oder
Tabulatorzeichen vorangestellt ist. An der Adresse der aktuellen
Zeile aendert sich nichts.

bereich m zeile
(m - move) 'bereich' wird hinter 'zeile' versetzt.
Die erste der versetzten Zeilen wird zur aktuellen Zeile.

n[datei]...

'datei' nicht angegeben:
(n - next) Die naechste der in der Kommandozeile aufgefuehrten
Dateien (Argumentliste) wird editiert (vgl. 'f' und 'ar').
Wurden am aktuellen Editor-Puffer seit dem letzten
'w'-Kommando (write) Aenderungen vorgenommen, so wird
eine Warnung ausgegeben
und das Kommando nicht ausgefuehrt. Sie koennen jedoch
die Ausfuehrung des 'n'-Kommandos erzwingen, indem
Sie dem 'n' ein Ausrufezeichen nachstellen: 'n!'.

'datei' angegeben:
Die bisherige Argumentenliste wird durch die angegebene(n)
Datei(en) ersetzt; die erste angegebene Datei wird editiert.

zeile nu n zusatz
bereich nu zusatz
zeile # n zusatz
bereich # zusatz
(nu - number) Die Zeilen werden, versehen mit den entsprechenden
Zeilennummern, ausgegeben. Der einzige hier sinnvolle Zusatz ist
'l'.
Die letzte der ausgegebenen Zeilen wird zur aktuellen Zeile.

pre
(pre - preserve) Der Inhalt des aktuellen Puffers wird gesichert,
als ob es zu einem Absturz des Systems gekommen waere. 'pre'
wird in Notfaellen verwendet, wenn z.B. 'w' (write)
nicht ausgefuehrt werden kann, weil die Platte voll ist,
und keine andere Moeglichkeit
zur Rettung des Pufferinhaltes vorhanden ist.

zeile p n
bereich p
(p - print) Die angegebenen Zeilen werden ausgegeben, wobei
nicht-druckbare Zeichen als Steuerzeichen in der Form
^x ausgegeben werden; DEL wird in Form von ^? ausgegeben.
Die zuletzt ausgegebene Zeile wird zur aktuellen Zeile.
@ 'print' unterscheidet sich von dem Kommando 'list' dadurch,
@ dass es Tabulatorzeichen und Zeilenende-Zeichen nur dann ausgibt,
@ wenn zusaetzlich vorher die Option 'set list' gesetzt wurde.

zeile pu puffer
  (pu - put) Die mit 'd' (delete) geloeschten oder mit 'y' (yank)
  in einem Puffer gesicherten Zeilen werden
  hinter der angegebenen Zeile 'zeile' eingefuegt.
  'puffer' ist der Name eines alphabetischen Puffers.

  'puffer' nicht angegeben:
  Der Text wird aus dem unbenannten Puffer geholt.

q
  (q - quit) Der Editor wird verlassen.
  Wurde der Puffer seit dem letzten
  'w'-Kommando (write) veraendert, so wird eine Warnung
  ausgegeben und 'q' wird nicht ausgefuehrt. Sie koennen jedoch
  die Ausfuehrung des 'q'-Kommandos erzwingen, indem
  Sie dem 'q' ein Ausrufezeichen nachstellen: 'q!'.
  Alle noch nicht (mit 'w') gesicherten Aenderungen am Editor-Puffer
  gehen dann verloren.

zeile r[datei]
  (r - read) Eine Kopie von 'datei' wird im
  Editor-Puffer hinter 'zeile' eingelesen.
  Die Zeilennummer von 'zeile' kann dabei 0 sein;
  die Textzeilen werden dann an den Anfang des Puffers gesetzt.
  Gibt es keine aktuelle Datei (Aufruf von
  'ex' ohne Dateinamen) wird 'datei' zur aktuellen Datei.
  Die letzte der
  eingelesenen Zeilen wird zur aktuellen Zeile; im 'vi' wird
  die erste der eingelesenen Zeilen zur aktuellen Zeile.

  'datei' nicht angegeben:
  die aktuelle Datei wird eingelesen.

  Wird statt 'datei' !'sinixkmdo' angegeben, so wird 'sinixkmdo'
  als SINIX-Kommando interpretiert und an die Shell uebergeben.
  Die Ausgabe des SINIX-Kommandos wird in den Puffer eingelesen.

rec datei
  'datei' wird nach einem Absturz des Systems oder des Editors
  wiederhergestellt.

rew
  (rew - rewind) Die Argumentliste (siehe 'ar')
  wird zurueckgesetzt, und die
  erste Datei in der Liste wird editiert.
  Wurde der Editor-Puffer seit dem letzten
  'w'-Kommando (write) veraendert, so wird eine Warnung
  ausgegeben und 'rew' wird nicht ausgefuehrt. Sie koennen jedoch
  die Ausfuehrung des 'rew'-Kommandos erzwingen, indem
  Sie dem 'rew' ein Ausrufezeichen nachstellen: 'rew!'.
  Alle noch nicht (mit 'w') gesicherten Aenderungen am Editor-Puffer
  gehen dann verloren.

se[parameter]
  (se - set) Mit dem 'set'-Kommando koennen Optionen aufgelistet

bzw. gesetzt werden. Es gibt zwei Typen von Optionen:
Optionen mit Boole'schen Werten und Optionen mit nicht-Boole'schen
Werten. Ein Beispiel fuer den Boole'schen Typ ist z.B. die
Option 'number'. Ist sie gesetzt, werden Zeilennummern ausgegeben,
ist 'nonumber' gesetzt, werden keine Zeilennummern ausgegeben.
Ein Beispiel fuer den nicht-Boole'schen Typ ist 'report'.
Der Wert dieser Option (Standard = 5) bestimmt die Anzahl der
Zeilen, die durch ein Kommando veraendert werden muss, damit
'ex' und 'vi' eine Meldung geben.

'parameter'

all
die Werte aller Optionen werden ausgegeben.

option?
der gegenwaertige Wert der angegebenen Option wird ausgegeben.

option
Nur fuer Optionen mit Boole'schen Werten:
die Option wird gesetzt.

nooption
Nur fuer Optionen mit Boole'schen Werten:
die Option wird nicht gesetzt.

option=wert
Nur fuer Optionen mit nicht-Boole'schen Werten:
der Option wird der Wert 'wert' zugewiesen.

parameter nicht angegeben:
'set' gibt diejenigen 'ex'-Optionen aus, deren Werte vom
Benutzer festgelegt wurden und daher von den Standard-Werten
abweichen.

Naehere Informationen ueber die 'ex'-Optionen finden Sie
im Abschnitt 'ex'-Optionen.

sh
Die Shell (siehe 'sh') wird aufgerufen. Nach Beendigung der
Shell wird die Editorsitzung an derselben Stelle fortgesetzt.

so datei
Kommandos werden aus 'datei' gelesen und ausgefuehrt.
'so'-Kommandos koennen ineinander verschachtelt werden.

zeile s/rA/ersatz/schalter n zusatz
bereich s/rA/ersatz/schalter zusatz
(s - substitute) Das erste Auftreten des
mit 'rA' (regulaerer Ausdruck)
getroffenen Musters in jeder Zeile des
angegebenen Bereiches wird durch die Zeichenkette
'ersatz' ersetzt (siehe 'Regulaere Ausdruecke'
und 'Ersetzungszeichenketten').
Der angegebene Bereich ist entweder 'bereich' oder
umfasst 'n' Zeilen von 'zeile' ab.

'schalter'

g
(g - global) 'alle' Zeichenketten in der Zeile,
zu denen 'rA' passt werden ersetzt.

c
(c - confirm) zunaechst wird die Zeile
ausgegeben, wobei die zu ersetzende Zeichenkette
in der darunterliegenden Zeile
durch ein ^ markiert wird. Die Eingabe des Buchstabens
'y' bewirkt dann, dass die Substitution durchgefuehrt wird;
bei jeder anderen Eingabe wird das Kommando abgebrochen. Die
Zeile, in der zuletzt eine Substitution durchgefuehrt wurde,
wird zur aktuellen Zeile.

una wort
(una - unabbreviate) 'wort' wird aus der Liste der
Abkuerzungen gestrichen (vgl. 'ab').

u
(u - undo) Die mit dem letzten Editierkommando
vorgenommenen Aenderungen werden
rueckgaengig gemacht. 'g' (global) und 'vi' (visual)
werden in diesem Fall
als eigenstaendige Kommandos betrachtet.
Kommandos, die die aeussere
Umgebung veraendern (z.B. 'w' (write), 'e' (edit) und 'n'
(next)) koennen nicht rueckgaengig gemacht werden.

Ein 'undo'-Kommando selbst kann mit 'u' wieder
rueckgaengig gemacht werden.

Bei 'u' gehen alle Markierungen fuer Zeilen, die
geaendert und dann wieder
zurueckgeholt wurden, verloren.

unm x
unm! x
(unm - unmap) Die mit 'map' vorgenommene Definition
des Makros 'x' wird aufgehoben.

bereich v/rA/kommandoliste
(v - vice versa) Hat dieselbe Funktion wie 'global',
nur wird 'kommandoliste' auf alle Zeilen angewandt,
zu denen das Muster 'rA' nicht passt.

ve
Die Versionsnummer des Editors wird ausgegeben.

zeile vi[stelle] n
Bei der angegebenen 'zeile' wird in den 'vi' gewechselt.
Der optionale Parameter 'stelle' kann eines der
Zeichen Bindestrich - oder Punkt . sein;

wie beim Kommando 'z' wird damit die Position
von 'zeile' auf dem Bildschirm angegeben.
Standard: oberer Rand des
Bildschirms. Mit 'n' wird die Groesse des Bildschirms angegeben;
standardmaessig hat er die Groesse, die ueber die 'ex'-Option
'window' definiert ist. Mit <ESCAPE> 'Q' wird der 'vi'
wieder verlassen und wieder in den 'ex' gewechselt.
Naehere Information siehe 'vi'.

[bereich]w[!][datei]
[bereich]wq[!][datei]

Format 1: In Datei schreiben

(w - write) Der angegebene 'bereich'
wird in die 'datei' geschrieben, wobei die Anzahl
der geschriebenen Zeilen und Zeichen ausgegeben wird.

Wird eine (vorhandene) 'sekundaere' Datei angegeben, so wird das
Kommando nicht ausgefuehrt, damit die sekundaere Datei nicht
versehentlich ueberschrieben wird. Die Ausfuehrung von 'w' kann
jedoch erzwungen werden, indem ein Ausrufezeichen ! angefuegt
wird: 'w! #'.

Soll am Ende von 'datei' angefuegt werden, so
muss das Kommando in der Form 'w>>' angegeben werden; existiert
'datei' nicht, so wird eine Fehlermeldung ausgegeben.

Wird statt 'datei' !'sinixkmdo' angegeben, so wird
'sinixkmdo' als SINIX-Kommando interpretiert
und an die Shell uebergeben;
der angegebene 'bereich' ist dann fuer das Kommando die
Standard-Eingabe.

'bereich' nicht angegeben:
die ganze aktuelle Datei wird nach 'datei' geschrieben.

'datei' nicht angegeben:
standardmaessig wird die aktuelle Datei verwendet. Wenn weder eine
aktuelle Datei vorhanden ist noch eine 'datei' angegeben wird, so
kann 'w' nicht ausgefuehrt werden.

Format 2: In Datei schreiben und 'ex' verlassen

(wq - write and quit) 'wq' hat dieselbe Funktion wie 'w',
auf das 'q' folgt; 'wq!' hat
dieselbe Funktion wie 'w!', auf das 'q' folgt.

x
(exit) Der Editor wird verlassen; wurden seit dem letzten Sichern
mit 'w' (write) Aenderungen vorgenommen, so werden diese
zuvor in die Datei geschrieben.

zeile ya puffer n
bereich ya puffer
Der angegebene 'bereich' bzw. 'n' Zeilen von 'zeile' ab werden

in den angegebenen 'puffer' gesichert.

'puffer' nicht angegeben:
Wird kein Puffer angegeben, so wird der unbenannte Puffer verwendet.

zeile z[stelle] n
(window) 'n' Zeilen aus der Umgebung von
'zeile' werden ausgegeben.

'n' nicht angegeben:
'n' hat den Wert der Variablen 'window'.

'stelle'
Fuer den optionalen Parameter 'stelle' kann ein
Bindestrich - oder Punkt
. angegeben werden; das Zeichen - bewirkt,
dass die 'zeile' an den unteren
Rand des ausgegebenen Bereiches gesetzt wird; das Zeichen . bewirkt,
dass sich die Zeile in der Mitte des ausgegebenen Bereiches
befindet.
Die letzte der ausgegebenen Zeilen wird zur aktuellen Zeile.

'stelle' nicht angegeben:
'zeile' befindet sich am oberen Rand des ausgegebenen Bereiches.

! kommando
(escape) Die auf das Ausrufezeichen ! folgenden Zeichen werden als
Shell-Kommando interpretiert und an den Kommandointerpreter
uebergeben. Wurden am Puffer seit dem letzten
'w'-Kommando (write) Aenderungen vorgenommen, so wird eine Warnung
ausgegeben. Dann wird das Kommando ausgefuehrt.
Wenn die 'Ausgabe des Kommandos' nicht umgelenkt wird,
erscheint sie auf dem Bildschirm, aendert aber an der
Datei nichts. Allerdings kann das 'Kommando selbst' die
Datei aendern, z.B. '!cat /etc/profile >> %'.
Nach erfolgreicher Ausfuehrung des Kommandos wird ein
! ausgegeben. An der Adresse der aktuellen Zeile aendert
sich nichts.

Kommen in 'kommando' das Prozentzeichen % und das Nummernzeichen
vor, so werden sie als Dateinamen expandiert
(siehe 'Aktuelle und sekundaere Datei').

Ein Ausrufezeichen ! in 'kommando' wird durch das
vorherige !-Kommando ersetzt. Mit !! wird also das letzte
!-Kommando wiederholt. Die so erweiterten Kommandozeilen
werden auf dem Bildschirm ausgegeben.

bereich ! kommando
(escape) Bei einem !-Kommando in dieser Form werden
die mit 'bereich' angegebenen Zeilen an 'kommando' als
Standard-Eingabe uebergeben und durch die Ausgabe von 'kommando'
ersetzt.

'bereich' nicht angegeben:
'bereich' wird nicht durch .,. ersetzt, sondern es gilt

die andere Form des !-Kommandos.

"
Kommandozeilen, die mit Anfuehrungszeichen " beginnen, werden ignoriert. Auf diese Weise koennen Sie in ein Kommandoskript Kommentarzeilen einfuegen.

zeile &schalter n zusatz
bereich&schalter zusatz
zeile sschalter n zusatz
bereichsschalter zusatz
(resubst) Das letzte 's'-Kommando (substitute) wird wiederholt, als ob '&' durch das letzte 's/rA/ersatz/' ersetzt wuerde. Dieselbe Funktion hat ein 's'-Kommando, in dem '/rA/ersatz/' weggelassen wird.

zeile < n
bereich <
(lshift) Die Zeilen innerhalb des angegebenen Bereichs bzw. 'n' Zeilen, die auf 'zeile' folgen, werden um 'shiftwidth' Positionen nach links verschoben. Leer- und Tabulatorzeichen gehen dabei verloren; die uebrigen Zeichen bleiben unveraendert erhalten. Die letzte der geaenderten Zeilen wird zur aktuellen Zeile.

zeile > n
bereich >
(rshift) Die Zeilen innerhalb des angegebenen Bereichs bzw. 'n' Zeilen, die auf 'zeile' folgen, werden durch Einfuegen von Leer- und Tabulatorzeichen um 'shiftwidth' Positionen nach rechts verschoben.

zeile=
(line number) Die Zeilen-Nummer der angegebenen 'zeile' wird ausgegeben. Die Position der aktuellen Zeile wird nicht veraendert.

'zeile' nicht angegebnen:
Die Zeilen-Nummer der letzten Zeile wird ausgegeben.

EX-OPTIONEN

Mit den Optionen des 'ex' kann das Verhalten der Editoren 'ex' und (vor allem) 'vi' beeinflusst werden (siehe 'vi', 'Voreinstellung des vi').

Fuer alle Optionen gibt es Standard-Einstellungen, die beim Aufruf von 'ex' oder 'vi' wirksam werden.
Mit dem 'ex'-Kommando 'se' (set) koennen Sie sich alle Optionen ausgeben lassen:

set all<ENTER>

Sie koennen aber auch mit diesem Kommando eine oder mehrere Optionen

nach Ihren Vorstellungen veraendern:

set showmode scroll=15<ENTER>

In diesem Fall wird im 'vi' der Eingabe-Modus in der
Status-Zeile angezeigt und die Anzahl der Zeilen, um die <CTRL>'D'
den Bildschirm rollt, auf '15' gesetzt.

Mit

set<ENTER>

werden Ihnen alle Optionen ausgegeben, die von den Standard-Werten
abweichen.

Es gibt zwei Typen von Optionen:
Optionen mit Boole'schen Werten und Optionen mit nicht-Boole'schen
Werten. Ein Beispiel fuer den Boole'schen Typ ist z.B. die
Option 'showmode'.
Ist diese Option nicht gesetzt, hat sie den Namen 'noshowmode'.
Ein Beispiel fuer den nicht-Boole'schen Typ ist 'scroll'.

autoindent, ai
noautoindent, noai (Standard)
   Ist die Option 'autoindent' gesetzt, so wird im Eingabe-Modus
   jede Zeile entsprechend der vorhergehenden Zeile eingerueckt (es
   werden Leer- und Tabulatorzeichen eingefuegt). Der Beginn der
   Einrueckung wird bestimmt von der Zeile,
   hinter die angefuegt ('a',)
   vor die eingefuegt ('i') oder die als erste geaendert ('c')

   wird. Zusaetzliche Einrueckungen koennen wie gewoehnlich
   durchgefuehrt werden, die folgenden Zeilen werden dann automatisch
   auf die neue Grenze eingerueckt.

   Soll der Text weniger eingerueckt werden, so
   koennen sie durch Eingabe von
   <CTRL>D den Zeilenanfang um 'shiftwidth' Positionen nach links
   verschieben. Mit einem Dach ^, gefolgt von einem <CTRL>'D', wird
   die Einrueckung fuer die aktuelle
   Zeile aufgehoben; mit einer Null '0',
   gefolgt von einem <CTRL>D werden alle Einrueckungen aufgehoben.

autoprint, ap
noautoprint, noap (Standard)
   Nach jedem Kommando, mit dem der Inhalt der Editor-Puffers
   geaendert wurde, wird die aktuelle Zeile ausgegeben. Bei
   globalen Suchen/Ersetzen-Kommandos (siehe 'g', 's' und 'v')
   wird 'autoprint' unterdrueckt.

autowrite, aw
noautowrite, noaw (Standard)
   Der Inhalt des Editor-Puffers wird in die aktuelle
   Datei geschrieben, wenn
   Aenderungen vorgenommen worden sind und eines der Kommandos
   'e' (edit), 'n' (next), 'rew' (rewind) oder

'!' (escape) aufgerufen wird.

beautify, bf
nobeautify, nobf (Standard)
Alle Steuerzeichen ausser dem Tabulatorzeichen, dem Neue Zeile- und dem Formularvorschubzeichen werden aus dem Eingabetext entfernt.

directory=dvz
dir=dvz
Mit 'dvz' wird das Dateiverzeichnis angegeben, in welchem der Editor-Puffer angelegt werden soll. Hat der Benutzer fuer dieses Dateiverzeichnis keine Schreiberlaubnis, so wird der Editor verlassen.

edcompatible, ed
noedcompatible, noed
Ist das 's'-Kommando (substitute) mit den Schaltern 'g' und 'c' versehen, so werden diese Schalter gespeichert; die nochmalige Angabe der Schalter hebt ihre Wirkung auf.

ignorecase, ic
noignorecase, noic (Standard)
Alle Grossbuchstaben im Text werden beim Suchen mit regulaeren Ausdruecken wie Kleinbuchstaben behandelt. Gleichfalls werden alle Grossbuchstaben in regulaeren Ausdruecken wie Kleinbuchstaben behandelt, ausser in Ausdruecken fuer Zeichenklassen.

lisp
nolisp (Standard)
Die Option 'autoindent' wird gesetzt und die 'vi'-Kommandos oeffnende runde Klammer (, schliessende runde Klammer), oeffnende geschweifte Klammer {, schliessende geschweifte Klammer }, doppelte oeffnende eckige Klammern [[und doppelte schliessende eckige Klammern]] werden angepasst.

list
nolist (Standard)
In den ausgegebenen Zeilen werden Tabulatorzeichen durch ^I ersetzt und das Ende jeder Zeile durch ein $ markiert.

magic (Standard)
nomagic
Aendert die Interpretation der in regulaeren Ausdruecken und Ersetzungszeichenketten verwendeten Zeichen. Siehe Abschnitt Regulaere Ausdruecke und Ersetzungszeichenketten.

number, nu
nonumber, nonu (Standard)
Die Zeilen werden mit vorangestellten Zeilennummern ausgegeben.

paragraphs=zeichenkette
para=zeichenkette
Dieser Option wird eine Zeichenkette zugewiesen. Jeweils zwei in dieser Zeichenkette enthaltene Zeichen stehen fuer den Namen eines 'nroff'-Makros,

das einen Absatz (Paragraphen) einleitet. Ein Makro wird in
einen Text im Format '.XX' eingetragen, wobei der Punkt
. das erste Zeichen der Zeile ist.

prompt (Standard)
noprompt
Ist 'prompt' gesetzt, so wird im Kommando-Modus
das Bereitzeichen Doppelpunkt : angezeigt; andernfalls wird
kein Bereitzeichen ausgegeben.

redraw (Standard)
noredraw
Werden Aenderungen vorgenommen, so sollen diese sofort auf dem
Bildschirm ausgegeben werden. Wegen der grossen Menge der
ausgegebenen Daten ist dies jedoch nur sinnvoll, wenn die
Daten mit einer hohen Geschwindigkeit uebertragen werden.

remap (Standard)
noremap
Ist 'remap' gesetzt, so werden bei der Makroumsetzung auch
Makros beruecksichtigt, die durch andere Makros definiert sind. Die
Umsetzung wird fortgesetzt, bis das gewuenschte Makro erstellt ist.
Ist diese 'noremap' gesetzt, so wird lediglich eine einstufige
Umsetzung durchgefuehrt.

report=n
Wenn die Zahl der Zeilen, die mit dem letzten Kommando
geaendert, geloescht oder kopiert wurde, groesser als 'n' ist,
erscheint in der Status-Zeile eine Meldung.

scroll=n
Der bei 'scroll' angegebene Wert 'n' steht fuer die
Anzahl von Zeilen, um die das Fenster bei einem
<CTRL>'D' verschoben werden soll, und die
bei einem 'z'-Kommando (der zweifache Wert von 'scroll')
ausgegeben werden sollen.

sections=zeichenkette
Dieser Option wird eine Zeichenkette zugewiesen. Jeweils
zwei in dieser Zeichenkette enthaltene Zeichen stehen fuer
den Namen eines 'nroff'-Makros,
das einen Abschnitt (Section) einleitet. Ein Makro wird in
einen Text im Format '.XX' eingetragen, wobei der Punkt
. das erste Zeichen der Zeile ist.

shiftwidth=n
sw=n
Der bei 'sw' angegebene Wert 'n' steht fuer die
Schrittweite eines Software-Tabulators
um die der Text bei gesetzter Option 'autoindent' und
den Kommandos Kleinerzeichen < und Groesserzeichen
> verschoben werden soll.

showmatch, sm
noshowmatch, nosm (Standard)

Wird im 'vi' eine schliessende runde Klammer) oder eine
schliessende geschweifte Klammer }
eingegeben, so wird die zugehoerige
(oder { angezeigt, falls sie sich noch auf dem Bildschirm
befindet.

showmode, smd
noshowmode, nosmd (Standard)
Wenn 'smd' gesetzt ist, wird in der Status-Zeile angezeigt,
wenn sich 'vi' im 'vi'-Eingabe-Modus befindet.

slowopen, slow
noslowopen, noslow (Standard)
Im 'vi' soll die Korrektur des Bildschirminhalts
waehrend der Eingabe verzoegert werden. Dadurch soll der Durchsatz
bei nicht intelligenten Datensichtstationen erhoeht werden.

tabstop=n
ts=n
'n' gibt die Software-Tabulatoren an, die vom Editor
benutzt werden, um Tabulatoren in der
editierten Datei zu expandieren.

terse
noterse (Standard)
Ist 'terse' gesetzt, so werden gekuerzte Fehlermeldungen
ausgegeben.

warn (Standard)
nowarn
Wenn 'warn' gesetzt ist, erscheint bei einem ':!'-Kommando
die Warnung 'No write since last change', wenn seit der letzten
Sicherung mit 'w' noch Aenderungen am Editor-Puffer erfolgten.

window=n
Die Anzahl der Zeilen in einem Textfenster des 'vi'.

wrapscan, ws (Standard)
nowrapscan, nows
Ist 'wrapscan' gesetzt, so wird das Suchen mit /.../ oder ?...?,
wenn das Ende bzw. der Anfang des Editor-Puffers
erreicht ist, am Anfang
bzw. Ende fortgesetzt. Falls 'nows' gesetzt ist,
wird das Suchen am Anfang bzw. am
Ende der Datei beendet.

wrapmargin=n
wm=n
Nur im 'vi' wirksam. Wenn 'n' > 0 ist,
wird an das letzte Wort in einer Eingabezeile automatisch
ein Neue-Zeile-Zeichen angefuegt,
so dass das Zeilenende um mindestens 'n' Positionen
vom rechten Rand des Bildschirms entfernt ist.

Mit 'n'=0 (Standard) kann der automatische Zeilenumbruch
ausgeschaltet werden.

writeany, wa
nowriteany, nowa (Standard)
   Wenn 'nowa' gesetzt ist, wird bei einem Sicherungskommando
   'w' eine Pruefung auf die Existenz der Datei gemacht.
   Damit wird verhindert, dass Sie versehentlich eine
   schon existierende Datei ueberschreiben.
   Mit 'w!' wird diese Pruefung umgangen.
   Wenn 'wa' gesetzt ist, wird nicht geprueft, ob die Datei
   schon existiert.

FEHLERMELDUNGEN

Tritt bei 'ex' ein Fehler auf, so sendet der Editor an die
Datensichtstation das akustische Signal BEL
(siehe A.4 'Moegliche Zeichensaetze') und
gibt eine Fehlermeldung aus. Bei einem
Unterbrechungssignal kehrt 'ex'
auf die Kommandoebene zurueck, d.h. Sie koennen ein neues
'ex'-Kommando eingeben.
Liest der Editor die Eingabe aus einer
Datei, so bricht 'ex' bei einem Unterbrechungssignal ab.

Stellt 'ex' einen internen Fehler fest, so wird versucht, den
Puffer zu retten, wenn die zuletzt vorgenommenen Aenderungen noch
nicht abgespeichert worden sind. Durch einen erneuten Aufruf von
'ex' mit der Option '-r' kann auf die gesicherten Daten
zurueckgegriffen werden.

DATEIEN

'$HOME/.exrc'
   Datei mit Voreinstellungen fuer 'ex' und 'vi'.
   Diese Voreinstellungen sind immer wirksam.

'/usr/lib/nls/intlinfo'
   Dateiverzeichnis mit den Datenbasen, deren Dateiname den NLS-
   Variablen als Wert zugewiesen werden kann (siehe 3 'Internationale'
   'Umgebung - NLS').

UMGEBUNGSVARIABLEN

TERM
   In der Umgebungsvariablen TERM muss der Typ der
   benutzten Datensichtstation
   festgelegt sein.

EXINIT
   Umgebungsvariable mit Voreinstellungen fuer 'ex' und 'vi'.
   Diese Voreinstellungen sind nicht wirksam, falls
   im 'HOME'-Dateiverzeichnis eine
   '.exrc' Datei mit widersprechenden Voreinstellungen
   existiert.

LANG
   beeinflusst den Zeichensatz und die Sprache der Meldungstexte.

INTERNATIONALE UMGEBUNG

Wenn Sie nicht in einer englischen, sondern in einer
anderssprachigen Umgebung arbeiten, dann geben
'ex/vi' die Meldungstexte in dieser
Sprache aus. Die Sprache wird durch die NLS-Umgebungsvariable
'LANG' definiert.

'ex/vi' ist 8-bit-transparent.
Bei nicht-druckbaren 8-bit-Zeichen wird
der Oktalwert des Zeichens ausgegeben.

Weitere Informationen zur internationalen Umgebung finden Sie im
Kapitel 3 'Internationale Umgebung - NLS' und im
Handbuch Internationalisation in SINIX [3].

BEISPIEL

Es werden einige 'ex'-Kommandos vorgefuehrt und erlaeutert:

```
$ ex zahlen                 -  ex aufrufen
"zahlen" 4 lines, 20 characters
:set number                 -  Zeilennummern ausgeben
:1,$p                       -  Zeile 1 bis letzte Zeile ausgeben
     1  eins                -  Ursruenglicher Inhalt von 'zahlen'
     2  zwei
     3  drei
     4  vier
:1 c 1                      -  Zeile 1 aendern
     1  EINS                -  Neue Zeile 1
     2  .                   -  Eingabemodus verlassen
:2,3 co 4                   -  Zeile 2 bis 3 hinter Zeile 4
     6  drei                   kopieren
:1,$ g/zwei/s//ZWEI/        -  In der gesamten Datei "zwei"
                            durch "ZWEI" ersetzen
:2 a                        -  Hinter Zeile 2 neue Zeile einfuegen
     3  DREI                -  Neue Zeile 3
     4  .                   -  Eingabemodus verlassen
:4 i                        -  Vor Zeile 4 neue Zeile einfuegen
     4  VIER                -  Neue Zeile 4
     5  .                   -  Eingabemodus verlassen
:5,$ d a                    -  Zeile 5 bis Dateiende loeschen
     4  VIER
:1,$p                       -  Zeile 1 bis letzte Zeile ausgeben
     1  EINS
     2  ZWEI
     3  DREI
     4  VIER
:wq                         -  Sichern des Pufferinhalts und ex
                               verlassen
"zahlen" 4 lines, 20 characters
```

SIEHE AUCH

'vi'

Handbuch Einfuehrung [2], 5 'Editoren', 5.2 'Der Editor vi. Eine Beispielsitzung'.

Handbuch Internationalisation in SINIX [3]

****

IPCRM EINRICHTUNGEN ZUR INTERPROZESS-KOMMUNIKATION ENTFERNEN
       REMOVE INTER-PROCESS COMMUNICATION FACILITIES)

'ipcrm' entfernt ein Semaphor oder eine Nachrichten-Warteschlange, Einrichtungen zur Interprozess-Kommunikation (IPC-Einrichtungen). Nachrichten-Warteschlangen und Semaphore, die Sie entfernen moechten, koennen Sie entweder durch deren Bezeichner oder durch den Schluessel, der bei der Erzeugung der jeweiligen IPC-Einrichtung verwandt wurde, angeben. Bezeichner und Schluessel von IPC-Einrichtungen erfahren Sie mit Hilfe des Kommandos 'ipcs' (Ausgabe-Spalten 'ID' bzw. 'Key'). Was genau bei der Entfernung einer Nachrichten-Warteschlange oder eines Semaphors passiert, erfahren Sie in CES [6b], 'msgctl()' bzw. 'semctl()'.

---

ipcrm[option]...

---

option

-q msgqid
   (q - queue) Die Nachrichten-Warteschlange mit der Kennzahl 'msgqid' wird aus dem System entfernt und die damit verbundenen Datenstrukturen werden zerstoert.

   msgqid
      Kennzahl der Nachrichten-Warteschlange,die entfernt werden soll. Diese Kennzahl gibt das Kommando 'ipcs' in der Spalte 'ID' aus.

-Q msgkey
   (Q - queue) Die Nachrichten- Warteschlange mit dem Schluessel 'msgkey' wird aus dem System entfernt und die damit verbundenen Datenstrukturen werden zerstoert.

   msgkey
      Schluessel der Nachrichten-Warteschlange, die entfernt werden soll. Diesen Schluessel gibt das Kommando 'ipcs' in der Spalte 'KEY' aus.

-s semid

(s - semaphor) Das Semaphor mit der Kennzahl 'semid' wird aus dem System entfernt, und die damit verbundenen Datenstrukturen werden zerstoert.

semid
Kennzahl des Semaphors, das entfernt werden soll. Diese Kennzahl gibt das Kommando 'ipcs' in der Spalte 'ID' aus.

-S semkey
(S - semaphor) Das Semaphor mit dem Schluessel 'semkey' wird aus dem System entfernt, und die damit verbundenen Datenstrukturen werden zerstoert.

semkey
Schluessel, mit dem das Semaphor erzeugt wurde, das Sie entfernen moechten. Diesen Schluessel gibt das Kommando 'ipcs' in der Spalte 'KEY' aus.

@ Die Angabe von KEY=00000 ist bei den Schaltern -S und -Q nicht
@ zulaessig. KEY=0 wird vom System intern verwendet und entspricht
@ IPC_PRIVATE.

PORTABILITAET

'ipcrm' ist nicht im X/Open Portability Guide, Issue 3, enthalten.

BEISPIEL

Zuerst lassen Sie mit 'ipcs' einen Bericht ueber den Zustand von Einrichtungen zur Interprozess-Kommunikation ausgeben.Danach entfernen Sie die Nachrichten-Warteschlange mit der Kennzahl 40 aus dem System:

```
$ ipcs
Time is 248bc52b
IPC status from /dev/kmem as of Tue Jun  6 13:35:55 1989
T     ID     KEY        MODE       OWNER    GROUP
Message Queues:
q     40 0x0000004b -Rrw-rw-rw-  michael    qm234
Semaphores:
$ ipcrm -q40
```

SIEHE AUCH

'ipcs' 'ipc()', 'msgctl()', 'msgget()', 'msgop()', 'semctl()', 'semget()', 'semop()' [6b]

****

PS PROZESSDATEN ABFRAGEN (PROCESS STATUS)

'ps' gibt Informationen ueber Prozesse aus.

---

ps[-adefl][n namelist][-p proclist][-t termlist][-u uidlist]

Keine Option angegeben
'ps' gibt Informationen ueber Prozesse aus, die mit der kontrollierenden Datensichtstation verbunden sind. Die Ausgabe besteht aus einer kurzen Auflistung

- der Prozessnummer PID
- der Nummer der Datensichtstation TTY
- der gesamten Ausfuehrzeit TIME und
- dem Kommandonamen CMD.

Die Bedeutung der Ausgabespalten ist im Abschnitt 'Ausgabe' genauer erlaeutert.

-a
Es werden Informationen ueber alle Prozesse ausgegeben, ausser ueber Prozesse, die Prozessgruppenfuehrer sind oder die nicht mit einer Datensichtstation verbunden sind.

-d
Es werden Informationen ueber alle Prozesse ausgegeben, ausser ueber Prozesse, die Prozess-Sitzungsfuehrer sind (siehe Handbuch CES [6b], 'setsid()').

-e
Es werden Informationen ueber alle Prozesse ausgegeben.

-f
Es werden ausfuehrliche Informationen ausgegeben. Dazu untersucht 'ps' den Speicher oder den Swap-Bereich. 'ps' versucht festzustellen, welcher Kommandoname und welche Argumente angegeben wurden, als der Prozess erzeugt wurde, und gibt den Kommandonamen und seine Argumente aus. Falls dies nicht gelingt, wird der Kommandoname eingeschlossen in eckigen Klammern [...] ausgegeben.

-g grplist
Es werden nur Informationen ueber Prozesse ausgegeben, deren Prozess-Sitzungsfuehrer (siehe Handbuch CES [6b], 'setsid()') in 'grplist' angegeben werden.

grplist
'grplist' ist eine Liste von Nummern von Prozess-Sitzungsfuehrern. 'grplist' hat folgendes Format:

Die Nummern werden durch ein Komma getrennt, oder die Liste wird in Anfuehrungszeichen eingeschlossen "...", wobei die Nummern dann durch Komma und/oder Leerzeichen getrennt

werden koennen.

-l
gibt eine lange Liste aus.

-n namelist
'namelist' kann als Alternative zur Systemdatei '/vmunix' angegeben werden. '/vmunix' enthaelt die Symboltabelle des Systems.

'name_list' nicht angegeben:
Die Symboltabelle des Systems wird in der Datei '/vmunix' gesucht.

-p proclist
Es werden nur Informationen ausgegeben zu Prozessen, deren Prozessnummern in 'proclist' angegeben sind.

proclist
'proclist' ist eine Liste von Prozessummern.

Die Nummern werden durch ein Komma getrennt, oder die Liste wird in Anfuehrungszeichen eingeschlossen "...", wobei die Nummern dann durch Komma und/oder Leerzeichen getrennt werden koennen.

-t termlist
Es werden nur Informationen ueber Prozesse ausgegeben, die mit den in 'termlist' genannten Datensichtstatioenen verbunden sind.

termlist
'termlist' ist eine Liste von Datensichtstationen.
Die Datensichtstationen koennen auf zwei Arten angegeben werden: entweder mit ihrem Geraetenamen (z.B. 'tty04') oder, falls der Geraetename mit 'tty' beginnt, nur durch die Nummer (z.B. '04').

Die Datensichtstationen werden durch ein Komma getrennt, oder die Liste
wird in Anfuehrungszeichen eingeschlossen "...", wobei die Datensichtstationen dann durch Komma und/oder Leerzeichen getrennt werden koennen.

-u uidlist
Es werden nur Informationen ueber Prozesse ausgegeben, fuer die die Benutzernummer UID oder die Benutzerkennung des Prozesseigentuemers in 'uidlist' angegeben
sind. Es wird die Benutzernummer ausgegeben, ausser wenn zusaetzlich die Option '-f' angegeben wird, dann wird die Benutzerkennung ausgegeben.

Ausgabe

Im folgenden werden die Ueberschriften der Spalten und die Bedeutung der Spalten in der Ausgabe von 'ps' erlaeutert. Die Buchstaben 'f' und 'l' bezeichnen die Option, bei der die entsprechende Spalte in der Ausgabe erscheint, 'alle' bedeutet, dass die Spalte bei allen

Optionen erscheint. Beachten Sie bitte, dass Sie mit den Optionen 'f' und 'l' nur bestimmen, welche Informationen Sie zu einem Prozess erhalten, und nicht, zu welchem Prozess Sie Informationen erhalten.
@ Bei der Ausgabe von 'ps' koennen die Werte so gross werden, dass die
@ einzelnen Felder miteinander verschmelzen. Leer- oder Tabulatorzeichen
@ koennen in diesem Fall nicht mehr als Feldtrenner genutzt werden.

F ('l')
Flags des Prozesses (oktal und additiv).
Die Flags sind maschinenabhaengig und werden deshalb hier nicht angegeben.

S ('l')
Status des Prozesses.

0 Prozess wird gerade ausgefuehrt
S Prozess schlaeft, wartet auf ein Ereignis
R Prozess ist bereit zum Laufen
I Zwischenstatus: Prozess ist im Entstehen
Z Zwischenstatus: Prozess terminiert
T getraceter Prozess ist gestoppt
X Prozess wartet auf mehr Speicherplatz

UID ('f, l')
(UID - user ID) Benutzernummer des Prozess-Eigentuemers.
Wenn die Option '-f' gesetzt ist, wird statt der UID die Benutzerkennung ausgegeben.

Es werden nur die ersten 7 Zeichen der Benutzerkennung ausgegeben.

PID ('all')
(PID - process ID) Prozessnummer.
Jeder Prozess erhaelt zum Zeitpunkt seiner Erzeugung eine eindeutig bestimmte Prozess-Nummer.
Unter dieser Nummer kann z.B. der Prozess mit 'kill' beendet werden.

PPID ('f, l')
(PPID - parent process ID) Prozess-Nummer des Vaterprozesses.

C ('f, l')
Prozessor-Auslastung fuer das Scheduling.

PRI ('l')
Prioritaetswert des Prozesses. Hoehere Zahlen bedeuten niedrigere Prioritaet.

NI ('l')
Wert, mit dem die Prozess-Prioritaet veraendert wurde (siehe 'nice').

ADDR ('l')
Adresse (pysikalische Seiten-Frame-Nummer) der user-area, falls resident, andernfalls die Platten-Adresse des ausgelagerten Prozesses.

SZ ('l')
   Die Groesse in Blocks des Speicher-Abbildes (core-image)
   des Prozesses.

WCHAN ('l')
   Nummer des Ereignisses, auf das der Prozess wartet oder
   schlaeft. Wenn die Spalte leer ist, laeuft der Prozess.

STIME ('f')
   Startzeit des Prozesses. Innerhalb der ersten 24 Stunden
   wird die Uhrzeit ausgegeben, danach das internationalisierte
   Datum.

TTY ('all')
   Die kontrollierende Datensichtstation des Prozesses.

TIME ('all')
   Gesamte Ausfuehrungszeit
   des Prozesses in Minuten und Sekunden.

CMD ('all')
   Der Name des Kommandos. Der vollstaendige Kommandoname
   und seine Argumente werden ausgegeben, wenn die Option '-f'
   gesetzt ist.

Ein Prozess, der sich beendet hat und der einen Vaterprozess hat, der
bisher noch nicht auf ihn gewartet hat, wird mit
<defunct> bezeichnet.

DATEIEN

'/vmunix'
   enthaelt die Symboltabelle des Systems

INTERNATIONALE UMGEBUNG

Wenn Sie nicht in einer englischen, sondern in einer
anderssprachigen Umgebung arbeiten, dann gibt
'ps' die Meldungstexte in dieser
Sprache aus. Die Sprache wird durch die NLS-Umgebungsvariable
'LANG' definiert.

'ps' ist 8-bit-transparent.

Weitere Informationen zur internationalen Umgebung finden Sie im
Kapitel 3 'Internationale Umgebung - NLS' und im
Handbuch Internationalisation in SINIX [3].

SIEHE AUCH

'kill', 'nice', 'sh'

****

TAR DATEIEN ARCHIVIEREN UND ARCHIVE BEARBEITEN
TAPE ARCHIVER)

Mit 'tar' koennen Sie:

- Dateien oder Dateiverzeichnisse auf Magnetband, Magnetbandkassette oder Diskette archivieren (Format 1).

- den Inhalt eines Band- oder Diskettenarchivs lesen (Format 2).

- ein Archiv vollstaendig oder einzelne Dateien und Dateiverzeichnisse von Magnetband, Magnetbandkassette oder Diskette in ein Dateiverzeichnis einlesen (Format 3).

Als Systemverwalter koennen Sie mit 'tar' beliebige Dateien und Dateiverzeichnisse sichern und wiederherstellen.

Arbeitsweise des Kommandos

'tar' arbeitet mit Archiven auf einem Datentraeger. 'tar' legt ein Archiv auf einem Datentraeger an. Dabei setzt es eine Archiv-Anfangsmarke und eine Archiv-Endemarke. Mit einem 'tar'-Kommando kann immer nur ein Archiv bearbeitet werden.

Auf einer Diskette kann jeweils immer nur ein Archiv angelegt werden.

Auf einem Magnetband bzw. einer Magnetbandkassette koennen mehrere Archive angelegt werden. Mit dem Kommando 'mt' kann an den Anfang oder das Ende eines Archivs positioniert werden. 'tar' liest immer nur aus dem aktuell positionierten Archiv. Befinden sich auf einem Magnetband mehrere Archive und sollen alle eingelesen werden, muss fuer jedes Archiv ein 'tar'-Kommando gegeben werden.

Wenn Sie beim Einlesen die entsprechende Geraetedatei ohne automatisches Zurueckspulen verwenden, ist die Bandposition nach dem Einlesen eines Archives automatisch der Beginn des naechsten Archivs. Sie brauchen dann nicht mit 'mt' neu zu positionieren.

Vor dem Aufruf beachten

Bevor Sie 'tar' ausfuehren koennen, muessen Sie das Magnetband, die Magnetbandkassette bzw. die Diskette in das entsprechende Laufwerk einlegen.

Wenn Sie ein Archiv anlegen wollen, muss der Schreibschutz

des Datentraegers entfernt sein. Weitere Informationen hierzu finden Sie in der Betriebsanleitung Ihres Rechners.

Wenn Sie ein Archiv auf Diskette anlegen wollen, muss diese Diskette formatiert sein. Dazu benutzen Sie das Kommando '/etc/flformat'.

Nach dem Aufruf beachten

Wenn Sie 'tar' ausgefuehrt haben, sollten Sie sofort den Datentraeger aus dem entsprechenden Laufwerk nehmen. Ein anderer Benutzer koennte versehentlich mit einem weiteren 'tar'-Kommando Ihr Archiv beschaedigen.

```
-------------------------------------------------------------------
 tar funktion[attribut...[ argument]...]... datei ...
 tar t[v][f geraetedatei][ datei]...
 tar x[attribut...][f geraetedatei][ datei]...
-------------------------------------------------------------------
```

Format 1: Datei oder Dateiverzeichnis auf Datentraeger archivieren

tar funktion[attribut...[argument]...]... datei ...

Als Benutzer ohne Systemverwalter-Rechte koennen Sie nur die Dateien und Dateiverzeichnisse archivieren, auf die Sie zugreifen duerfen. Fuer Dateien brauchen Sie das Leserecht, fuer Dateiverzeichnisse das Ausfuehrrecht.

funktion

Fuer 'funktion' geben Sie genau einen der folgenden Buchstaben (ohne Bindestrich -) an:

c

Diskette:

'tar' legt auf der Diskette ein neues Archiv an und schreibt die angegebene Datei an den Beginn des neuen Archivs. Ein eventuell bereits vorhandenes Archiv auf dem Datentraeger wird geloescht. Werden mehrere Dateien angegeben, werden sie fortlaufend eingetragen. Wird fuer 'datei' ein Dateiverzeichnis angegeben, werden alle Unterbaeume des Dateiverzeichnisses archiviert.

Magnetband und Magnetbandkassette:

'tar' schreibt die angegebene Datei oder das angegebene Dateiverzeichnis ab der aktuellen Bandposition des eingelegten Magnetbandes bzw. Magnetbandkassette. Auf einem Magnetband kann mit dem Kommando 'mt' an das Ende eines Archivs positioniert werden. So koennen mit 'tar c' auf einem Magnetband mehrere Archive aneinandergehaengt werden. Sie koennen verhindern, dass das Band nach dem Zugriff des 'tar'-Kommandos zurueckgespult wird, indem Sie die entsprechende Geraetedatei ansprechen

(siehe A.3 'Geraetedateien fuer Datentraeger' im Anhang).

Befinden sich mehrere Archive auf einem Magnetband und Sie ueberschreiben eines, so sind alle folgenden Archive nicht mehr lesbar.

Beispiel

Sie haben vier Archive auf einem Band. Sie positionieren auf den Beginn des zweiten und beschreiben ab dieser Position das Band mit einem neuen Archiv. Die alten Archive 2, 3 und 4 sind damit ueberschrieben.

Vorsicht

Passt eine Datei nicht mehr auf den Datentraeger, bricht 'tar' mit folgender Fehlermeldung ab:
tar: write error
Die letzte Datei ist dann nicht ordnungsgemaess gesichert.

r

Diese Funktion gilt nur fuer Disketten oder Plattenarchive. Die angegebene Datei bzw. die Dateien werden an das Ende eines bereits bestehenden Archivs angehaengt.

u

Diese Funktion gilt nur fuer Disketten oder Plattenarchive. Die angegebene Datei bzw. Dateien werden dann an das Ende eines bereits bestehenden Archivs angehaengt, wenn

- sie noch nicht im Archiv vorhanden sind, oder
- die Zeit der letzten Aenderung der Datei im Archiv frueher liegt, als die Zeit der letzten Aenderung der zu archivierenden Datei.

attribut

Die ausgewaehlte Funktion kann durch Angabe eines oder mehrerer Funktionsattribute gesteuert werden. Diese Attribute werden ohne Leerzeichen an die Funktion angefuegt. Einige Attribute erwarten ausserdem die Eingabe von Argumenten. Dabei ist folgendes zu beachten:

1. Geben Sie zuerst alle Attribute ohne Leerzeichen ein.
2. Geben Sie dann die Argumente getrennt durch Leerzeichen an. Die Reihenfolge der Argumente wird bestimmt durch die Reihenfolge, in der die zugehoerigen Attribute eingegeben wurden.

'attribut' kann sein:

v

(v - verbose) 'tar' gibt zu jeder Datei, die gerade bearbeitet wird, folgende Informationen aus:

a pfadname n blocks

Dabei bedeutet:
a (a - append) die angegebene Datei wurde in das Archiv eingetragen.
pfadname der Pfadname, unter dem die Datei eingetragen wurde.
n blocks die Anzahl der Bloecke, die die Datei belegt. Die Blockgroesse ist abhaengig vom verwendeten Medium.

'v' nicht angegeben:
'tar' gibt keine Meldungen aus.

w

'tar' erwartet fuer jede Datei eine Bestaetigung, bevor die Aktion ausgefuehrt wird. 'tar' mit dem Attribut 'w' gibt aus:

r dateiname:

Dabei bedeutet:
dateiname der Name der Datei oder des Dateiverzeichnisses, das archiviert werden soll

Nach dem Doppelpunkt : erwartet 'tar' Ihre Eingabe.
Nur wenn Sie bejahen, fuehrt 'tar' die Aktion aus, sonst nicht.
In welcher Sprache Sie Ihre Antwort eingeben koennen, haengt vom Wert der NLS-Variablen 'LANG' ab. Wenn der Wert von 'LANG' mit 'En' fuer Englisch beginnt, bejahen Sie mit 'y' und verneinen mit 'n', wenn er mit 'De' fuer Deutsch beginnt, bejahen Sie mit 'j' und verneinen mit 'n' (siehe 'Internationale Umgebung').

'w' nicht angegeben:
'tar' fuehrt die Aktion ohne Bestaetigung aus.

f geraetedatei

'tar' erwartet die Eingabe einer Geraetedatei, in die geschrieben werden soll, wenn Sie nicht die Standard-Geraetedatei benutzen wollen. 'tar' interpretiert das zugehoerige Argument als Geraetedatei.

geraetedatei
Name der Geraetedatei, in die geschrieben werden soll. Normalerweise verwenden Sie hier die Geraetedateien fuer Disketten-, Magnetband- oder Magnetbandkassettenlaufwerke, soweit Sie solche Geraete an Ihrem Rechner angeschlossen haben. Die Geraetedateien fuer alle Geraete befinden sich im Dateiverzeichnis '/dev'.

Geben Sie fuer 'geraetedatei' einen Bindestrich - an, dann schreibt 'tar' auf die Standard-Ausgabe. Auf diese Weise koennen Sie 'tar' in einer Pipe verwenden. So kann 'tar' benutzt werden, um Dateiverzeichnisse oder Dateisysteme zu kopieren oder zu verschieben (siehe Beispiel).
@ Achten Sie darauf, dass die Option 'v' (verbose) auf keinen Fall
@ gesetzt sein darf, wenn 'tar' seine Daten auf die Standard-Ausgabe
@ schreiben soll, denn 'verbose' benutzt ebenfalls die Standard-

@ Ausgabe. Die Meldungen werden somit in die zu kopierenden
@ Dateien geschrieben. Dadurch werden die Dateien unlesbar.

'f geraetedatei' nicht angegeben:
'tar' schreibt auf die Standard-Geraetedatei '/dev/mt0',
die Geraetedatei fuer das 1/2-Zoll Magnetbandlaufwerk.

l
(l - link) 'tar' meldet, wenn ein Verweis auf andere Dateien nicht aufgeloest werden kann. Dies gilt nicht fuer symbolische Verweise.

'l' nicht angegeben:
'tar' gibt keine Fehlermeldung aus, wenn ein Verweis nicht aufgeloest werden kann.

b blockungsfaktor
Dieses Attribut koennen Sie nur verwenden, wenn Sie ein Magnetband bzw. Magnetbandkassette mit 'tar' beschreiben. 'tar' interpretiert das dazugehoerige Argument als Blockungsfaktor.

blockungsfaktor
Als Blockungsfaktor koennen Sie eine Zahl zwischen 1 und 20 angeben.

Der Blockungsfaktor gibt an, wieviele Band-Bloecke 'tar' pro Bandzugriff schreiben soll. Je hoeher der Blockungsfaktor, desto mehr Daten passen auf das Band.

'b blockungsfaktor' nicht angegeben:
Blockungsfaktor = 1

argument
Zu den Attributen 'f' und 'b' gehoeren Argumente ('geraetedatei' bzw. 'blockungsfaktor'). Diese Argumente geben Sie anschliessend an die Funktion und alle Attribute an, getrennt durch Leerzeichen. Das Kommando 'tar' ordnet die Argumente anhand ihrer Reihenfolge den Attributen zu. Das heisst: geben Sie die Attribute 'f' und 'b' in der Reihenfolge 'fb' ein, so wird das erste folgende Argument als 'geraetedatei' und das zweite als 'blockungsfaktor' interpretiert.

Beispiel

```
$ tar cfvb /dev/rfl2 10 *.c
      |         |  |
      |         |  Dateien, die archiviert werden sollen
      |         |
      |         blockungsfaktor zu b
      geraetedatei zu f
```

datei
Name der Datei bzw. des Dateiverzeichnisses, die bzw. das archiviert werden soll. Mindestens ein Name muss angegeben sein. Sie koennen auch mehrere Dateien oder Dateiverzeichnisse angeben.

Ist die angegebene Datei ein Dateiverzeichnis, dann archiviert 'tar' dieses Dateiverzeichnis mit allen darin enthaltenen Dateien und Unterdateiverzeichnissen.

Den Dateinamen koennen Sie auch mit Sonderzeichen der Shell angeben.

Ist eine Datei zu gross fuer einen Datentraeger, dann muss sie vor dem Archivieren mit dem Kommando 'split' geteilt werden. Sie kann dann spaeter mit dem Kommando 'cat' wieder zusammengefuegt werden.

Format 2: Inhalt eines Band- oder Diskettenarchivs lesen

tar t[v][f geraetedatei][datei]...

t

'tar' gibt den Namen der angegebenen Datei aus, falls sie im Archiv steht. Ist keine Datei angegeben, gibt 'tar' das Inhaltsverzeichnis des gesamten Archivs aus.

v

(v - verbose) 'tar' gibt zu jeder Datei, die im Archiv gelesen wird, folgende Informationen aus:

- die gesetzten Zugriffsrechte
- Benutzernummer/Gruppennummer (UID/GID)
- Groesse in byte
- Datum und Uhrzeit der Erstellung der Datei
- den Namen der Datei

Das Format, in dem Datum und Uhrzeit ausgegeben werden, haengt ab vom Wert der NLS-Umgebungsvariablen 'LC_TIME' bzw. 'LANG' (siehe 'Internationale Umgebung').

Beispiel

```
rw------- 33/1   1045 Mar 16 15:01 1988 liste
```

b

'v' muss ohne Leerzeichen an 't' angefuegt werden.

'v' nicht angegeben:
Das Kommando 'tar' gibt keine Meldungen aus.

f geraetedatei

'tar' erwartet die Eingabe einer Geraetedatei, von der gelesen werden soll, wenn Sie nicht die Standard-Geraetedatei benutzen wollen (siehe Format 1).

datei

Name der Datei, deren Name ausgegeben werden soll, wenn sie im

Archiv steht. Sie koennen mehrere Dateien angeben.

Ist die angegebene Datei ein Dateiverzeichnis, dann gibt 'tar' dieses Dateiverzeichnis mit allen darin enthaltenen Dateien und Unterdateiverzeichnissen aus.

'datei' nicht angegeben:
'tar' gibt den Inhalt des gesamten Archivs aus.

Format 3: Dateien von Datentraeger einlesen

tar x[attribut]...[f geraetedatei][datei]...

x

'tar' liest die angegebene Datei vom Archiv ein. Steht die Datei mit relativem Pfadnamen im Archiv, so wird sie in das aktuelle Dateiverzeichnis kopiert. Steht die Datei mit absolutem Pfadnamen im Archiv, so wird sie in das entsprechende Dateiverzeichnis kopiert.

Als Benutzer ohne Systemverwalter-Rechte brauchen Sie das Schreibrecht fuer das Dateiverzeichnis, in das eingelesen wird.

Ist die angegebene Datei ein Dateiverzeichnis, so werden alle darin enthaltenen Dateien und Unterdateiverzeichnisse kopiert.

Existiert die angegebene Datei noch nicht in dem Dateiverzeichnis, in das sie kopiert werden soll, so wird diese Datei angelegt.

Benutzernummer des Eigentuemers, Gruppennummer und Zeit der letzten Aenderung werden von der archivierten Datei uebernommen. Den archivierten Benutzer- und Gruppennummern werden die Namen zugewiesen, die in den Dateien '/etc/passwd' bzw. '/etc/group' eingetragen sind. Gibt es in diesen Dateien keinen Eintrag zu den archivierten Nummern, so gehoert die kopierte Datei der entsprechenden Nummer als Eigentuemer bzw. als Gruppe.
Der Systemverwalter kann mit 'chown' bzw. 'chgrp' dieser Datei den gewuenschten Eigentuemer und die gewuenschte Gruppe zuordnen. Das gleiche gilt, falls die angegebene Datei ein Dateiverzeichnis ist.

Die Zugriffsrechte werden so gesetzt, wie sie mit 'umask' definiert sind.

Existiert die angegebene Datei bereits in dem Dateiverzeichnis, in das sie kopiert werden soll, so werden die Zugriffsrechte nicht geaendert. Das s-Bit wird nur beruecksichtigt, wenn 'tar' vom Systemverwalter aufgerufen wird. Modifikationszeit, Eigentuemer und Gruppe bleiben unveraendert.

Befinden sich Dateien gleichen Namens mehrfach im Archiv (siehe Format 1) so ueberschreibt die nachfolgend vom Archiv eingelesene Datei eine bereits vorher kopierte Datei gleichen Namens. In diesem Fall wird die Modifikationszeit nicht beruecksichtigt.

attribut

Das Kommando 'tar x' kann durch Angabe von Funktionsattributen gesteuert werden. Die Attribute werden ohne Leerzeichen an die Funktion 'x' angefuegt.

v
'tar' gibt zu jeder Datei bzw. Dateiverzeichnis, das eingelesen wird, aus:

x pfadname, k bytes, n tape blocks

b
Dabei bedeutet:

x	(x - extract) die angegebene Datei wurde in das entsprechende Dateiverzeichnis eingetragen.
k byte	Die Datei ist 'k byte' gross.
n tape blocks	Die Datei belegt 'n' Bloecke. Die Blockgroesse ist abhaengig vom verwendeten Medium.

'v' nicht angegeben:
Das Kommando 'tar' gibt keine Meldungen aus.

w
'tar' erwartet fuer jede Datei eine Bestaetigung, bevor die Aktion ausgefuehrt wird. 'tar xw' gibt aus:

x dateiname:

Dabei bedeutet:

x	extract
dateiname	Name der Datei, die eingelesen werden soll.

'tar' erwartet nach dem Doppelpunkt : Ihre Eingabe.
Nur wenn Sie bejahen, fuehrt 'tar' die Aktion aus, sonst nicht.
In welcher Sprache Sie Ihre Antwort eingeben koennen, haengt vom Wert der NLS-Variablen 'LANG' ab. Wenn der Wert von 'LANG' mit 'En' fuer Englisch beginnt, bejahen Sie mit 'y' und verneinen mit 'n', wenn er mit 'De' fuer Deutsch beginnt, bejahen Sie mit 'j' und verneinen mit 'n' (siehe 'Internationale Umgebung').

'w' nicht angegeben:
'tar' fuehrt die Aktion ohne Bestaetigung aus.

m
'tar' setzt beim Kopieren der angegebenen Datei aus dem Archiv die Zeit der letzten Aenderung fuer die Kopie auf das aktuelle Datum mit Uhrzeit.

'm' nicht angegeben:
Die im Archiv gespeicherte Zeit der letzten Aenderung bleibt unveraendert.

o
   Die aus dem Archiv kopierte Datei erhaelt als Eigentuemer
   den Benutzer, der 'tar' aufgerufen hat. Als Gruppe wird der Datei
   ebenfalls die Gruppe des Benutzers zugewiesen, der 'tar'
   aufgerufen hat.

   'o' nicht angegeben:
   Die im Archiv gespeicherten Nummern fuer Eigentuemer und
   Gruppe werden uebernommen.

f geraetedatei
   'tar' erwartet die Eingabe einer Geraetedatei, von der gelesen
   werden soll, wenn sie nicht die Standard-Geraetedatei benutzen
   wollen (siehe Format 1).

datei
   Name der Datei, die eingelesen werden soll, wenn sie im Archiv
   steht. Sie koennen auch mehrere Dateien angeben, getrennt durch
   Leerzeichen.

   Ist die angegebene Datei ein Dateiverzeichnis, dann liest 'tar'
   dieses Dateiverzeichnis mit allen darin enthaltenen Dateien und
   Unterdateiverzeichnissen ein.

   Wenn der Dateiname Sonderzeichen der Shell enthaelt, muessen Sie
   diesen in Hochkommata '...' einschliessen. Geben Sie die Hochkomma-
   ta nicht an, so versucht die Shell, die angegebenen Sonderzeichen
   entsprechend der Dateinamen im aktuellen Dateiverzeichnis und nicht
   bezueglich der im Archiv vorhandenen Dateien zu ersetzen.

   'datei' nicht angegeben:
   'tar' liest das gesamte Archiv ein.

FEHLERMELDUNGEN

cannot open /dev/mt0

Sie haben keine Geraetedatei angegeben und die Standard-Geraetedatei
'/dev/mt0' steht nicht zur Verfuegung.

tape write error

Sie haben den Datentraeger nicht im richtigen Laufwerk oder die
falsche Geraetedatei angegeben.
Tritt diese Fehlermeldung erst auf, wenn 'tar' bereits angefangen
hat zu arbeiten, ist es moeglich, dass die angegebenen Dateien nicht
auf den Datentraeger passen. Die letzte Datei ist dann nicht
ordnungsgemaess gesichert.

directory checksum error

Der Datentraeger ist leer oder eine Datei ist nicht ordnungsgemaess

gesichert.

DATEI

'/usr/lib/nls/intlinfo'
   Dateiverzeichnis mit den Datenbasen, deren Dateiname den
   NLS-Variablen als Wert zugewiesen werden kann (siehe 3
  'Internationale' 'Umgebung - NLS').

UMGEBUNGSVARIABLEN6>

LANG
   beeinflusst den Zeichensatz und die Sprache der Meldungstexte
   bzw. Ihrer Antworten.
   Ist die Umgebungsvariable 'LC_TIME'
   leer oder nicht definiert, dann
   beeinflusst 'LANG' zusaetzlich
   das Format von Datums- und Zeitangaben.

LC_TIME
   beeinflusst das Format von Datums- und Zeitangaben.

INTERNATIONALE UMGEBUNG

Wenn Sie nicht in einer englischen, sondern in einer anderssprachigen Umgebung arbeiten, dann gibt 'tar' die Meldungstexte in dieser Sprache aus, und Sie geben Ihre Antworten in dieser Sprache ein. Die Sprache wird durch die NLS-Umgebungsvariable 'LANG' definiert.

Die Umgebungsvariable 'LC_TIME' beeinflusst das Format, in dem das Datum und die Uhrzeit ausgegeben werden, wenn Sie den Inhalt eines Archivs mit dem Funktionsattribut 'v' auflisten.

Ist die Variable 'LC_TIME' nicht definiert oder ist ihr die leere Zeichenkette zugewiesen, dann gilt fuer diese Variable als Standardwert der Wert von 'LANG'.

Hat eine der Variablen 'LANG, LC_CTYPE, LC_TIME, LC_COLLATE,' 'LC_NUMERIC' und 'LC_MONETARY' einen ungueltigen Wert oder ist die Variable 'LANG' nicht definiert bzw. leer, dann verhaelt sich 'tar', als waere das System nicht internationalisiert.

'tar' ist 8-bit-transparent.

Fuer die Datenuebertragung zwischen X/Open-Systemen ist der Zeichensatz ISO 8859-1 vereinbart (siehe A.4 'Moegliche Zeichensaetze'). 8-bit-Daten und 8-bit-Dateinamen sind jedoch unter Umstaenden nicht portabel zu nicht-internationalisierten Systemen. In diesen Faellen wird empfohlen, fuer die Datenuebertragung zwischen verschiedenen Rechnern nur Zeichen aus dem 7-bit-ASCII-Zeichenbereich und fuer Dateinamen nur Zeichen aus dem fuer portable Dateinamen vereinbarten Zeichensatz (Portable Filename Character Set) zu verwenden.

Weitere Informationen zur internationalen Umgebung finden Sie im Kapitel 3 'Internationale Umgebung - NLS' und im Handbuch Internationalisation in SINIX [3].

ABWEICHUNGEN ZU SINIX V2.X

Das 'tar'-Kommando im sie-Universum kann Dateien, die nicht auf einen Datentraeger passen, auf mehrere Datentraeger aufteilen und auch wieder einlesen. Das 'tar'-Kommando des X/Open-Universums kann das nicht. Wenn Sie einen mit 'sie tar' beschriebenen Datentraeger mit 'xopen tar' einlesen, kann das zu einer Fehlermeldung fuehren.

BEISPIELE

Archivieren auf Diskette

Alle im aktuellen Dateiverzeichnis und seinen Unterverzeichnissen enthaltenen Dateien sollen auf eine Diskette geschrieben werden. Es soll ausgegeben werden, welche Dateien archiviert werden. Ein eventuell vorhandenes Archiv soll ueberschrieben werden.

```
$ tar cvf /dev/fl2 *
a  hans/briefe/privat/tante  9 blocks
a  hans/briefe/privat/onkel  6 blocks
a  hans/briefe/dienst/chef1  3 blocks
a  hans/buch/manuscript  158 blocks
a  hans/buch/fehler  16 blocks
a  hans/tools/programm  20 blocks
    ...
```

Archivieren auf Magnetbandkassette

Das Dateiverzeichnis 'dokumente' soll auf Magnetbandkassette als zweites Archiv mit Blockungsfaktor 10 archiviert werden. Das Band wurde bereits positioniert. Nach dem Zugriff soll das Band nicht zurueckgespult werden. Deshalb wird die Geraetedatei '/dev/rts8' verwendet.

```
$ tar cbf 10 /dev/rts8 dokumente
```

Archivieren auf Magnetband

Die Datei 'brief' soll auf das Magnetband mit Blockungsfaktor 20, Schreibdichte 1600 cpi archiviert werden. Es soll danach zurueckgespult werden.

```
$ tar cfb /dev/rmt0 20 brief
```

Inhalt aller Archive eines Bandes lesen

Es soll der Inhalt eines ganzen Bandes gelesen und in ausfuehrlicher Form ausgegeben werden. Auf dem Band befinden sich drei Archive. Deshalb wird die Geraetedatei ohne Zurueckspulen benutzt und das 'tar'-Kommando mehrfach eingegeben.

```
$ tar tvf /dev/nrmt4; tar tvf /dev/nrmt4; tar tvf /dev/nrt4
tar: blocksize = 20
rw------- 33/1   1586   Sep 19  13:40 1988 hans/briefe/privat/tante
rw------- 33/1   2024   Sep 19  15:23 1988 hans/briefe/privat/onkel
rw-rw---- 33/1   1365   Oct 20  08:12 1988 hans/briefe/dienst/chef1
   ...
tar: blocksize = 20
rw-r--r-- 45/3   2345   Jan 18  13:20 1989 otto/test
rwxr-xr-x 45/3    800   Jan 27  12:50 1989 otto/programm
  ...
tar: blocksize = 20
rw------- 40/3   4567   Apr 10  07:58 1988 fritz/texte/kap1
rw------- 40/3   2367   Apr 10  08:50 1988 fritz/texte/kap2
  ...
```

Dateien von Archiv in aktuelles Dateiverzeichnis einlesen

Der Benutzer 'hans' mit UID 108 und GID 10 liest das Dateiverzeichnis 'fritz/texte' von einem Archiv auf Diskette in sein aktuelles Dateiverzeichnis ein. Die Dateien im Archiv haben als Eigentuemer die UID 40 und als Gruppe die GID 3. Der Benutzer 'hans' moechte, dass er bei den eingelesenen Dateien als Eigentuemer gesetzt wird.

```
$ tar xfo /dev/fl2 fritz/texte
```

b Mit dem Kommando 'ls -l' koennen die Eintraege fuer Eigentuemer und Gruppe ueberprueft werden.

```
$ ls -l
...
./fritz:
drwx-------2 hans    other    236   Apr  2   8:15 texte

./fritz/texte:
-rw------- 1 hans    other   4567   Apr 10  07:58 kap1
-rw------- 1 hans    other   2367   Apr 10  08:50 kap2
```

Verwendung von tar in einer Pipe

Das Dateiverzeichnis /usr1/hans soll nach /usr/hans verschoben werden.

```
$ (cd /usr1/hans; tar cf - .) | (cd /usr/hans; tar xf -)
```

b SIEHE AUCH

'flformat, mt'

A.3 'Geraetedateien fuer Datentraeger'

Handbuch Internationalisation in SINIX [3]

****

TR   ZEICHEN ERSETZEN ODER LOESCHEN (TRANSLITERATE)

'tr' liest einen Eingabetext von der Standard-Eingabe, ersetzt (Format 1) oder loescht (Format 2) einzelne Zeichen und schreibt das Ergebnis auf die Standard-Ausgabe.

---

tr[-c][-s][zeichenkette1[zeichenkette2]] tr -d[-c][-s][zeichenkette1[zeichenkette2]]

Wenn Sie bei einem Aufruf mehrere Optionen angeben, dann muessen Sie diese in beiden Formaten mit einem fuehrenden Bindestrich und ohne Leerzeichen aneinanderreihen, z.B. -cs oder -dc.

Format 1: Zeichen ersetzen

tr[-c][-s][zeichenkette1[zeichenkette2]]

'tr' ersetzt im Eingabetext jedes Zeichen, das in 'zeichenkette1' vorkommt, durch das entsprechende Zeichen in 'zeichenkette2': Das i-te Zeichen in 'zeichenkette1' wird im Eingabetext ersetzt durch das i-te Zeichen in 'zeichenkette2'. Ist 'zeichenkette2' kuerzer als 'zeichenkette1', dann werden die Zeichen aus 'zeichenkette1', zu denen es kein entsprechendes Zeichen in 'zeichenkette2' gibt, nicht ersetzt (siehe 'Beispiel 1').

-c
   (c - complement) 'zeichenkette1' wird bezueglich des aktuell gueltigen Zeichensatzes komplementiert. Die komplementierte 'zeichenkette1' enthaelt dann alle Zeichen des aktuell gueltigen Zeichensatzes ausser den in der urspruenglichen 'zeichenkette1' angegebenen Zeichen.
   Anschliessend ersetzt 'tr' im Eingabetext das i-te Zeichen in der komplementierten 'zeichenkette1' durch das i-te Zeichen in 'zeichenkette2'.

-s
   (s - squeeze)
   Nach der Ersetzung verkuerzt 'tr' jede Folge von gleichen Zeichen, die in 'zeichenkette2' vorkommen, zu einem Zeichen (siehe 'Beispiel 4').

zeichenkette1 [zeichenkette2]
In 'zeichenkette1' geben Sie die Zeichen an, die ersetzt werden sollen. 'zeichenkette2' ist die Ersetzungszeichenkette.

In beiden Zeichenketten muessen Sie die Zeichen ohne Leer- oder sonstige Trennzeichen aneinanderreihen.

Enthaelt eine Zeichenkette Zeichen, die fuer die Shell eine besondere Bedeutung haben, muessen Sie die Zeichenkette in Hochkommata '...' einschliessen oder die Zeichen mit einem vorangestellten Gegenschraegstrich \ entwerten.

Sie koennen jedes Zeichen folgendermassen oktal angeben: \\oktalzahl oder '\oktalzahl' oder "\oktalzahl", wobei 'oktalzahl' eine ein-, zwei- oder dreistellige Oktalzahl ist. Den Gegenschraegstrich muessen Sie, wie angegeben, entwerten, damit die Ziffern als Oktalzahl erkannt werden.

'tr' bearbeitet das Zeichen NUL (oktal 000) in den angegebenen Zeichenketten nicht; kommt das Zeichen NUL als Eingabezeichen vor, wird es stets geloescht.

Zeichenbereiche oder Folgen von gleichen Zeichen koennen Sie in beiden Zeichenketten wie folgt abkuerzen; die eckigen Klammern [und] muessen Sie hier explizit angeben!

[a-z]
steht fuer die Folge der Zeichen von 'a' bis 'z' einschliesslich. Die Zeichen sind gemaess der aktuell gueltigen
@ Sortierreihenfolge geordnet (die in der Regel NICHT mit der
@ ASCII-Tabelle uebereinstimmt). Im Unterschied zu internationalisierten regulaeren Ausdruecken duerfen 'a' und 'z' nur einfache Zeichen, d.h. keine Ausdruecke fuer Aequivalenzklassen '[=c=]' oder Zeicheneinheits-Symbole '[.cc.]' sein (siehe A.1 'Regulaere Ausdruecke'). Die Sortierreihenfolge wird in einer internationalisierten Umgebung durch die Umgebungsvariablen 'LC_COLLATE' bzw. 'LANG' definiert (siehe 'Internationale Umgebung').
@ Wenn man Zeichenbereichsangaben verwenden will, um zum Beispiel
@ Grossbuchstaben in Kleinbuchstaben umzuwandeln,
@ (z.B. mit "tr '[A-Z]' '[a-z]' datei")
@ muessen in der vorgegebenen Sortierreihenfolge folgende
@ Kriterien erfuellt sein:
@ 1. Gleiche Anzahl Gross- und Kleinbuchstaben.
@ 2. Gleiche relative Sortierreihenfolge von Grossbuchstaben
@ einerseits und Kleinbuchstaben andererseits.
@ 3. Beide Sortierreihenfolgen sind getrennt und lueckenlos.
@ 4. Es gibt nur eins-zu-eins Umsetzungen.
@ Solche Bedingungen sind in den meisten Zeichensaetaen nicht
@ erfuellt (z.B. existiert das deutsche Eszett nur als
@ Kleinbuchstabe). Im Zweifelsfall ist es ratsam, beide

@ Zeichenketten explizit anzugeben.

[[:klasse:]]
steht fuer die Folge aller Zeichen, die in der Zeichenklasse 'klasse' enthalten sind. Die Zeichen sind gemaess der aktuell gueltigen Sortierreihenfolge geordnet. Die Zeichenklassen werden in einer internationalisierten Umgebung durch die Umgebungsvariablen 'LC_CTYPE' bzw. 'LANG' definiert, die Sortierreihenfolge wird durch die Umgebungsvariablen 'LC_COLLATE' bzw. 'LANG' festgelegt (siehe 'Internationale Umgebung').

'klasse' kann sein:

alpha	alle Buchstaben
upper	alle Grossbuchstaben
lower	alle Kleinbuchstaben
digit	alle Dezimalziffern ('0' bis '9')
xdigit	alle Hexadezimalziffern ('0' bis '9', 'a' bis 'f' und 'A' bis 'F')
alnum	alle alphanumerischen Zeichen (Buchstaben und Ziffern)
space	alle Zeichen, die bei der Textdarstellung Zwischenraum produzieren (z.B. Leer- oder Tabulatorzeichen)
punct	alle Trennzeichen
print	alle abdruckbaren Zeichen (einschliesslich der Zeichen in 'space')
graph	alle sichtbaren abdruckbaren Zeichen (ohne die Zeichen in 'space')
cntrl	alle Steuerzeichen

Siehe auch A.1 'Regulaere Ausdruecke', 'Internationalisierte regulaere Ausdruecke' und 'Beispiel 2a'.

[[=c=]]
steht fuer die Aequivalenzklasse von 'c' - das sind alle Zeichen bzw. Zeicheneinheiten, die in der aktuell gueltigen Sortierreihenfolge die gleiche relative Ordnung wie 'c' haben. Die Sortierreihenfolge wird in einer internationalisierten Umgebung durch die Umgebungsvariablen 'LC_COLLATE' bzw. 'LANG' definiert (siehe 'Internationale Umgebung'). Siehe auch A.1 'Regulaere Ausdruecke', 'Internationalisierte regulaere Ausdruecke' und 'Beispiel 2b'.

[[.cc.]]
steht fuer die Zeicheneinheit 'cc'. Zeicheneinheiten, die aus mehreren Zeichen bestehen, muessen in dieser Form dargestellt werden, um sie von einfachen Zeichen zu unterscheiden. 'tr' betrachtet eine solche Zeicheneinheit als ein einziges Zeichen. 'cc' muss in der aktuell gueltigen Sortierreihenfolge als zulaessige Zeicheneinheit definiert sein. Die Sortierreihenfolge wird in einer internationalisierten Umgebung durch die Umgebungsvariablen 'LC_COLLATE' bzw. 'LANG'

definiert (siehe 'Internationale Umgebung').
Siehe auch A.1 'Regulaere Ausdruecke', 'Internationalisierte regulaere Ausdruecke' und 'Beispiel 2c'.

[a*n]
steht fuer 'n'-mal das Zeichen 'a'.
Z.B. [a*3] steht fuer aaa.
Ist die erste Ziffer von 'n' 0, so wird 'n' als Oktalzahl interpretiert, andernfalls als Dezimalzahl.
Wenn 'n' fehlt oder 0 ist, dann steht der Ausdruck fuer so viele Wiederholungen des Zeichens 'a', wie noetig sind, um 'zeichenkette2' auf die Laenge von 'zeichenkette1' aufzufuellen (siehe 'Beispiel 1').

'zeichenkette2' nicht angegeben:
Fuer 'zeichenkette2' wird die (evtl. komplementierte, siehe '-c') 'zeichenkette1' genommen.

'zeichenkette1' und 'zeichenkette2' nicht angegeben:
'zeichenkette1' ist die leere Zeichenkette.
Fuer 'zeichenkette2' wird entweder die leere Zeichenkette (ohne Option '-c') oder der gesamte, aktuell gueltige Zeichensatz (mit Option '-c') genommen.

Format 2: Zeichen loeschen

tr -d[-c][-s][zeichenkette1[zeichenkette2]]

Ein Aufruf in diesem Format ist nur dann sinnvoll,wenn 'zeichenkette1' angegeben ist. 'zeichenkette2' wird ignoriert, wenn Option '-s' nicht angegeben ist.

-d
(d - delete)
Alle Eingabezeichen, die in 'zeichenkette1' vorkommen, werden geloescht.
Wenn Option '-s' nicht angegeben ist, wird 'zeichenkette2' ignoriert.

-c
(c - complement) 'zeichenkette1' wird bezueglich des aktuell gueltigen Zeichensatzes komplementiert. Die komplementierte 'zeichenkette1' enthaelt dann alle Zeichen des aktuell gueltigen Zeichensatzes ausser den in der urspruenglichen 'zeichenkette1' angegebenen Zeichen.
Anschliessend loescht 'tr' alle Eingabezeichen, die in der komplementierten 'zeichenkette1' vorkommen.

-s
(s - squeeze)

'tr' verkuerzt jede Folge von gleichen Zeichen,
die in 'zeichenkette2' vorkommen, zu einem
Zeichen. Ist 'zeichenkette2' nicht
angegeben, ist die Option '-s' wirkungslos.

zeichenkette1 [zeichenkette2]
In 'zeichenkette1' geben Sie die Zeichen an,
die geloescht werden sollen. 'zeichenkette2'
enthaelt die Zeichen, die nur einmal ausgegeben werden
sollen, falls sie in der Ausgabe mehrfach hintereinander
auftreten (siehe Option '-s').

Im Abschnitt 'Format 1' ist beschrieben, wie Sie die
beiden Zeichenketten eingeben koennen.

'zeichenkette1' und 'zeichenkette2' nicht angegeben:
Ist Option '-c' nicht angegeben, werden die Eingabezeichen
unveraendert auf die Standard-Ausgabe kopiert. Ist '-c'
angegeben, wird nichts auf die Standard-Ausgabe ausgegeben,
da 'tr' saemtliche Eingabezeichen loescht.

DATEI

'/usr/lib/nls/intlinfo'
Dateiverzeichnis mit den Datenbasen, deren Dateiname den
NLS-Variablen als Wert zugewiesen werden kann (siehe 3
'Internationale' 'Umgebung - NLS').

UMGEBUNGSVARIABLEN

LANG
beeinflusst den Zeichensatz, die Sprache der Meldungstexte und
alles zusammen, was mit den folgenden Variablen einzeln beeinflusst
werden kann.

LC_CTYPE
beeinflusst die Zeichenklassifizierung (Ausdruck '[[:klasse:]]').

LC_COLLATE
beeinflusst die Sortierreihenfolge, d.h. legt fest, wie
Zeichenbereiche '[a-z]', Ausdruecke fuer Aequivalenzklassen
'[[=c=]]' und Zeicheneinheits-Symbole '[[.cc.]]' ausgewertet
und in welcher Reihenfolge die zu einer Zeichenklasse
(Ausdruck '[[:klasse:]]') oder Zeichenmenge gehoerigen Zeichen
angeordnet werden.

ABWEICHUNGEN ZU SINIX V2.X

In SINIX V2.x koennen Zeichenbereiche auch ohne eckige Klammern
angegeben werden, z.B. tr a-z A-Z <datei.

INTERNATIONALE UMGEBUNG

Wenn Sie nicht in einer englischen, sondern in einer anderssprachigen Umgebung arbeiten, dann gibt 'tr' die Meldungstexte in dieser Sprache aus.Die Sprache wird durch die NLS-Umgebungsvariable 'LANG' definiert.

Die NLS-Umgebungsvariable 'LC_COLLATE' legt fest, wie Zeichenbereiche '[a-z]', Ausdruecke fuer Aequivalenzklassen '[[=c=]]' und Zeicheneinheits-Symbole '[[.cc.]]' ausgewertet werden. 'LC_CTYPE' legt fest, welche Zeichen in den einzelnen Zeichenklassen enthalten sind (Ausdruck '[[:klasse:]]'), und 'LC_COLLATE' bestimmt die Reihenfolge, in der die zu 'klasse' gehoerigen Zeichen angeordnet werden. Bei Option '-c' legt 'LANG' fest, welche Zeichen in der aktuell gueltigen Zeichenmenge enthalten sind, und 'LC_COLLATE' bestimmt die Reihenfolge der Zeichen.

Ist eine der Variablen 'LC_CTYPE' und 'LC_COLLATE' nicht definiert oder ist ihr die leere Zeichenkette zugewiesen, dann gilt fuer diese Variable als Standardwert der Wert von 'LANG'.

Hat eine der Variablen 'LANG, LC_CTYPE, LC_TIME, LC_COLLATE,' 'LC_NUMERIC' und 'LC_MONETARY' einen ungueltigen Wert oder ist die Variable 'LANG' nicht definiert bzw. leer, dann verhaelt sich 'tr', als waere das System nicht internationalisiert.

In internationalisierten Umgebungen, in denen Zeicheneinheiten, die aus mehreren Zeichen bestehen, definiert sind, tritt folgender Nebeneffekt auf: Ist die Standard-Eingabe ein zeichenorientiertes Geraet (z.B. eine Datensichtstation) und geben Sie ein Zeichen ein, das moeglicherweise das erste Zeichen einer solchen Zeicheneinheit sein kann, dann wartet 'tr' auf weitere Zeichen (bis maximal 3 weitere Zeichen) oder das Dateiende-Zeichen, bevor es die Ausgabezeichen ausgibt.

'tr' ist 8-bit-transparent.

Weitere Informationen zur internationalen Umgebung finden Sie im Kapitel 3 'Internationale Umgebung - NLS' und im Handbuch Internationalisation in SINIX [3].

BEISPIELE

Zeichen ersetzen (Format 1)

1. 'tr' ohne Option - einfache Beispiele, die die Wirkungsweise von 'tr' zeigen

   ```
   $ cat tage
   Montag Dienstag Mittwoch Donnerstag Freitag Samstag Sonntag
   $ tr MD md <tage
   montag dienstag mittwoch donnerstag Freitag Samstag Sonntag
   ```

   'tr' ersetzt alle 'M' durch 'm' und alle 'D' durch 'd'.

   ```
   $ tr MDF md <tage
   montag dienstag mittwoch donnerstag Freitag Samstag Sonntag
   ```

Hier ist die zweite Zeichenkette kuerzer als die erste.
'tr' ersetzt alle 'M' durch 'm' und alle 'D'
durch 'd', das 'F' bleibt jedoch unveraendert.

Nun soll jeder Kleinbuchstabe durch 'x' ersetzt
werden. Dabei wird vorausgesetzt, dass in der aktuell
gueltigen Umgebung der Zeichenbereich 'a-z'
genau die Kleinbuchstaben enthaelt.
Der folgende Aufruf leistet das Gewuenschte 'nicht':

```
$ tr '[a-z]' x <tage
Montxg Dienstxg Mittwoch Donnerstxg Freitxg Sxmstxg Sonntxg
```

Hier ersetzt 'tr' nur das 'a' durch 'x'. Sollen
alle Kleinbuchstaben ersetzt werden, dann muss man 'tr'
folgendermassen aufrufen:

```
$ tr '[a-z]' '[x*]' <tage
Mxxxxx Dxxxxxxx Mxxxxxxx Dxxxxxxxxx Fxxxxxx Sxxxxxx Sxxxxxx
```

Hier gehoert zu jedem Zeichen in Zeichenkette1
ein 'x' in Zeichenkette2, da durch den
Stern * die Zeichenkette2 mit 'x' aufgefuellt
wird. Die Hochkommata sind notwendig, da die
Zeichenketten Shell-Sonderzeichen enthalten.

Vorsicht

Die Bedeutung des Intervalls '[a-z]'
ist abhaengig von der aktuell gueltigen
Sortierreihenfolge: In manchen internationalisierten
Umgebungen enthaelt der Bereich 'a-z'
auch Grossbuchstaben. Besser ist es deshalb, fuer den
Bereich der Kleinbuchstaben den Ausdruck '[[:lower:]]'
zu verwenden (siehe 'Beispiel 2a').

2. 'tr' ohne Option - weitere Beispiele

a) Zeichenbereiche und Zeichenklassen

Alle Kleinbuchstaben sollen in Grossbuchstaben
umgewandelt werden:

```
$ cat tage
Montag Dienstag Mittwoch Donnerstag Freitag Samstag Sonntag
$ tr '[[:lower:]]' '[[:upper:]]' <tage
MONTAG DIENSTAG MITTWOCH DONNERSTAG FREITAG SAMSTAG SONNTAG
```

Beachten Sie: Die Auswertung
der Zeichenklassen '[[:lower:]]' und '[[:upper:]]'
ist abhaengig von der aktuell gueltigen Zeichenklassifizierung
(Variable 'LC_CTYPE') und der aktuell gueltigen
Sortierreihenfolge (Variable 'LC_COLLATE').
Das Beispiel funktioniert also nur in

Umgebungen, in denen jedem Kleinbuchstaben genau ein
Grossbuchstabe und umgekehrt zugeordnet ist und
in denen Klein- und Grossbuchstaben in analoger
Reihenfolge sortiert sind.

b) Aequivalenzklassen

Ist die franzoesische Sortierreihenfolge
'LC_COLLATE=Fr_FR.88591'
gueltig, dann umfasst der Ausdruck '[[=e=]]' das Zeichen 'e'
sowie alle akzentuierten 'e'. Der folgende 'tr'-Aufruf
ersetzt deshalb jedes akzentuierte 'e' durch ein einfaches 'e':

```
 $ tr '[[=e=]]' '[e*]' <datei
```

Ist dagegen z.B. die deutsche Sortierreihenfolge
'LC_COLLATE=De_DE.88591' gueltig, leistet der obige
'tr'-Aufruf das Gewuenschte nicht, da '[[=e=]]' in einer
solchen Umgebung nur
fuer das Zeichen 'e' steht. Siehe 'Umgebungsvariablen'.

c) Zeicheneinheits-Symbole

In der spanischen Umgebung
'LANG=LC_COLLATE=Es_SP.88591'
ist 'ch' als zulaessige Zeicheneinheit definiert.
Der folgende 'tr'-Aufruf wandelt jedes 'ch'
in Grossschreibung 'CH' um. Die einfachen Zeichen
'c' und 'h' werden unveraendert
ausgegeben, wenn sie nicht zu der Zeichenkette
'ch' gehoeren.

```
 $ tr '[[.ch.]]' '[[.CH.]]' <datei
```

3. 'tr' mit Option '-c'

Alle "Nicht-Buchstaben" sollen durch einen Doppelpunkt
ersetzt werden:

```
$ cat text
"Hoffentlich ist morgen schoenes Wetter", sagte er.
$ tr -c '[[:alpha:]]' '[:*]' <text
:Hoffentlich:ist:morgen:schoenes:Wetter:::sagte:er::
```

4. 'tr' mit Option '-s'

Es soll eine Liste aller Woerter, die in der Datei
'text' (siehe 'Beispiel 3') vorkommen, erstellt werden,
und zwar so, dass jedes Wort auf einer eigenen
Zeile steht. Ein Wort sei eine
maximale Zeichenkette, die nur aus Buchstaben besteht.

```
$ tr -cs '[[:alpha:]]' '[\012*]' <text

Hoffentlich
ist
morgen
schoenes
Wetter
sagte
er

Hier ersetzt 'tr' zunaechst alle "Nicht-Buchstaben"
durch je ein Neue-Zeile-Zeichen (Option '-c',
siehe 'Beispiel 3'). Anschliessend verkuerzt 'tr'
jede Folge von Neue-Zeile-Zeichen zu einem einzigen
Neue-Zeile-Zeichen (Option '-s').
```

Zeichen loeschen (Format 2)

5. Nichtdruckbares Zeichen aus einer Datei loeschen ('tr -d'):

```
$ tr -d '\016' <datei

'tr' loescht aus der Datei das Zeichen, dessen Code oktal
den Wert 016 hat, und gibt das Ergebnis auf die
Standard-Ausgabe aus. Welches Zeichen das ist, haengt vom aktuell
gueltigen Zeichensatz ab (siehe NLS-Umgebungsvariable 'LANG');
normalerweise ist es das Zeichen <CTRL> N.
```

6. Alle Zeichen ausser Ziffern loeschen ('tr -cd'):

```
$ cat datei
Addition: 1+1=?
Multiplikation: 9*9=...

$ tr -cd '[[:digit:]]' <datei
1199
```

SIEHE AUCH

'sed'

Handbuch Internationalisation in SINIX [3]