The
UNIX™ System
Quick Guide

Ⓐ **Bell Labs**   *June 1982*

This card is an Introduction and a Quick Guide to the UNIX* system. It explains how to log in ... how to use basic Shell commands ... and how to create, edit, and use files with the ed editor. Several tables explain Shell and ed special characters. Numerous examples illustrate the commands and special characters.

*Trademark of Bell Laboratories

M. I. Bolsky
Systems Training Department (45272)

Printed in U.S.A.

TO THE USER

- **Shell** — The "Shell" is the overall command structure for the *UNIX* system. This card summarizes frequently used options, of frequently used Shell commands.

- **ed** — The *UNIX* system has several text editors. *ed* is believed to be the one that is most widely used. This card explains all of the *ed* commands. With *ed*, you can — Create files containing any kind of alphanumeric text ... Add or delete text ... Locate any word or string of characters ... Move lines of text around within a document or program ... Duplicate lines of text ... Substitute words or strings of characters, for other words or strings of characters.

- **Different Versions** — There may be differences in the *UNIX* systems at different locations. Check with your local *UNIX* System Administrator, or other designated person. (This card describes the Bell Laboratories *UNIX* System Release 4.0.)

- **Bell Labs Systems Training Department (45272).** Courses on the *UNIX* system are offered, for Bell Labs people, at Piscataway, NJ and at other Bell Labs locations. For information, or to enroll, write or phone — Registrar, Bell Labs PY 3B-110B, 6 Corporate Place, Piscataway, NJ 08854. Phone PY x7940 *or* (Bell System only) 8+246-7940 *or* (201) 981-7940. For documents on the *UNIX* system, consult your Computing Information Service Library. *Note* — Others in the Bell System, and those outside the Bell System, should consult their System Administrator about courses and documents.

- **Comments** — Your comments on this card are welcomed. Please write to M. I. Bolsky, Bell Labs PY 3G-150, 6 Corporate Place, Piscataway, NJ 08854. Phone x6426 *or* (Bell System only) 8+246-6426 *or* (201) 981-6426.

Grateful acknowledgement is due to Joan Lorincz for her skillful typesetting of this card.

NOTATION

Note — The following notation is used *on this card*. It is *not* for use in *UNIX* system commands.

F1,F2 D1,D2	• Sample file (*F1,F2*) and directory (*D1,D2*) names used in examples on this card.
Prints	• When the word "Prints" is used, it means the standard output — a printout at your terminal or a printer ... a display on your terminal ... or output to a file.
...	• One or more of the preceding elements may be used, each separated by a space. *F1* ... means you can specify either one or several file names. *Note* — 3 dots (ellipsis) should be distinguished from *1 dot* (pathname of current directory), and *2 dots* (pathname of parent directory).
[ ]	• Optional element.
{ }	• Use only *one* of the listed elements.
Examples	• What *you* type, is shown in the examples on this card in **bold italic** typeface. What the *UNIX* system prints, is shown in regular typeface.

SHELL

ACCESS

WHAT YOU NEED

- **Phone Number** of *UNIX* system. (If you are using a dial-up terminal.)

- **login**

- **password**

- **Terminal**

  - **Characteristics** — ASCII (128 characters), with an acoustic coupler or a data set (modem).
  - **Setting** —
    - Speed (30 cps, 300 baud, high speed). **Note** — If you have a choice, 1200 baud or higher cuts down on transmission time (but does not necessarily improve response time).
    - Full duplex.
    - Lowercase.
    - Parity (even or none).
    - On line (Remote).
    - Power ON.

TYPING CONVENTIONS

- **Space(s)** — The command name and each element must be separated by one or more spaces or tabs.

- **RETURN** — Press the **RETURN** key after you type your login ... your password ... and after each response to a system prompt, each command, and each line of input text.

- **CNTL/d, CNTL/i, CNTL/s** — Press the Control key (labeled **CONTROL** or **CNTL**) and, at the same time, the character *d* or *i* or *s*, respectively, as indicated.

- **# is the default erase character**.

  - Deletes the previous character.
  - May be repeated to delete multiple characters.
  - Must be input before you press **RETURN**.
  - **Example** — $ *day#te*
    Wed Feb 06 09:11 EST

- **@ is the default line delete**.

  - Deletes entire line.
  - Automatically generates a new line (but *not* a new $ prompt).
  - Must be input before you press **RETURN**.
  - **Example** — $ *stty -tub@*
    stty -tabs

- **To enter # or @** (or any other *special* character), as a *literal*, precede with *reverse slash* \.
  Example —
  Enter (i.e., type) — *Item \#5*
  Is printed out as — *Item #5*

- **To erase a reverse slash (\) itself,** type *two #s* — *\##*.

- **To redefine these characters** — The # and @ can be redefined; e.g., on a display terminal, people often make the **BACKSPACE** key the erase character. This is done via the **stty** command. (See page 8.)

- To **TEMPORARILY** stop Shell or *ed* output — Press **CNTL/s**. This is useful on display terminals to prevent output from disappearing before you can read it. Press **any other character** to resume output.

- To **TERMINATE** Shell or *ed* output before a command has finished execution — Press the **DEL** or **BREAK** or **RUB** key.

LOGGING ON

- **Turn on the terminal.**

- **Dial UNIX** system number (on dataset press **TALK** or **TK** before dialing). You will receive one of the following —

  - Recorded message.
  - Busy signal — Means recorded message is busy. Dial again.
  - No answer (continuous ringing) — Line trouble, or system is down.
  - High-pitched tone.

- **Upon receiving high-pitched tone** —

  - **For dataset** — Press **DATA** or **DM**, and hang up.
  - **For acoustic coupler** — Place handset into coupler according to diagram on coupler.

- **When connection is completed** the **UNIX** system will ask for your login and then your password —

  - login: Type your login.
  - Password: Type password (it will *not* be printed).
  - **Note 1** — If "login" does not print, check terminal and dataset settings. If these are correct, there may be a hardware problem. Dialing a different system (if available) may help.
  - **Note 2** — If your login directory contains a file named *.profile*, the commands in that file are executed every time you log on. Commonly *.profile* specifies (via the Shell variable, TERM) what terminals you may be using; consult your System Administrator to find out what is appropriate. You can use a **cat .profile** command to see what is in it, and you can modify *.profile* with **ed** (or some other editor).

    - **MAIL=/usr/mail/login** — Tells you when new mail comes in. (*login* is your login name)

- **If login sequence is incorrect** —

  - Login incorrect
  - login: Type login again.
  - Password: Type password again.

LOGGING OFF

- If you or someone else wants to log in immediately under some other login name —

  - Press **CNTL/d**.
  - login:
  - Password:

- Otherwise —

  - Hang up the phone and/or turn off the terminal. **CAUTION** — On some systems, your terminal would still be on-line and thus your account would be charged for the time! On such systems, **before** you hang up, you must press **CTRL/d** to log off.

FILES

- Created via *ed*.

- Purpose is to store — Data ... Memos ... Program source code ... Executable code.

- Maximum size is 1 billion characters — This is the physical limit. System Administrators usually impose much lower limits.

- Identified by a *filename* —

  - Maximum 14 characters. **Note** — The **UNIX** system distinguishes between uppercase and lowercase. Thus filename *A* is a different filename than *a*.
  - Do **not begin** a filename with plus (+) or minus (−) sign.
  - Avoid as part of a filename (these characters are troublesome) —
    * / [ ] ( ) ; & ? $ ˆ | \ # @ > < ' ' "
  - For your convenience in remembering it, a filename should describe the contents of the file. At the same time, for your convenience in typing it, it should be brief.

- **The Current Directory** (. — *referred to as "dot"*). The file system is arranged in a hierarchy of directories. When the System Administrator gave you a user name, he or she also created a directory for you (ordinarily with the same name as your user name, and known as your *login* or *home* directory). When you log in, that directory becomes your *current* or *working* directory, and any file name you type is by default assumed to be in that directory. *Example* —
  - **cp /tmp/F1 /tmp/F2 .** — Copies files F1 and F2 from directory *tmp* into the current directory.

- **Path Names.** To refer to files not in the current directory, you must use a path name. Full path names begin with /, which is the name of the *root* directory of the whole file system. After the slash comes the name of each directory containing the next sub-directory (followed by a /), until finally the file name is reached.
  Example — **/usr/ac/filex** refers to the file *filex* in directory *ac*, while *ac* is itself a subdirectory of *usr*; *usr* springs directly from the root directory. In this example, *usr* is the parent directory of *ac*. If you are in directory *ac*, you can refer to *usr* as /usr or as ... .
  Note — If your current directory contains subdirectories, the path names of files therein begin with the name of the corresponding subdirectory (*without* a prefixed /). Usually, a path name may be used anywhere a file name is required.

SPECIAL CHARACTERS

$ Default Shell prompt

| Pipes output of one command into another command. **Example** —
  - $ **who** | **sort** — Output from **who** command is sent to **sort** command. Thus you get a sorted listing.

<   Redirects the input of a command, from the standard input to a file. **Example** —
  - $ **mail abc** < **F2** — Sends message in file F2 to login *abc*.

> Redirects the output, from the standard output to a file. Overwrites (if it already exists), or creates (if it doesn't) F3. *Example —*
- **$ cat F1 F2 > F3** — Concatenates F1 and F2 and places result in F3.

>> Redirects the output from a command, from the standard output to a file, and appends it to the end of that file. Creates F2 if it doesn't already exist. *Example —*
- **$ cat F1 >> F2** — Appends F1 to F2. Creates F2 if it doesn't already exist.

. Path name of current directory

.. Path name of parent directory

/ Path name of root directory

EXPANSION CHARACTERS

Note 1 — These expansion characters apply to filenames.

Note 2 — In the examples below, the files shown (chap2,5,9,10,11,20,21 and para1,2,3,4,5, etc.) would, of course, be printed only if those files exist in the current directory.

[] Delimits a set of characters, any one of which can be considered a match for the character position identified by the brackets. *Examples —*
- **$ pr chap[259]** — Prints chap2, chap5, chap9.
- **$ pr chap[12][01]** — Prints chap10, chap11, chap20, chap21.

– Used in brackets to indicate a range of characters. *Examples —*
- **$ cat para[1-3]** — Concatenates and prints para1, para2, para3.
- **$ cat para[!1-3]** — Concatenates and prints para4, para5, etc.

? Represents any single character. *Example —*
- **$ cat para?** — Concatenates and prints all files that have 5-character names, beginning with *para* and ending with any single ASCII character. *Note* — See the *Note* for * immediately below.

* Represents zero or more occurrences of any and all characters except /. *Note* — A dot (.) at the beginning of a filename must be explicitly matched. Thus .* will match filenames beginning with dot, while * will match all filenames *not* beginning with dot. *Example —*
- **$ cat para*** — Concatenates and prints all files whose names begin with *para*, regardless of how long the names are or what characters are in the names.

COMMANDS

Note — You can type two or more commands on a single line by separating them with semicolons(;) or ampersands (&). *Example: $ cp F1 F2; cat F2*
Examples —

$ banner HI • Prints word(s) in large letters.

$ cat F1 ... • Concatenates/prints file(s).

$ cd • Returns to login directory.
$ cd D1 • Changes to directory D1.
$ cd .. • Changes to immediate parent directory (.. is pathname of parent directory).

$ cp F1 F2 • Copies file F1; names it F2.
$ cp F1 ... D1 • Copies file(s) to directory D1.
$ cp F1 ... .. • Copies file(s) to parent directory (.. is pathname of parent directory).

$ crypt [key] • Encodes/decodes text. If **key** not given, **crypt** demands a key from terminal.

$ date • Prints current time/date.

$ echo HI • Prints *HI*. Ends in Newline.
$ echo -n HI • Does not end in Newline.

$ file F1 • Prints the type of file F1.

$ grep "abc" F1 ...
• Prints from file(s), lines with string "abc". *-v* = All lines *except* matching lines. *–c* = Count of matching lines. *–n* = Matching lines with line numbers.

$ help [name ...]
• Explains message number(s) and/or command name(s).

$ ls • Lists files in current directory.
$ ls -l • Shows permissions, etc.
$ ls D1 • Lists files in directory.
$ ls F1 ... • Checks existence of file(s).

$ mail • Prints contents of your mailbox, one message at a time. *Press RETURN* = Go to next message. *d* = Delete last message. *p* = Print message again. *–* = Go back to previous message. *s* = Save message in *mbox* file.

$ mail abc • Sends mail to login *abc*. Type message on line(s) following command. End with dot(.) on line by itself.

$ man titles • Prints named *UNIX* system manual pages.

$ mesg • Reports current *write* command permission state of this terminal.
$ mesg n • Forbids messages via *write*.
$ mesg y • Reinstates *write* permission.

$ mkdir D1 ... • Creates directory(ies) in your current directory.

$ mv F1 F2 • Renames file F1 to F2.
$ mv D1 D2 • Renames directory D1 to D2.
$ mv F1 ... D1 • Moves file(s) to directory D1.

$ *news title*	• Prints news items. Press *DEL* to stop item; press *DEL* again within 1 second to exit *news*. *-a* = All items. *-n* = Names of current items. *-s* = Number of current items.
$ *passwd*	• Changes your password. The system asks for your old, and then your new password.
$ *pr F1 ...*	• Paginates/prints file(s).
$ *pwd*	• Prints path name of current directory.
$ *rm F1 ...*	• Removes file(s).
$ *rmdir D1 ...*	• Removes empty directory(ies).
$ *sort*	• Sorts lines from standard input in ASCII sequence, and outputs.
$ *sort F1 ...*	• Sorts lines in file(s).
$ *spell F1 ...*	• Prints misspelled/unusual words.
$ *stty*	• Prints your current terminal I/O settings.
$ *stty erase '-h'*	• Makes *BACKSPACE* key (instead of #) the character erase key.
$ *stty -tabs*	• Sets simulated tab stops to default tab settings, if *no TAB* key. *CNTL/i* serves as *TAB* key.
$ *tabs*	• Sets tabs to default (if there *is TAB* key). (Default = tab every 8 spaces.)
$ *teach*	• Accesses the *CAIAC* system for computer assisted instruction. You will be presented with a course menu.
$ *tty*	• Prints path name of your terminal device.
$ *wc F1 ...*	• Prints line/word/character counts for file(s). *-l*, *-w,* or *-c* option counts only lines, words, or characters, respectively.
$ *who*	• Prints information about on-line terminals.
$ *who am i*	• Prints information about your terminal only.
$ *write user [id]*	
	• Writes to *user* who is currently logged in. Press *CTRL/d* to return to $ prompt.
$ *wwb F1 ...*	• Runs Writer's Workbench System on file(s). *-s* = Short version.
$ *wwbaid*	• On-Line User's Aid.
$ *wwbhelp*	• Prints information on commands/topics.
$ *wwbinfo*	• Prints table of commands/functions.

ed

ACCESS

• *Introduction*
- • *Buffer* — *ed* operates on a copy of the file it is editing; changes made to the copy have no effect on the file until a *write* command is given. The copy of the text being edited resides in a temporary file called the *buffer*. There is only *one* buffer. *Note* — The *write* and some other *ed* commands have similar names as Shell commands, but they are *different*.
- • *Size limitations* — 512 characters per line, 256 characters per global command list, 64 characters per file name, and 128K characters in the buffer. The limit on the number of lines depends on the amount of user memory: each line takes 1 word.

• You can access *ed*
- • *Without* a filename — $ *ed*
  Note — The first name used in an *edit, read,* or *write* command becomes the remembered filename.
- • With an *existing* filename — $ *ed F1*
- • With a *new* filename — $ *ed F1*

• *ed* options in addition to *filename*
         ed [−] [−x] [filename]
- • − Suppresses information messages, and printing of character counts by the *edit*, *read* and *write* commands, and of diagnostics by *edit* and *quit*, and of *!* prompt after a !shell command.
- • −x causes a *crypt* command to be simulated first to handle an encrypted file.

• *PROMPT (P)* — *ed*'s default mode does *not* prompt you (unlike the Shell which *does* prompt you with $). To turn on *ed*'s prompt (*), type *P*. Subsequent *P* commands turn the prompt alternately off and on. *Note* — Even with the prompt on, it is *not* printed in the input mode of *append, change,* and *insert*.

• *HELP (h,H)* — If *ed* doesn't understand your command, it prints ? for command errors, or *?file* for inaccessible file. If you want explanation of —
- • Only the last *?* — Type *h*.
- • The last *?*, plus all subsequent *?s* — Type *H*. Subsequent *H* commands turn this mode alternately off and on.

• *To escape ed, execute a Shell command, and return to ed* — Type *!* followed by Shell command. *Example* — !banner HELLO. *Note* — Type *!!* to repeat your previous Shell command.

SAMPLE SESSIONS

$ *ed*	Creates a file.
a	Appends the following text.
Today	Text of file being created.
is	
Friday.	
.	Dot ends append (input) mode.
w F1	Writes contents of buffer to F1.
17	17 characters written, including Newlines.
q	Quits *ed*.
$	Shell prompt.

$ *ed F2* 73	Invokes *ed* and reads *F2* into buffer. *F2* is 73 characters long, including Newlines. *Note* — If *F2* were a *new* file (instead of a previously created file, as in this example), *ed* would print "?F2". This would mean that the file did not exist in your directory. You could then input the text as in the preceding example.
1,6p	Prints lines 1 through 6.
now is the time for all good men to come to the aid of the party. party.	The word "party." is duplicated.
6d	Deletes line 6.
1s/n/N/p Now is the time	Substitutes "N" for "n" on line 1 and prints the line.
2s/men/people/	Substitutes "people" for "men" on line 2.
6p ?	Attempts to print line 6. Line number out of range.
5p of the party.	Prints line 5.
s/e/eir/	Substitutes "eir" for "e" on current line (line 5).
w 71	Writes the buffer to *F2*. F2 is 71 characters long, including Newlines.
1,5p Now is the time for all good people to come to the aid of their party.	Prints lines 1 through 5.
q $	Quits *ed*. Shell prompt.

LINE ADDRESSING

INTRODUCTION

Line addressing can be done in one of the three ways
listed immediately below. In addition, each address
can take one of the five forms discussed below.

- *Single line*
  Example — *4p* prints line 4.

- *Range of lines* — Expressed as the beginning line
  and the ending line, separated by a comma.
  Example — *4,6p* prints lines 4-6.

- *Default* — Default line numbers are shown in
  parentheses in the format for each command.
  Example — See "*Default*" on top of page *14*.

FORM 1 — SPECIAL SYMBOLS

$ Last line of the buffer.

. Current line.

, When used *alone* with a command as a line
   address, stands for the first line through the
   last line of the buffer (that is, 1,$).
   Example — ,p

;   When used *alone* with a command as a line address, stands for the current line through the last line of the buffer (that is, .,$).
Example — ;p

√   "Escape" character. Used to temporarily escape *ed*, enter a Shell level command and execute it, and return to *ed*.
Example — !spell F1

FORM 2 — NUMBERS

Examples —
- *3* = Line 3
- *32,76* = Lines 32 - 76

FORM 3 — ARITHMETIC EXPRESSIONS

If an address —

- *BEGINS* with one or more + or − signs — Then one or more is added to, or subtracted from, the current line. *Examples* —
  - −5 = .−5
  - −−5 = .−6
  - +++1 = .+3
  - +++2 = .+4

- *ENDS* with one or more + or − signs — Then 1 or more is added to, or subtracted from, the address. *Examples* —
  - 3+   = 4
  - 3+++   = 6
  - 3−−   = 1
  - 2−   = 1

- Consists of *ONLY* + or − signs — It refers to the number of lines preceding or following the current line. *Example* —
  - +++ = 3 lines after current line.

- Is *FOLLOWED* by a + or − sign, followed by a decimal number — This indicates the address, plus or minus the specified number of lines. The + sign may be omitted. *Examples* —
  - .+5   Skips ahead 5 lines from current line
  - $−10,$p   Prints last 10 lines of file

FORM 4 — REGULAR EXPRESSIONS (REs)

Formats

/string/   *FORWARD* context search for *string*. Starts search at line *following* the current line ... goes *down* to end of buffer ... then wraps around to start of buffer and continues down to the current line.
- *Example* — /A/,/e/p
          A computer is
          executing.

//   As above ... except searches for *last remembered* string.
- *Example* — /8/p
          on page 8.
          //p
          chapter 8

?string?   *BACKWARD* context search for *string*. Starts search at line *preceding* the current line ... goes *up* to start of buffer ... then wraps around to end of buffer and continues up to current line.
Note — If used as a string delimiter in the *global, GLOBAL,* or *substitute* command, *?* would *not* cause a backward search.

??   As above ... except searches for *last remembered* string.

Note 1   Typically, addresses are separated from each other by a *comma* (,). They may also be separated by a *semicolon* (;). In the latter case, the current line (.) is set to the first address, and only then is the second address calculated. This feature can be used to determine the starting line for forward and backward searches. The *second address* of any two-address sequence must correspond to a line that follows, in the buffer, the line corresponding to the first address. *Example* — /string/;/word/p Prints the lines from the next line containing *string*, through the line after that line which contains *word*.

Note 2   If a closing delimiter would be the last character before a Newline, that delimiter may be omitted, in which case the addressed line is printed. *Examples* — The following pairs of commands are equivalent —

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

Special Characters

****   Escapes the special meaning of a character. *Example* —
- s/*/star/ — Replaces * with *star* on current line.

.   Dot (.) matches any single character except Newline. *Example* —
- /c.pe/p — Matches and prints cape, cope, caper, ...

*   Matches zero or more occurrences of the preceding character. If there is a choice, the longest leftmost string that allows a match is chosen. *Example* —
- /sto*p/ — Matches zero or more o's — stp, stop, stoop, ...

.* Matches zero or more occurrences of any character string. *Examples —*
- /car.*n/p — Matches carton, cartoon, carmen, ...
- s/xyz.*//p — Removes from *xyz* to end of current line.

[] Matches one of the enclosed characters. *Example —*
- /b[aeu]rn/p — Matches barn, auburn, ...

[][] Matches two character positions. *Example —*
- /the[ir][er]/p — Matches their, theie, there, or therr.

[D-F] Matches a range of characters or numbers. *Example —*
- /Version[D-F]/p — Matches VersionD, VersionE, or VersionF.

[^0-9] ^ as the *first* character means do *not* match the specified characters or numbers, or Newline. *Example —*
- /Version[^0-9]/p — Does *not* match Version0 - Version9.

^string ^ as the *first* character means the following characters should be the first characters on a line. *Examples —*
- /^Columbus/p — Prints line that *starts* with *Columbus*.
- /^[0-9][0-9]$/p — Prints next line containing only a 2-digit number.

\{ \} /^[0-9]\{2\}$/p is equivalent to the example immediately above.

^ ^ as the *only* character means to insert a word at the start of the line. *Example —*
- s/^/and/ — Inserts word *and* at the start of the current line.

/string$/ $ as the *last* character means the preceding characters should be the last characters in a line. *Example —*
- /boat$/   Matches *boat* at the big boat   the *end* of a line.

s/$/string/ $ as the *only* character means to append a word at the end of the line. *Example —*
- s/$/./ — Places . at the end of the current line.

/^string$/ Finds a line that consists of only the characters specified. *Examples —*
- /^boat$/ — Finds a line that contains only the string *boat*.
- /^aa*$/p — Prints next line containing only the character *a*, any number of times.

FORM 5 — CAN MIX FORMS

Examples —
- 6,$-1p — Prints lines 6 through next-to-last line.
- .,/zz/p — Prints the current line through the next line containing *zz*.
- ?abc?,.p — Prints the lines from the line containing *abc* before the current line, through the current line.
- /a1/,/a4/p — Prints the lines, from the next line containing *a1* through the next line containing *a4*.

COMMANDS

- All commands are *single letters*. Most are lowercase but some, as shown, are uppercase.
- *Spaces* are optional between the address, and the command. (You do not need *any* spaces.)
- *One Command Per Line* — It is illegal for more than one command to appear on a line. However, any command (except *edit, filename, read,* and *write*) may be suffixed by *l* or *n* or *p*, as discussed under the *list/number/print* commands.
- *Current line after execution* — This is shown for each command. If you are in any doubt, just type .= to have the line number of the current line printed.
- *Default* — Default line numbers are shown in parentheses in the format for each command. *CAUTION* — The parentheses are *NOT* part of the format! *Example* — The format for the *move* command is *(.,.)mz*. This means that you can specify just one address, or a starting and ending address. If you specify *no* address, the current line (dot) is the default. See examples under *move*, below.

```
append — (.)a
change — (.,.)c
insert  — (.)i
```

Examples —
- Adds lines —
  - *0a* — At start of buffer.
  - *a* — After current line.
  - *3a* — After line 3.
  - *$a* — At end of buffer.
  - *0i* — Illegal.
  - *i* — Before current line.
  - *3i* — Before line 3.
  - *$i* — Inserts before last line of buffer.

- Type *a* or *c* or *i* on line by itself ... then the line(s) of text ... conclude with dot (.) on line by itself.
- *c* is similar to *a*, except that *c* —
  - Also deletes lines.
  - Can have 2 addresses, e.g., *3,6c* deletes lines 3-6 and then adds lines.
- *Current line after execution* = Last line added. For *a* and *i*, if *no* lines were added, current line is left at addressed line. For *c*, if *no* lines were added, current line is left at first line not deleted. For *a, c,* and *i* — If lines were added, current line is left at last line added. *Note* — When you first edit a file, the last line of the buffer is the current line.

```
copy — (.,.)tz
move — (.,.)mz
```

Examples —
- Moves lines —
  - *4,6m0* — 4-6 to start of file.
  - *4,6m8* — 4-6 to after line 8.
  - *.m8* or *m8* — Current line, to after line 8.
  - *4,6m5* — Illegal (5 is within range of 4,6).
  - *18m22* — 18, to after line 22.
  - *.,3m10* — Current line to 3, to after line 10.

- *z* is the location to move to.
- *m deletes* original lines ... *t saves* original lines.
- *Current line after execution* = New address of last line moved.

crypt — X

- A key string is demanded from the standard input. Subsequent *edit, read* and *write* commands will encrypt and decrypt the text with this key. An explicitly empty key turns it off.
- To turn the key off, enter *X* and then, when a key is requested, press *RETURN*.
- *Current line after execution* = Unchanged.

delete — (.,.)d

Examples —

- Deletes —
  - *d* — Current line
  - *3d* — Line 3
  - *3,6d* — Lines 3-6
- *Current line after execution* = Line after deleted lines. If deleted lines were at end of file, the new last line becomes the current line.

edit — $\begin{Bmatrix} e \\ E \end{Bmatrix}$ [*filename*]

read — ($)r [*filename*]

Examples —

- *e F1* — Clears buffer ... reads in file F1. Before clearing the buffer, a *?* is printed if you changed the buffer since your last *write* command. You can then use the *write* command to save your changes ... or repeat the *e* command to lose them.
  Note — This check is *not* made if you accessed *ed* with the minus (−) option.
- *E F1* — As above. Does *not* check for changes.
- Reads F1 into buffer —
  - *r F1* — After last line
  - *$r F1* — After last line
  - *10r F1* — After line 10
  - *0r F1* — At start of buffer
  - *.r F1* — After current line.
- Applies to both *edit* and *read* —
  - If *filename IS* specified — Current filename is *not* changed, unless it is the *first* filename specified since *ed* was invoked.
  - If *filename NOT* specified — Current filename is used. If there is none, you must specify it now.
  - *filename* is remembered for possible use as default file in subsequent *edit*, *read*, and *write* commands.
  - *Current line after execution* = Last line read in.
  - If filename begins with ! — It is assumed to be a Shell command whose output is to be read in. This filename is *not* remembered as the current filename.

filename — f [filename]

Examples —

- *f* — Prints current filename.
- *f F2* — Changes filename to F2, and then prints it.
- *Current line after execution* = Unchanged.

global — (1,$) $\begin{Bmatrix} g \\ v \end{Bmatrix}$ /string/commands

Examples —

- *g/abc/p* — Prints lines containing *abc*.
- *g/thier/s//their/gp* — Substitutes *their* for each occurrence of *thier*.
- *1,10g/abc/s//xyz/gp* — Substitutes in lines 1-10.
- *g/xyz/.m$p* — Moves to end of buffer, then prints lines containing *xyz*.
- *v?xyz?* — Scans buffer, printing lines that do *not* contain *xyz*. *Note* — Instead of slashes, you can use any other character (*?* is used in this example) as the delimiter, *except* Spaces and Newlines.
- *g* and *v* search buffer and mark lines that —
  - *g* — *MATCH* the *string*.
  - *v* — *DON'T* match the *string*.
- *commands*
  - Commands are executed for every line marked, with current line initially set to that line.
  - *p* is default if no commands specified.
  - A single command, or the first of a list of commands, should be on the same line as the global command. All lines of a multiline list, except the last line, must be ended with a backslash (\).
  - *append, change,* and *insert* commands and associated input are allowed. You can omit the dot (.) terminator if it is the last line of the command list.
  - *Not* allowed — *g, G, v, V.*
- *Current line after execution* = Last line found if *match* for *g* (or if *non*-match for *v*). Otherwise, unchanged.

GLOBAL INTERACTIVE —
(1,$) $\begin{Bmatrix} G \\ V \end{Bmatrix}$ /string/

- *Example —*
  G/[0-9][0-9][0-9][0-9][0-9]/
  Old style code 10000
  s//0&/ — Adds leading zero
  this is new style code 010001
  Press RETURN — No change
  another old style code 10002
  & — Repeats previous command
  text containing numbers 12345
  Press RETURN — No change
- *G* and *V* are the same, except that lines are marked that —
  - *G* — *MATCH* the *string*.
  - *V* — *DON'T match* the *string*.
- For every line that is marked —
  - The line is printed.
  - Current line is set to that line.
  - You can type any *ONE* command *except append ... change ... insert ... g ... G ... v ... V.*
  - The command is executed.
  - The next marked line is printed, etc.

- *Newline* — Acts as a null command.
- *&* — Causes reexecution of the most recent command executed within the current invocation of *G*.
- The commands input as part of the execution of the *G* command may address and affect *any* lines in the buffer.
- To terminate *G* command — Press *DEL* or *BREAK*.
- *Current line after execution* = Last line found if *match* for *G* (or if *non-match* for *V*). Otherwise, unchanged.

join — (.,.+1)j

Examples —
- *10,20j* — Removes Newline characters between lines 10-20.
- *j* — Removes Newline between current line and next line.
- *10j* — Does nothing.
- *Current line after execution* = First line joined.

list   — (.,.)l
number — (.,.)n
print   — (.,.)p

Examples —	Prints —
• *.,.p* or *.p* or *p* or *.*	Current line
• *4p* or *4*	Line 4
• *$p* or *$*	Last line
• *1,$p* or *,p*	Line 1 to last line (all lines)
• *.,$p* or *;p*	Current line to last line
• *2,4p*	Lines 2-4
• *4,$p*	Line 4 to last line
• *.±±±p* or *.±3p*	Line 3 (—before, + after) current line
• *$———p* or *$—3p*	3rd line before last line.
• *——*	2 lines before current line
• *.+1p*	Line after current line
• *+++*	The line 3 lines past current line
• *.—4,.+4p*	4 lines before, through 4 lines after, current line (9 lines)
• *—,+p*	One line before, to one line after, current line (3 lines)
• *$-1p* or *$-p* or *$-*	Next-to-last line

- Prints lines with —
  - *list* — Non-printing characters (tab, backspace, etc.).
  - *number* — Line numbers and tabs.
  - *print* — No line numbers or tabs.
- *Default* — An address with no commands, assumes *p*. *Example* — *10* prints line 10. However, *10,20* would print only line 20.

- *Appending* — *l, n,* or *p* can be appended to any other command except *edit, filename, read,* and *write. Example* — *dp* deletes the current line, and prints the new current line.
- *RETURN* — Press this key to print next line.
- *($)=* — Prints specified line number. Default is last line. *Example* — *.=* prints number of current line.
- *Current line after execution* = Last line printed.

mark — (.)kx

Examples —
- */abc/kg* — Next line containing *abc* can be addressed by single quote, followed by the mark, e.g., *'g.*
- *'gp* — Prints line marked by */abc/kg/.*
- *mark* — The mark following the *k* can be *any* lower-case letter, *a-z*. If the same mark (e.g., *g*) is reused, original address is erased.
- *Current line after execution* = Unchanged.

quit — q
QUIT — Q

- *q* — Exits *ed* and returns to Shell mode ($ prompt). Prints *?* if you made changes since your last *write* command that wrote entire buffer. You can then use the *write* command to save your changes … or repeat the *q* command to lose them. *Note* — This check is *not* made if you accessed *ed* with the minus (—) option.
- *Q* — As above. Does *not* check for changes.

substitute — (.,.)s/oldstring/newstring/[g]

Examples —
- *1,6s/Pa/Pat/* Substitutes 1st occurrence of *Pa* with *Pat* on lines 1-6.
- *s/es//* Removes *es* on current line.
- *pat peters* Substitutes *P* for every occurrence *s/p/P/gp* of *p* on the current line; then prints Pat Peters current line.

- Searches addressed lines for occurrences of *oldstring*. If *g* —
  - *IS* specified — *ALL* (non-overlapped) matched oldstrings are replaced by *newstring*.
  - Is *NOT* specified — Only *FIRST* occurrence of *oldstring* is replaced by *newstring*.
  - *Note* — If the substitution *fails* on *all* addressed lines, *?* is printed.
- *Current line after execution* = Last line on which a substitution occurred.
- *Delimiters* — Instead of slashes (/), you can use any other characters *except* Spaces and Newlines. *Example* — *1,6s;I/O;Input/Output;*