# IEEE P1003.2a Draft 8 – December 1991

*Current column*: Move to the first nonblank character of the sentence.

#### 5.35.7.1.29 Move forward to beginning of sentence

*Synopsis*: `[`*count*`] )`

Move forward to the beginning of a sentence. A *count* shall move forward that many sentences.

*Current line*: Move to the line containing the beginning of the sentence.

*Current column*: Move to the first nonblank character of the sentence.

#### 5.35.7.1.30 Move back to preceding paragraph

*Synopsis*: `[`*count*`] {`

Move back to the beginning of the preceding paragraph. A *count* specifies the number of paragraphs to move backward. This command shall be affected by the `paragraph` option; see 5.10.7.5.12.

*Current line*: Move to the line containing the beginning of the previous paragraph.

*Current column*: Move to the first nonblank character of the current line.

#### 5.35.7.1.31 Move forward to next paragraph

*Synopsis*: `[`*count*`] }`

Move forward to the beginning of the next paragraph. A *count* specifies the number of paragraphs to move forward. This command shall be affected by the `paragraph` option; see 5.10.7.5.12.

*Current line*: Move to the line containing the beginning of the next paragraph.

*Current column*: Move to the first nonblank character of the current line.

#### 5.35.7.1.32 Move to specific column position

*Synopsis*: `[`*count*`] |`

Place the cursor in the column position identified by *count* (if that position exists on the line). If *count* is omitted, it shall default to 1. If the length of the current line is less than *count*, the cursor shall be placed at the end of the current line. If the column position is spanned by a `<tab>` or a wide character, the cursor shall be placed on the character following the *count*-th column position.

*Current line*: Unchanged.

*Current column*: Move to the column position represented by *count*.

#### 5.35.7.1.33 Reverse find character

*Synopsis* : [*count*] `,`

Reverse the last `f`, `F`, `t`, or `T` command, looking the other way in the current line. A *count* shall be equivalent to repeating the search that many times.

*Current line* : Unchanged.

*Current column* : Move to the first column position of the next character found.

#### 5.35.7.1.34 Move to first nonblank of previous line

*Synopsis* : [*count*] `-`

Move to the first nonblank character in the previous line. A *count* specifies how many lines to move back.

*Current line* : *line* − *count*

*Current column* : Move to the first nonblank character of the current line or the last character if the line is a blank line.

#### 5.35.7.1.35 Repeat

*Synopsis* : [*count*] `.`

Repeat the last command that changed the buffer. A *count* shall be passed on to the command being repeated.

*Current line* : Dependent on the results of the execution of the last command that changed the buffer.

*Current column* : Dependent on the results of the execution of the last command that changed the buffer.

#### 5.35.7.1.36 Find regular expression

*Synopsis* : `/`

Prompt the user to enter a string on the last line on the screen, interpret it as a regular expression (see 5.10.7.3), and scan forward for the next occurrence of a matching string. The search shall begin when a `<newline>` or an `<ESC>` is entered to terminate the pattern; it can be prematurely terminated with an interrupt. 7

When used with an operator to specify an extent of text, the defined region is from the current cursor position to the beginning of the matched string. Whole lines can be specified by giving an offset from the matched line (using a closing `/` followed by a `+`*n* or −*n*).

Regular expressions are described in `ex`; see 5.10.7.3.

*Current line* : Move to the line in which the first (or next) regular expression match occurred.

*Current column*: Move to the column of the first character of the matched string.

##### 5.35.7.1.37 Move to first character in line

*Synopsis*: `0` (zero)

Move to the first character on the current line. (The zero shall not be interpreted as a command when it is preceded by a nonzero digit.)

*Current line*: Unchanged.

*Current column*: Column position 1.

##### 5.35.7.1.38 Execute an `ex` command

*Synopsis*: `:`

Begin an `ex` command. The `:` character, as well as the entered command, shall be displayed on the bottom line. The command shall be executed when the input
is terminated by entering a `<newline>` or an `<ESC>`. Typing a `control-V` (or 8
`control-Q`) character shall escape the following character, allowing its literal 8
value to be entered. 8

NOTE: This special property of `control-V` and `control-Q` is only effective in visual or open modes 8
and in `ex` commands called from visual/open mode with the `:` command.

If the `ex` command causes the screen to be scrolled or causes an escape to the shell, `vi` shall display a message indicating that it is waiting for a `<newline>` character. When a `<newline>` or an `<ESC>` character is entered, the screen shall be refreshed.

*Current line*: Dependent on the results of the execution of the `ex` command.

*Current column*: Dependent on the results of the execution of the `ex` command.

##### 5.35.7.1.39 Repeat find

*Synopsis*: [*count*] `;`

Repeat the last character find using `f`, `F`, `t`, or `T`. A *count* shall move the cursor to (or next to) the *count*-th occurrence of the character.

*Current line*: Unchanged.

*Current column*: Move to the column containing the character being searched for. If no matching character is found, it shall be unchanged.

##### 5.35.7.1.40 Shift left

*Synopsis*: [*count*] `<` [*motion*]

Shift lines left one *shiftwidth* (see 5.10.7.5.20). The command can be followed by a motion command to specify lines. A *count* shall be passed through to the motion command.

When *motion* is <, it shall shift the current line (or *count* lines starting at the current one).

*Current line* : Unchanged.

*Current column* : Move to the first nonblank character of the line or the last character if the line is a blank line.

#### 5.35.7.1.41 Shift right

*Synopsis* : [*count* ] > [*motion* ]

Shift lines right one *shiftwidth* (see 5.10.7.5.20). The command can be followed by a motion command to specify lines. A *count* shall be passed through to the motion command.

When *motion* is >, it shall shift the current line (or *count* lines starting at the current one).

Empty lines shall not be changed. 7

*Current line* : Unchanged.

*Current column* : Move to the first nonblank character of the line or the last character if the line is a blank line.

#### 5.35.7.1.42 Scan backwards for regular expression

*Synopsis* : `?`

Scan backwards, the reverse of `/`. See 5.35.7.1.36.

*Current line* : Move to the line in which the next succeeding regular expression match occurred.

*Current column* : Move to the column of the first character of the matched string.

#### 5.35.7.1.43 Execute 7

*Synopsis* : @*buffer* 7

Execute each line of the named buffer as one or more `vi` commands (including `ex` 7
commands, as described in 5.35.7.1.38). If the buffer name is @, then the last 7
buffer executed shall be used; if there is no last buffer, an error shall occur. The 7
text of a macro may contain an @ character calling another macro; however, recur- 7
sively calling the same macro produces undefined behavior. A `vi` error shall ter- 7
minate all currently executing macros. All changes made during a macro call 7
shall be treated as a unit and can be undone with a single `u` command. 7

*Current line* : Dependent on the results of the execution of the `vi` command(s). 7

*Current column* : Dependent on the results of the execution of the `vi` command(s). 7

#### 5.35.7.1.44 Reverse case

*Synopsis* : [*count* ] ~

Reverse the case of the *count* character(s) beginning at the current cursor position. Lowercase alphabetic characters shall be changed to uppercase and uppercase characters changed to lowercase. This command shall have no effect on nonalphabetic characters.

*Current line* : Unchanged.

*Current column* : Move to the next column unless it is on the last character of the line, in which case it shall remain unchanged.

#### 5.35.7.1.45 Append

*Synopsis* : [*count* ] `a`

Enter input mode, appending the entered text after the current cursor position. A
*count* shall cause the inserted text up to the first `<newline>` to be replicated that 8
many times; if a `<newline>` appears in the inserted text, only one occurrence of it 8
and any characters following it shall be inserted. For example, if the current line 8
is `abc`, the command 8

```
03afoo<newline>bar
```
8

changes that line to 8

```
afoofoofoo
barbc
```
8
8

*Current line* : Unchanged. 7

*Current column* : *column* + 1. 7

#### 5.35.7.1.46 Append at end of line

*Synopsis* : [*count* ] `A`

Append *count* copies of the input text at the end of the current line, equivalent to `$`[*count* ] `a`. See the `a` command in 5.35.7.1.45.

*Current line* : Unchanged. 8

*Current column* : See 5.35.7.1.45.

#### 5.35.7.1.47 Move backward to preceding word

*Synopsis* : [*count* ] `b`

Move the cursor backward to the beginning of a word by repeating the following
algorithm *count* times: If the current position is at the beginning of a word, the 7
current position shall move to the first character of the preceding word. Otherwise, the current position shall move to the first character of the word at the current position. If no preceding word exists on the current line, the current position shall move to the first character of the last word on the first preceding line

that contains a word. For this command, an empty or blank line shall be con- 7
sidered to contain exactly one word. 7

*Current line* : Set to the line containing the word selected.

*Current column* : Set to the first character of the word selected.

#### 5.35.7.1.48 Move backward to preceding bigword

*Synopsis* : [ *count* ] `B`

Move the cursor backward to the beginning of a bigword by repeating the following algorithm *count* times: If the current position is at the beginning of a bigword or the character at the current position cannot be part of a bigword, the current position shall move to the first character of the preceding bigword. Otherwise, the current position shall move to the first character of the bigword at the current position. If no preceding bigword exists on the current line, the current position shall move to the first character of the last bigword on the first preceding line that
contains a bigword. For this command, an empty or blank line is considered to 7
contain exactly one bigword. 7

*Current line* : Set to the line containing the bigword selected. 7

*Current column* : Set to the first character of the bigword selected. 7

#### 5.35.7.1.49 Change

*Synopsis* : [ *buffer* ] [ *count* ] `c` *motion* 8

Delete the specified region of text and enter input mode to replace it with the entered text. If more than part of a single line is affected, the deleted text shall be saved in the numeric buffers. If only part of the current line is affected, then the last character to be deleted shall be marked with a $. A *count* shall be passed through to the motion command.

*Current line* : The current line and column position are dependent on the *motion*
command following the `c`. For example, `cw` changes a word, `c[[` changes from the 7
beginning of the current section, etc. 7

*Current column* : See *Current line* , above

#### 5.35.7.1.50 Change to end-of-line

*Synopsis* : [ *buffer* ] [ *count* ] `C` 8

Change text from the current position through the end of line (*line* + *count* − 1);
equivalent to [ *buffer* ] [ *count* ] `c$`. 8

*Current line* : See the `c` command in 5.35.7.1.49.

*Current column* : See 5.35.7.1.49.

#### 5.35.7.1.51 Delete

*Synopsis* : **[** *buffer* **] [** *count* **]** `d` *motion* 8

Delete the specified region of text. If more than part of a line is affected, the text
shall be saved in the numeric buffers as well as any named **[** *buffer* **]**. A *count* 8
shall be passed through to the motion command.

*Current line* : The current line and column position are dependent on the *motion* command following the `d`. For example, `dw` deletes a word, `d$` deletes to the end of the line, etc.

*Current column* : See *Current line* , above.

#### 5.35.7.1.52 Delete to end-of-line

*Synopsis* : **[** *buffer* **] [** *count* **]** `D` 8

Delete the text from the current position to the end of the current line; equivalent
to **[** *buffer* **] [** *count* **]** `d$`. 8

*Current line* : Unchanged.

*Current column* : Set to maximum of *column* − 1 or 1. 7

#### 5.35.7.1.53 Move to end of word

*Synopsis* : **[** *count* **]** `e`

Move the cursor forward to the end of a word, by repeating the following algo- 7
rithm *count* times: If the current position is the end of a word, the current posi- 7
tion shall move to the last character of the following word. Otherwise, the current 7
position shall move to the last character of the word at the current position. If no 7
succeeding word exists on the current line, the current position shall move to the 7
last character of the first word on the next following line that contains a word. 7
For this command, an empty or blank line is considered to contain exactly one 7
word. 7

*Current line* : Set to the line containing the word selected.

*Current column* : Set to the last character of the word selected.

#### 5.35.7.1.54 Move to end of bigword

*Synopsis* : **[** *count* **]** `E`

Move the cursor forward to the end of a bigword, by repeating the following algorithm *count* times: If the current position is the end of a bigword or the character at that position cannot be part of a bigword, the current position shall move to the last character of the following bigword. Otherwise, the current position shall move to the last character of the bigword at the current position. If no succeeding bigword exists on the current line, the current position shall move to the last character of the first bigword on the next following line that contains a bigword.

For this command, an empty or blank line is considered to contain exactly one bigword. 7

*Current line*: Set to the line containing the bigword selected. 7

*Current column*: Set to the last character of the bigword selected. 7

#### 5.35.7.1.55 Find character in current line (forward)

*Synopsis*: [*count*] `f` *character*

Scan the rest of the current line for the single character *character* and move the cursor to it if it is found. A *count* shall repeat the find that many times.

*Current line*: Unchanged.

*Current column*: Set to the column position containing the character that was scanned for. Unchanged if insufficient characters were found to satisfy the *count*.

#### 5.35.7.1.56 Find character in current line (reverse)

*Synopsis*: [*count*] `F` *character*

Scan backwards in the current line for the single character *character*, moving the cursor to it if it is found. A *count* shall be equivalent to repeating the search that many times. If the search is unsuccessful, the terminal shall be alerted and the current column position shall remain unchanged.

*Current line*: Unchanged.

*Current column*: Set to the last character found moving backwards.

#### 5.35.7.1.57 Move to line

*Synopsis*: [*count*] `G`

Move the cursor to line *count*, or to the last line of the editing buffer if *count* is not specified.

*Current line*: *count*, if specified; otherwise, the last line.

*Current column*: Move to the first nonblank character of the current line or the last character if the line is a blank line.

#### 5.35.7.1.58 Move to top of screen

*Synopsis*: [*count*] `H`

Move the cursor to the line (*count* − 1) lines from the top of the current window (i.e., the *count*-th line, one-based, currently displayed).

*Current line*: *count* on the screen, as described above.

*Current column*: Move to the first nonblank character of the current line or the last character if the line is a blank line.

#### 5.35.7.1.59 Insert before cursor

*Synopsis* : [*count* ] `i` 8

Enter input mode, inserting the entered text before the cursor. (See also the `a`
command in 5.35.7.1.45.) A *count* shall cause the inserted text up to the first 8
`<newline>` to be replicated that many times; if a `<newline>` appears in the 8
inserted text, only one occurrence of it and any characters following it shall be 8
inserted. 8

*Current line* : Unchanged.

*Current column* : Unchanged.

#### 5.35.7.1.60 Insert at beginning of line

*Synopsis* : [*count* ] `I` 8

Enter input mode at the beginning of a line. This command shall behave identically to the `^`[*count* ]`i` command. See 5.35.7.1.59. 8

*Current line* : Unchanged.

*Current column* : Move to the first nonblank character of the current line or the last character if the line is a blank line.

#### 5.35.7.1.61 Join

*Synopsis* : [*count* ] `J`

Join the current line with the next one, supplying appropriate spacing (in the 7
POSIX Locale): one `<space>` between words, two `<space>`s after a period, and no 7
`<space>`s at all when the last character of the first line is a `<blank>` or when the first character of the next line is `)`. In the case of lines ending with `<blank>`s, the `<blanks>` shall be retained, no spaces shall be added, and the lines shall be joined together. A *count* shall cause that many lines (minimum 2) to be joined, rather than 2. A count of 1 shall be treated as 2.

*Current line* : Unchanged.

*Current column* : Move to the character after the last character of the original line.

#### 5.35.7.1.62 Move to bottom of screen

*Synopsis* : [*count* ] `L`

Move the cursor to the first nonblank character of the last line on the screen. A *count* shall move to that line counting from the bottom. When used with an operator, whole lines shall be affected.

*Current line* : *last line* − *count*

*Current column* : Move to the first nonblank character of the current line or the last character if the line is a blank line.

#### 5.35.7.1.63 Mark position

*Synopsis* : `m` *letter*

Mark the current position of the cursor with a single POSIX Locale lowercase letter *letter*. The exact position is referred to by `` ` ``*letter*; the line is referred to by ´*letter*.

*Current line* : Unchanged.

*Current column* : Unchanged.

#### 5.35.7.1.64 Move to middle of screen

*Synopsis* : `M`

Move the cursor to the middle line on the screen, at the first nonblank position on the line.

*Current line* : Move to a line approximately in the middle of the display window.

*Current column* : Move to the first nonblank character of the current line or the last character if the line is a blank line.

#### 5.35.7.1.65 Repeat regular-expression find (forward)

*Synopsis* : `n`

Repeat the last `/` or `?` scanning command.

*Current line* : Set to the line containing the character string found by the scan. Unchanged if no match was found.

*Current column* : Move to the first character of the character string found by the scan. Unchanged if no match was found.

#### 5.35.7.1.66 Repeat regular-expression find (reverse)

*Synopsis* : `N`

Scan for the next match of the last pattern given to `/` or `?`, but in the reverse direction; this is the reverse of `n`.

*Current line* : Move to the matched line if a match is found or unchanged if no match is found.

*Current column* : Move to the first column position of the matched string or unchanged if no match is found.

#### 5.35.7.1.67 Insert empty line below

*Synopsis* : `o`

Open a line below the current line and enter input mode.

*Current line* : The newly-created line, (*line* +1).

*Current column* : If the editor option `autoindent` is not set, move to the first column; otherwise, as in the description of `autoindent` in `ex` (see 5.10.7.5.1).

#### 5.35.7.1.68 Insert empty line above

*Synopsis* : `O`

Open a new line above the current line and enter input mode.

*Current line* : Unchanged.

*Current column* : If the editor option `autoindent` is not set, move to the first column; otherwise, as described in the description of `autoindent` in `ex`.

#### 5.35.7.1.69 Put from buffer following

*Synopsis* : [*buffer*] `p` 7

Insert text from buffer *buffer* following the current column or current line, as 7
described for the `P` command (see 5.35.7.1.70). If *buffer* is omitted, the unnamed buffer shall be used.

*Current line* : (*Current line* +1) if the buffer contains whole lines; otherwise, unchanged.

*Current column* : First nonblank character of the inserted text if the buffer contained whole lines; otherwise, the last character of the inserted text.

#### 5.35.7.1.70 Put from buffer before

*Synopsis* : [*buffer*] `P`

Put the last deleted text back before/above the cursor. The text shall go back as whole lines above the cursor if it was deleted as whole lines. Otherwise, the text shall be inserted just before the cursor.

The command can be preceded by a named buffer specification (`"`*x*), to retrieve the contents of the buffer.

*Current line* : Unchanged.

*Current column* : Move to the last column position of the inserted characters. If the buffer contains whole lines, the current column shall be the first nonblank character of the inserted characters.

#### 5.35.7.1.71 Enter `ex` mode

*Synopsis* : `Q`

Quit from `vi` and enter `ex` command mode.

*Current line* : Unchanged.

*Current column* : Unchanged.

#### 5.35.7.1.72 Replace character

*Synopsis* : [*count* ] `r` *character*

Replace a character on the screen with the character entered. If *count* is specified, the single character entered shall replace the current character and each of the next *count* − 1 characters in the line. (For example, `7rx` changes the next seven characters to be `xxxxxxx`). Unlike the `R` command (see 5.35.7.1.73), entering a `<newline>` shall replace the character with a `<newline>`; however, when used with *count* , the next *count* characters in the line shall be replaced by a single `<newline>`.

*Current line* : Unchanged unless the replacement character is a `<newline>`, in which case the current line shall be incremented by 1.

*Current column* : Set to the last column position occupied by the replacement character, unless the replacement character is a `<newline>`, in which case the current column position shall be 1.

#### 5.35.7.1.73 Replace characters

*Synopsis* : `R`

Replace characters on the screen with characters entered. If the end of the existing line is encountered, it shall be as if insert mode was entered at that point. If a `<newline>` is entered before the end of the existing line, it shall be entered into the editing buffer as if it was inserted, and replacement shall continue on the newly created line. No other line but the one in which the `R` command was given shall be affected.

*Current line* : Set to the line at the end of the replaced text.

*Current column* : Set to the end of the replaced text, if text was replaced. If no text was replaced, set to the column position of the character after which text would have been replaced if that character is on the current line; otherwise, 1. 7

#### 5.35.7.1.74 Substitute character

*Synopsis* : [*buffer* ] [*count* ] `s` 8

Substitute *count* characters in the current line starting with the current column position; equivalent to [*buffer* ] [*count* ] `c<space>.` 8

*Current line* : Unchanged.

*Current column* : Unchanged.

#### 5.35.7.1.75 Substitute lines

*Synopsis* : [*buffer* ] [*count* ] `S` 8

Change whole lines (same as `cc`). A *count* shall change that many lines; *count* lines shall be deleted and `vi` shall enter input mode.

*Current line* : Unchanged.

*Current column* : Set to column position 1 or to the position indicated by the previous line if `autoindent` is set.

#### 5.35.7.1.76 Move cursor to before character (forward)

*Synopsis* : [*count* ] `t` *character*

Move the cursor forward within the current line to the character position just before a subsequent occurrence of *character* . A *count* shall place the cursor just before the *count* -th occurrence of the character when searching the line forwards. If the search is unsuccessful, the terminal shall be alerted and the cursor position shall remain unchanged.

*Current line* : Unchanged.

*Current column* : Move to the position before an occurrence of *character* following *column* (noninclusive).

#### 5.35.7.1.77 Move cursor to after character (reverse)

*Synopsis* : [*count* ] `T` *character*

Scan backwards in the current line for the single character *character* and if found, place the cursor just after that character. A *count* shall place the cursor just after the *count* -th occurrence of the character when searching the line backwards. If *count* -th occurrences of the character do not exist, the terminal shall be alerted and the current column position shall remain unchanged.

*Current line* : Unchanged.

*Current column* : Set to the column after the scanned character. Unchanged if the character was not found.

#### 5.35.7.1.78 Undo

*Synopsis* : `u`

Reverse the last change made to the current buffer. If repeated, the command shall alternate between these two states, thus is its own inverse. When used after an insert that inserted text on more than one line, the lines shall be saved in the numeric named buffers.

*Current line* : Move to the position of the first line changed, if the reversal affects only one line or represents an addition or change; otherwise, move to the line preceding the deleted text.

*Current column* : Move to the first nonblank character on the updated current line.

#### 5.35.7.1.79 Undo current line

*Synopsis* : `U`

Restore the current line to its state before the cursor was last moved to it.

*Current line* : Unchanged.

*Current column* : Set to column position 1 or to the position indicated by the previous line if `autoindent` is set.

#### 5.35.7.1.80 Move to beginning of word

*Synopsis* : [*count* ] `w`

Move the cursor forward to the beginning of a word by repeating the following
algorithm *count* times: If the current position is at the beginning of a word, the 7
current position shall move to the first character of the next word. If no subsequent word exists on the current line, the current position shall move to the first character of the first word on the first following line that contains a word.

For this command, an empty or blank line is considered to contain exactly one 7
word. 7

*Current line* : Set to the line containing the word selected. 7

*Current column* : Set to the first character of the word selected. 7

#### 5.35.7.1.81 Move to beginning of bigword

*Synopsis* : [*count* ] `W`

Move the cursor forward to the beginning of a bigword by repeating the following algorithm *count* times: If the current position is within a bigword or the character at that position cannot be part of a bigword, the current position shall move to the first character of the next bigword. If no subsequent bigword exists on the current line, the current position shall move to the first character of the first big-
word on the first following line that contains a bigword. For this command, an 7
empty or blank line is considered to contain exactly one bigword. 7

*Current line* : Set to the line containing the bigword selected.

*Current column* : Set to the first character of the bigword selected.

#### 5.35.7.1.82 Delete character at cursor

*Synopsis* : [*buffer* ] [*count* ] `x` 8

Delete *count* characters in the current line starting with the current column posi-
tion; equivalent to [*buffer* ] [*count* ] `dl`. If the number of characters to be deleted 8
is greater than or equal to the number of characters to the end of the line, delete all of the characters from the current position to the end of the line and move the cursor to the new last character on the line.

*Current line* : Unchanged.

*Current column* : Set to the greater of (*current column* − the width of the *count* 8
deleted characters) or 1. 8

#### 5.35.7.1.83 Delete character before cursor

*Synopsis* : **[** *buffer* **] [** *count* **]** `X` 8

Delete the character before the cursor, equivalent to **[** *buffer* **] [** *count* **]** `dh`. A *count* 8
shall repeat the effect, but only characters on the current line shall be deleted.

*Current line* : Unchanged.

*Current column* : Set to the greater of (*current column* − the width of the *count* deleted characters) or 1.

#### 5.35.7.1.84 Yank

*Synopsis* : **[** *buffer* **] [** *count* **]** `y` *motion*

Copy (‘‘yank’’) the area of text specified by *count* and *motion* into the unnamed buffer, as well as into any named buffer specified by *buffer* .

*Current line* : Unchanged.

*Current column* : Unchanged.

#### 5.35.7.1.85 Yank current line

*Synopsis* : **[** *buffer* **] [** *count* **]** `Y`

Place (yank) a *count* copies of the current line, equivalent to a **[** *buffer* **] [** *count* **]** `yy` 8
command; see 5.35.7.1.84. 8

*Current line* : Unchanged.

*Current column* : Unchanged.

#### 5.35.7.1.86 Redraw window

*Synopsis* : **[** *count1* **]** `z` **[** *count2* **]** *character*

Redraw the screen with a window *count2* lines long containing line *count1* placed as specified by *character* : `<newline>` specifies the top of the screen, `.` specifies the center of the screen, and `−` specifies the bottom of the screen. If *count1* is not specified, it shall default to the current line; if *count2* is not specified, it shall default to *window* .

*Current line* : *count1* , if specified; otherwise, unchanged.

*Current column* : Move to the first nonblank character of the current line or the first character if the line is a blank line.

#### 5.35.7.1.87 Exit

*Synopsis* : `ZZ`

Exits the editor, writing out the buffer if it was changed since the last write to any file. This command shall behave identically to the `ex` command `xit`; see 5.10.7.2.38.

### 5.35.8 Exit Status

The `vi` utility shall exit with one of the following values:

- 0 Successful completion.
- >0 An error occurred.

### 5.35.9 Consequences of Errors

Default.

### 5.35.10 Rationale. *(This subclause is not a part of P1003.2a)*

**Usage, Examples**

The following table is a condensed version of information contained in the normative text. It is presented here to facilitate the review for accuracy of the `vi` commands and the editor options that affect those commands.

| `vi` Command | Editor Option | Description |
|---|---|---|
| `control-B` | `window` | Determines screen size for paging. |
| `control-D` | `scroll` | Determines number of lines to be scrolled. |
| `control-F` | `window` | Determines screen size for paging. |
| `control-T` | `shiftwidth` | Move cursor forward shiftwidth positions. |
| `control-U` | `scroll` | Determines number of lines to be scrolled. |
| `{ }` | `paragraphs` | Defines additional paragraph boundaries for the `{` and `}` commands. |
| `[[ ]]` | `sections` | Defines additional section boundaries for the `[[` and `]]` commands. |
| `/` | `ignorecase` | Characters that have upper- and lowercase representations shall have those representations considered as equivalent for purposes of regular expression comparison. |
| | `magic` | When set, the characters `ˆ $ . * [ ~` have special meaning in regular expressions. When unset, only characters `ˆ $` have special |

| | | |
|---|---|---|
| | | meaning in regular expressions. |
| `<` | `shiftwidth` | Determines the number of character positions to be shifted. |
| `>` | `shiftwidth` | Determines the number of character positions to be shifted. |
| `?` | `ignorecase` | Characters that have upper- and lowercase representations shall have those representations considered as equivalent for purposes of regular expression comparison. |
| | `magic` | When set, the characters `^ $ . * [ ~` have special meaning in regular expressions. When unset, only characters `^ $` have special meaning in regular expressions. |
| `O, o` | `autoindent` | Starting indentation determined by the line on which the command is given. |
| `S` | `autoindent` | Starting indentation determined by the line being changed. |
| `ZZ` | `readonly` | Shall not allow a write to a file. |
| `z` | `window` | Redraw window with the number of lines equal to the default set by window. |

The `control-L` and `control-R` commands differ in their treatment of deleted ‘‘@’’lines. The former always redisplays the screen and does not affect the status of any @ line; the latter redisplays only when needed to process the deleted @ lines.

When executing an individual `ex` command by entering `:`, it is not possible to enter a `<newline>` as part of the command because it is considered the end of the command. As explained in 5.10.7.4, a `control-M` preceded by a `control-V` (or `control-Q`) escape character in a regular expression substitution is mapped into a `<newline>`. A different approach is to enter `ex` command mode by using the `vi` `Q` command (and later resuming visual mode with the `ex` ‘`vi`’’command). In `ex` command mode, the single-line limitation does not exist. So, for example, the following is valid:

```
Q
s/break here/break\
here/
vi
```

The uses described for `control-V` can also be accomplished with `control-Q`, which is useful on terminals that use `control-V` for the down-arrow function. However, most historical implementations use `control-Q` for the *termios* START character, so the editor will generally not receive the `control-Q` unless `stty –ixon` mode is set. Any of the command characters described in this standard can be made ineffective by their selection as *termios* control characters, using the `stty` utility or other methods described in POSIX.1 {8}.

**History of Decisions Made**

Historically the `control-B` command skips two lines between each pair of pages it displays from a file. The working group viewed this as a bug and decided to go along with how the command was documented (as overlapping, not skipping, two lines) instead of standardizing the historical implementation.

See the Rationale for the `ex` utility for more information on `vi`.

A ''restricted editor'' (both the historical `red` utility and modifications to `vi`) were considered and rejected for inclusion. Neither option provided the level of security that many users might expect.

The `[[`, `]]`, `(`, `)`, `}`, and `}` commands are all affected by ''section boundaries,'' but in current implementations not all of the commands recognize the same section boundaries. This is considered to be a bug rather than a feature and a unique section-boundary algorithm was not described for each command.

The `e` and `E` commands described here differ slightly from some historical imple- 7
mentations in that an empty or blank line is described as containing exactly one 7
word or bigword, just as for `b`, `B`, `w`, and `W`. On some systems, `E` command skips 7
over empty/blank lines. The working group felt that a consistent definition of big- 7
word between the `E` command and the others was sufficient justification to change 7
one so that they all treat empty/blank lines the same way. 7

The `=` (reindent) command was omitted because it was LISP language-specific. (See the `ex` Rationale for the reasons LISP support was omitted.)

The text in Standard Output reflects the usage of the verb ''display'' in this clause; some `vi` implementations actually use standard output to write to the terminal, but POSIX.2 does not require that to be the case.

The action taken for signals other than SIGINT, SIGCONT, and SIGHUP is 7
unspecified because some implementations attempt to save the editing buffer in a 7
useful state when other signals are received. 7

# 5.36 **who** — Display who is on the system

## 5.36.1 Synopsis

```
who [−mTu]
```

## 5.36.2 Description

The `who` utility shall list various pieces of information about accessible users. The domain of accessibility is implementation defined.

The default output shall contain at least the following information for accessible users, in an unspecified format: the user's login name, terminal name, and time at which the user logged in.

## 5.36.3 Options

The `who` utility shall conform to the utility argument syntax guidelines described in 2.10.2.

The following options shall be supported by the implementation:

- `−m` Output only information about the current terminal.
- `−T` Show the state of each terminal, as described in 5.36.6.1.
- `−u` Output each displayed user's ''idle time'' in addition to any other information. The idle time is the time since any activity occurred on the user's terminal. The method of determining this is unspecified. 8

## 5.36.4 Operands

None.

## 5.36.5 External Influences

### 5.36.5.1 Standard Input

None.

### 5.36.5.2 Input Files

None.

#### 5.36.5.3 Environment Variables

The following environment variables shall affect the execution of `who`:

**LANG** This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

**LC_ALL** This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

**LC_CTYPE** This variable shall determine the locale for the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments).

**LC_TIME** This variable shall determine the format and contents of the date and time strings.

**LC_MESSAGES** This variable shall determine the language in which messages should be written.

#### 5.36.5.4 Asynchronous Events

Default.

### 5.36.6 External Effects

#### 5.36.6.1 Standard Output

The `who` utility writes its default information to the standard output in an implementation-defined format, subject only to the requirement of containing the information described above.

The following format shall be used for the `−T` option:

`"%s %c %s %s\n"`, *<user name>*, *<terminal state>*,
*<terminal name>*, *<time of login>*

where *<terminal state>* is one of the following characters:

- `+` The terminal allows write access to other users.
- `-` The terminal denies write access to other users.
- `?` The terminal write-access state cannot be determined.

In the POSIX Locale, the *<time of login>* shall be equivalent in format to the output of:

```
date +"%b %e %H:%M"
```
7