

NAME

ispell, buildhash, munchlist, findaffix, tryaffix, icombine, ijoin – Interactive spelling checking

SYNOPSIS

ispell [*common-flags*] [-M|-N] [-L*context*] [-V] *files*

ispell [*common-flags*] -I

ispell [*common-flags*] [-f *file*] [-s] [-a|-A]

ispell [-d *file*] [-w *chars*] -c

ispell [-d *file*] [-w *chars*] -e[e]

ispell [-d *file*] -D

ispell -v[v]

common-flags:

[-t] [-n] [-H] [-b] [-x] [-B] [-C] [-P] [-m] [-S] [-d *file*] [-p *file*] [-w *chars*] [-W *n*] [-T *type*] [-k*name list*] [-F *program*]

buildhash [-s] *dict-file affix-file hash-file*

buildhash -s *count affix-file*

munchlist [-l *aff-file*] [-c *conv-file*] [-T *suffix*] [-s *hash-file*] [-D] [-v] [-w *chars*] [*files*]

findaffix [-p|-s] [-f] [-c] [-m *min*] [-M *max*] [-e *elim*] [-t *tabchar*] [-l *low*] [*files*]

tryaffix [-p|-s] [-c] *expanded-file affix*[+*addition*]

icombine [-T *type*] [*aff-file*]

ijoin [-s|-u] *join-options file1 file2*

DESCRIPTION

Ispell is fashioned after the *spell* program from ITS (called *ispell* on Twenex systems.) The most common usage is "ispell filename". In this case, *ispell* will display each word which does not appear in the dictionary at the top of the screen and allow you to change it. If there are "near misses" in the dictionary (words which differ by only a single letter, a missing or extra letter, a pair of transposed letters, or a missing space or hyphen), then they are also displayed on following lines. As well as "near misses", *ispell* may display other guesses at ways to make the word from a known root, with each guess preceded by question marks. Finally, the line containing the word and the previous line are printed at the bottom of the screen. If your terminal can display in reverse video, the word itself is highlighted. You have the option of replacing the word completely, or choosing one of the suggested words. Commands are single characters as follows (case is ignored):

R	Replace the misspelled word completely.
Space	Accept the word this time only.
A	Accept the word for the rest of this <i>ispell</i> session.
I	Accept the word, capitalized as it is in the file, and update private dictionary.
U	Accept the word, and add an uncapitalized (actually, all lower-case) version to the private dictionary.
0-n	Replace with one of the suggested words.
L	Look up words in system dictionary (controlled by the WORDS compilation option).
X	Write the rest of this file, ignoring misspellings, and start next file.
Q	Exit immediately and leave the file unchanged.
!	Shell escape.
^L	Redraw screen.
^Z	Suspend <i>ispell</i> .

? Give help screen.

If the **-M** switch is specified, a one-line mini-menu at the bottom of the screen will summarize these options. Conversely, the **-N** switch may be used to suppress the mini-menu. (The minimenu is displayed by default if *ispell* was compiled with the MINIMENU option, but these two switches will always override the default).

If the **-L** flag is given, the specified number is used as the number of lines of context to be shown at the bottom of the screen (The default is to calculate the amount of context as a certain percentage of the screen size). The amount of context is subject to a system-imposed limit.

If the **-V** flag is given, characters that are not in the 7-bit ANSI printable character set will always be displayed in the style of "cat -v", even if *ispell* thinks that these characters are legal ISO Latin-1 on your system. This is useful when working with older terminals. Without this switch, *ispell* will display 8-bit characters "as is" if they have been defined as string characters for the chosen file type.

"Normal" mode, as well as the **-I**, **-a**, and **-A** options and interactive mode (see below) also accepts the following "common" flags on the command line:

- t** The input file is in TeX or LaTeX format.
- n** The input file is in nroff/troff format.
- H** The input file is in SGML/HTML format. (This should really be **-s**, but for historical reasons that flag was already taken.)
- b** Create a backup file by appending ".bak" to the name of the input file.
- x** Don't create a backup file.
- B** Report run-together words with missing blanks as spelling errors.
- C** Consider run-together words as legal compounds.
- P** Don't generate extra root/affix combinations.
- m** Make possible root/affix combinations that aren't in the dictionary.
- S** Sort the list of guesses by probable correctness.
- d file** Specify an alternate dictionary file. For example, use **-d deutsch** to choose a German dictionary in a German installation.
- p file** Specify an alternate personal dictionary.
- w chars**
Specify additional characters that can be part of a word.
- W n** Specify length of words that are always legal.
- T type** Assume a given formatter type for all files.

The **-H**, **-n**, and **-t** options select whether *ispell* runs in HTML (**-H**), nroff/troff (**-n**), or TeX/LaTeX (**-t**) input mode. (The default mode is controlled by the DEFTXFLAG installation option, but is normally nroff/troff mode for historical reasons.) Unless overridden by one of the mode-selection switches, TeX/LaTeX mode is automatically selected if an input file has the extension ".tex", and HTML mode is automatically selected if an input file has the extension ".html" or ".htm".

In HTML mode, HTML tags delimited by <> signs are skipped, except that the "ALT=" construct is recognized if it appears with no spaces around the equals sign, and the text inside is spell-checked.

In TeX/LaTeX mode, whenever a backslash ("\") is found, *ispell* will skip to the next whitespace or TeX/LaTeX delimiter. Certain commands contain arguments which should not be checked, such as labels and reference keys as are found in the \cite command, since they contain arbitrary, non-word arguments. Spell checking is also suppressed when in math mode. Thus, for example, given

```
\chapter {This is a Ckapter} \cite{SCH86}
```

ispell will find "Ckapter" but not "SCH". The **-t** option does not recognize the TeX comment character

"%", so comments are also spell-checked. It also assumes correct LaTeX syntax. Arguments to infrequently used commands and some optional arguments are sometimes checked unnecessarily. The bibliography will not be checked if *ispell* was compiled with **IGNOREBIB** defined. Otherwise, the bibliography will be checked but the reference key will not.

References for the *tib*(1) bibliography system, that is, text between a “[.” or “<.” and “.]” or “.>” will always be ignored in TeX/LaTeX mode.

The **-b** and **-x** options control whether *ispell* leaves a backup (.bak) file for each input file. The .bak file contains the pre-corrected text. If there are file opening / writing errors, the .bak file may be left for recovery purposes even with the **-x** option. The default for this option is controlled by the DEFNOBACKUPFLAG installation option.

The **-B** and **-C** options control how *ispell* handles run-together words, such as "notthe" for "not the". If **-B** is specified, such words will be considered as errors, and *ispell* will list variations with an inserted blank or hyphen as possible replacements. If **-C** is specified, run-together words will be considered to be legal compounds, so long as both components are in the dictionary, and each component is at least as long as a language-dependent minimum (3 characters, by default). This is useful for languages such as German and Norwegian, where many compound words are formed by concatenation. (Note that compounds formed from three or more root words will still be considered errors). The default for this option is language-dependent; in a multi-lingual installation the default may vary depending on which dictionary you choose.

The **-P** and **-m** options control when *ispell* automatically generates suggested root/affix combinations for possible addition to your personal dictionary. (These are the entries in the "guess" list which are preceded by question marks.) If **-P** is specified, such guesses are displayed only if *ispell* cannot generate any possibilities that match the current dictionary. If **-m** is specified, such guesses are always displayed. This can be useful if the dictionary has a limited word list, or a word list with few suffixes. However, you should be careful when using this option, as it can generate guesses that produce illegal words. The default for this option is controlled by the dictionary file used.

The **-S** option suppresses *ispell*'s normal behavior of sorting the list of possible replacement words. Some people may prefer this, since it somewhat enhances the probability that the correct word will be low-numbered.

The **-d** option is used to specify an alternate hashed dictionary file, other than the default. If the filename does not contain a "/", the library directory for the default dictionary file is prefixed; thus, to use a dictionary in the local directory "-d ./xxx.hash" must be used. This is useful to allow dictionaries for alternate languages. Unlike previous versions of *ispell*, a dictionary of /dev/null is illegal, because the dictionary contains the affix table. If you need an effectively empty dictionary, create a one-entry list with an unlikely string (e.g., "qqqqq").

The **-p** option is used to specify an alternate personal dictionary file. If the file name does not begin with "/", \$HOME is prefixed. Also, the shell variable WORDLIST may be set, which renames the personal dictionary in the same manner. The command line overrides any WORDLIST setting. If neither the **-p** switch nor the WORDLIST environment variable is given, *ispell* will search for a personal dictionary in both the current directory and \$HOME, creating one in \$HOME if none is found. The preferred name is constructed by appending ".ispell_" to the base name of the hash file. For example, if you use the English dictionary, your personal dictionary would be named ".ispell_english". However, if the file ".ispell_words" exists, it will be used as the personal dictionary regardless of the language hash file chosen. This feature is included primarily for backwards compatibility.

If the **-p** option is *not* specified, *ispell* will look for personal dictionaries in both the current directory and the home directory. If dictionaries exist in both places, they will be merged. If any words are added to the personal dictionary, they will be written to the current directory if a dictionary already existed in that place; otherwise they will be written to the dictionary in the home directory.

The **-w** option may be used to specify characters other than alphabetics which may also appear in words. For instance, **-w "&"** will allow "AT&T" to be picked up. Underscores are useful in many technical documents. There is an admittedly crude provision in this option for 8-bit international characters. Non-printing characters may be specified in the usual way by inserting a backslash followed by the octal

character code; e.g., "\014" for a form feed. Alternatively, if "n" appears in the character string, the (up to) three characters following are a DECIMAL code 0 - 255, for the character. For example, to include bells and form feeds in your words (an admittedly silly thing to do, but aren't most pedagogical examples):

```
n007n012
```

Numeric digits other than the three following "n" are simply numeric characters. Use of "n" does not conflict with anything because actual alphabets have no meaning - alphabets are already accepted. *IsPELL* will typically be used with input from a file, meaning that preserving parity for possible 8 bit characters from the input text is OK. If you specify the -l option, and actually type text from the terminal, this may create problems if your stty settings preserve parity.

It is not possible to use -w with certain characters. In particular, the flag-marker character for the language (defined in the affix file, but usually "/") can never be made into a word character.

The -W option may be used to change the length of words that *ispell* always accepts as legal. Normally, *ispell* will accept all 1-character words as legal, which is equivalent to specifying "-W 1." (The default for this switch is actually controlled by the MINWORD installation option, so it may vary at your installation.) If you want all words to be checked against the dictionary, regardless of length, you might want to specify "-W 0." On the other hand, if your document specifies a lot of three-letter acronyms, you would specify "-W 3" to accept all words of three letters or less. Regardless of the setting of this option, *ispell* will only generate words that are in the dictionary as suggested replacements for words; this prevents the list from becoming too long. Obviously, this option can be very dangerous, since short misspellings may be missed. If you use this option a lot, you should probably make a last pass without it before you publish your document, to protect yourself against errors.

The -T option is used to specify a default formatter type for use in generating string characters. This switch overrides the default type determined from the file name. The *type* argument may be either one of the unique names defined in the language affix file (e.g., **nroff**) or a file suffix including the dot (e.g., **.tex**). If no -T option appears and no type can be determined from the file name, the default string character type declared in the language affix file will be used.

The -k option is used to enhance the behavior of certain deformatters. The *name* parameter gives the name of a deformatter keyword set (see below), and the *list* parameter gives a list of one or more keywords that are to be treated specially. If *list* begins with a plus (+) sign, it is added to the existing keywords; otherwise it replaces the existing keyword list. For example, **-ktextskip1 +bibliographystyle** adds "bibliographystyle" to the TeX skip-1 list, while **-khtmlignore pre,strong** replaces the HTML ignore list with "pre" and "strong". The lists available are:

textskip1

TeX/LaTeX commands that take a single argument that should not be spell-checked, such as "bibliographystyle". The default is "end", "vspace", "hspace", "cite", "ref", "parbox", "label", "input", "nocite", "include", "includeonly", "documentstyle", "documentclass", "usepackage", "selectlanguage", "pagestyle", "pagenumbering", "hyphenation", "pageref", and "psfig", plus "bibliography" in some installations. These keywords are case-sensitive.

textskip2

TeX/LaTeX commands that take two arguments that should not be spell-checked, such as "setlength". The default is "rule", "setcounter", "addtocounter", "setlength", "addtolength", and "settowidth". These keywords are case-sensitive.

htmlignore

HTML tags that delimit text that should not be spell-checked until the matching end tag is reached. The default is "code", "samp", "kbd", "pre", "listing", and "address". These keywords are case-insensitive. (Note that the content inside HTML tags, such as HREF=, is not normally checked.)

htmlcheck

Subfields that should be spell-checked even inside HTML tags. The default is "alt", so that the ALT= portion of IMG tags will be spell-checked. These keywords are case-insensitive.

All of the above keyword lists can also be modified by environment variables whose names are the same as above, except in uppercase, e.g., TEXSKIP1. The **-k** switch overrides (or adds to) the environment variables, and the environment variables override or add to the built-in defaults.

The **-F** switch specifies an external deformatter program. This program should read data from its standard input and write to its standard output. The program *must* produce exactly one character of output for each character of input, or *ispell* will lose synchronization and corrupt the output file. Whitespace characters (especially blanks, tabs, and newlines) and characters that should be spell-checked should be passed through unchanged. Characters that should not be spell-checked should be converted into blanks or other non-word characters. For example, an HTML deformatter might turn all HTML tags into blanks, and also blank out all text delimited by tags such as "code" or "kbd".

The **-F** switch is the preferred way to deformat files for *ispell*, and eventually will become the only way.

If *ispell* is invoked without any filenames or mode switches, it enters an interactive mode designed to let the user check the spelling of individual words. The program repeatedly prompts on standard output with "word:" and responds with either "ok" (possibly with commentary), "not found", or "how about" followed by a list of suggestions.

The **-l** or "list" option to *ispell* is used to produce a list of misspelled words from the standard input.

The **-a** option is intended to be used from other programs through a pipe. In this mode, *ispell* prints a one-line version identification message, and then begins reading lines of input. For each input line, a single line is written to the standard output for each word checked for spelling on the line. If the word was found in the main dictionary, or your personal dictionary, then the line contains only a '*'. If the word was found through affix removal, then the line contains a '+', a space, and the root word. If the word was found through compound formation (concatenation of two words, controlled by the **-C** option), then the line contains only a '-'.

If the word is not in the dictionary, but there are near misses, then the line contains an '&', a space, the misspelled word, a space, the number of near misses, the number of characters between the beginning of the line and the beginning of the misspelled word, a colon, another space, and a list of the near misses separated by commas and spaces. Following the near misses (and identified only by the count of near misses), if the word could be formed by adding (illegal) affixes to a known root, is a list of suggested derivations, again separated by commas and spaces. If there are no near misses at all, the line format is the same, except that the '&' is replaced by '?' (and the near-miss count is always zero). The suggested derivations following the near misses are in the form:

[prefix+] root [-prefix] [-suffix] [+suffix]

(e.g., "re+fry-y+ies" to get "refries") where each optional *px* and *sfx* is a string. Also, each near miss or guess is capitalized the same as the input word unless such capitalization is illegal; in the latter case each near miss is capitalized correctly according to the dictionary.

Finally, if the word does not appear in the dictionary, and there are no near misses, then the line contains a '#', a space, the misspelled word, a space, and the character offset from the beginning of the line. Each sentence of text input is terminated with an additional blank line, indicating that *ispell* has completed processing the input line.

These output lines can be summarized as follows:

```
OK:      *
Root:    + <root>
Compound:
        -
Miss:    & <original> <count> <offset>: <miss>, <miss>, ..., <guess>, ...
Guess:   ? <original> 0 <offset>: <guess>, <guess>, ...
None:    # <original> <offset>
```

For example, a dummy dictionary containing the words "fray", "Frey", "fry", and "refried" might produce the following response to the command "echo 'frqy refries | ispell -a -m -d ./test.hash":

```
(#) International Ispell Version 3.0.05 (beta), 08/10/91
& frqy 3 0: fray, Frey, fry
& refries 1 5: refried, re+fry-y+ies
```

This mode is also suitable for interactive use when you want to figure out the spelling of a single word.

The **-A** option works just like **-a**, except that if a line begins with the string "&Include_File&", the rest of the line is taken as the name of a file to read for further words. Input returns to the original file when the include file is exhausted. Inclusion may be nested up to five deep. The key string may be changed with the environment variable **INCLUDE_STRING** (the ampersands, if any, must be included).

When in the **-a** mode, *ispell* will also accept lines of single words prefixed with any of '*', '&', '@', '+', '-', '~', '#', '!', '%', '^, or '~'. A line starting with '*' tells *ispell* to insert the word into the user's dictionary (similar to the I command). A line starting with '&' tells *ispell* to insert an all-lowercase version of the word into the user's dictionary (similar to the U command). A line starting with '@' causes *ispell* to accept this word in the future (similar to the A command). A line starting with '+', followed immediately by **tex** or **nroff** will cause *ispell* to parse future input according the syntax of that formatter. A line consisting solely of a '+' will place *ispell* in TeX/LaTeX mode (similar to the **-t** option) and '-' returns *ispell* to nroff/troff mode (but these commands are obsolete). However, the string character type is *not* changed; the '~' command must be used to do this. A line starting with '~' causes *ispell* to set internal parameters (in particular, the default string character type) based on the filename given in the rest of the line. (A file suffix is sufficient, but the period must be included. Instead of a file name or suffix, a unique name, as listed in the language affix file, may be specified.) However, the formatter parsing is *not* changed; the '+' command must be used to change the formatter. A line prefixed with '#' will cause the personal dictionary to be saved. A line prefixed with '!' will turn on *terse* mode (see below), and a line prefixed with '%' will return *ispell* to normal (non-*terse*) mode. A line prefixed with '^' will turn on verbose-correction mode (see below); this mode can only be disabled by turning on terse mode with '%'.

Any input following the prefix characters '+', '-', '#', '!', '%', or '^' is ignored, as is any input following the filename on a '~' line. To allow spell-checking of lines beginning with these characters, a line starting with '~' has that character removed before it is passed to the spell-checking code. It is recommended that programmatic interfaces prefix every data line with an uparrow to protect themselves against future changes in *ispell*.

To summarize these:

*	Add to personal dictionary
@	Accept word, but leave out of dictionary
#	Save current personal dictionary
~	Set parameters based on filename
+	Enter TeX mode
-	Exit TeX mode
!	Enter terse mode
%	Exit terse mode
^	Enter verbose-correction mode
^	Spell-check rest of line

In *terse* mode, *ispell* will not print lines beginning with '*', '+', or '-', all of which indicate correct words. This significantly improves running speed when the driving program is going to ignore correct words anyway.

In *verbose-correction* mode, *ispell* includes the original word immediately after the indicator character in output lines beginning with '*', '+', and '-', which simplifies interaction for some programs.

The **-s** option is only valid in conjunction with the **-a** or **-A** options, and only on BSD-derived systems. If specified, *ispell* will stop itself with a **SIGTSTP** signal after each line of input. It will not read more input until it receives a **SIGCONT** signal. This may be useful for handshaking with certain text editors.

The **-f** option is only valid in conjunction with the **-a** or **-A** options. If **-f** is specified, *ispell* will write its results to the given file, rather than to standard output.

The **-v** option causes *ispell* to print its current version identification on the standard output and exit. If the switch is doubled, *ispell* will also print the options that it was compiled with.

The **-c**, **-e[1-4]**, and **-D** options of *ispell*, are primarily intended for use by the *munchlist* shell script. The **-c** switch causes a list of words to be read from the standard input. For each word, a list of possible root words and affixes will be written to the standard output. Some of the root words will be illegal and must be filtered from the output by other means; the *munchlist* script does this. As an example, the command:

```
echo BOTHER | ispell -c
```

produces:

```
BOTHER BOTHE/R BOTH/R
```

The **-e** switch is the reverse of **-c**; it expands affix flags to produce a list of words. For example, the command:

```
echo BOTH/R | ispell -e
```

produces:

```
BOTH BOTHER
```

An optional expansion level can also be specified. A level of 1 (**-e1**) is the same as **-e** alone. A level of 2 causes the original root/affix combination to be prepended to the line:

```
BOTH/R BOTH BOTHER
```

A level of 3 causes multiple lines to be output, one for each generated word, with the original root/affix combination followed by the word it creates:

```
BOTH/R BOTH
BOTH/R BOTHER
```

A level of 4 causes a floating-point number to be appended to each of the level-3 lines, giving the ratio between the length of the root and the total length of all generated words including the root:

```
BOTH/R BOTH 2.500000
BOTH/R BOTHER 2.500000
```

Finally, the **-D** flag causes the affix tables from the dictionary file to be dumped to standard output.

Unless your system administrator has suppressed the feature to save space, *ispell* is aware of the correct capitalizations of words in the dictionary and in your personal dictionary. As well as recognizing words that must be capitalized (e.g., George) and words that must be all-capitals (e.g., NASA), it can also handle words with "unusual" capitalization (e.g., "ITCorp" or "TeX"). If a word is capitalized incorrectly, the list of possibilities will include all acceptable capitalizations. (More than one capitalization may be acceptable; for example, my dictionary lists both "ITCorp" and "ITcorp".)

Normally, this feature will not cause you surprises, but there is one circumstance you need to be aware of. If you use "I" to add a word to your dictionary that is at the beginning of a sentence (e.g., the first word of this paragraph if "normally" were not in the dictionary), it will be marked as "capitalization required". A subsequent usage of this word without capitalization (e.g., the quoted word in the previous sentence) will be considered a misspelling by *ispell*, and it will suggest the capitalized version. You must then compare the actual spellings by eye, and then type "I" to add the uncapitalized variant to your personal dictionary. You can avoid this problem by using "U" to add the original word, rather than "I".

The rules for capitalization are as follows:

- (1) Any word may appear in all capitals, as in headings.
- (2) Any word that is in the dictionary in all-lowercase form may appear either in lowercase or capitalized (as at the beginning of a sentence).
- (3) Any word that has "funny" capitalization (i.e., it contains both cases and there is an uppercase character besides the first) must appear exactly as in the dictionary, except as permitted by rule (1). If the word is acceptable in all-lowercase, it must appear thus in a dictionary entry.

buildhash

The *buildhash* program builds hashed dictionary files for later use by *ispell*. The raw word list (with affix flags) is given in *dict-file*, and the the affix flags are defined by *affix-file*. The hashed output is written to *hash-file*. The formats of the two input files are described in *ispell*(4). The *-s* (silent) option suppresses the usual status messages that are written to the standard error device.

munchlist

The *munchlist* shell script is used to reduce the size of dictionary files, primarily personal dictionary files. It is also capable of combining dictionaries from various sources. The given *files* are read (standard input if no arguments are given), reduced to a minimal set of roots and affixes that will match the same list of words, and written to standard output.

Input for *munchlist* contains of raw words (e.g from your personal dictionary files) or root and affix combinations (probably generated in earlier *munchlist* runs). Each word or root/affix combination must be on a separate line.

The *-D* (debug) option leaves temporary files around under standard names instead of deleting them, so that the script can be debugged. Warning: this option can eat up an enormous amount of temporary file space.

The *-v* (verbose) option causes progress messages to be reported to *stderr* so you won't get nervous that *munchlist* has hung.

If the *-s* (strip) option is specified, words that are in the specified *hash-file* are removed from the word list. This can be useful with personal dictionaries.

The *-I* option can be used to specify an alternate *affix-file* for munching dictionaries in languages other than English.

The *-c* option can be used to convert dictionaries that were built with an older affix file, without risk of accidentally introducing unintended affix combinations into the dictionary.

The *-T* option allows dictionaries to be converted to a canonical string-character format. The suffix specified is looked up in the affix file (*-I* switch) to determine the string-character format used for the input file; the output always uses the canonical string-character format. For example, a dictionary collected from TeX source files might be converted to canonical format by specifying *-T tex*.

The *-w* option is passed on to *ispell*.

findaffix

The *findaffix* shell script is an aid to writers of new language descriptions in choosing affixes. The given dictionary *files* (standard input if none are given) are examined for possible prefixes (*-p* switch) or suffixes (*-s* switch, the default). Each commonly-occurring affix is presented along with a count of the number of times it appears and an estimate of the number of bytes that would be saved in a dictionary hash file if it were added to the language table. Only affixes that generate legal roots (found in the original input) are listed.

If the *-c* option is not given, the output lines are in the following format:

```
strip/add/count/bytes
```

where *strip* is the string that should be stripped from a root word before adding the affix, *add* is the affix to be added, *count* is a count of the number of times that this *strip/add* combination appears, and *bytes* is an estimate of the number of bytes that might be saved in the raw dictionary file if this combination is added to the affix file. The field separator in the output will be the tab character specified by the *-t* switch; the

default is a slash ("/").

If the **-c** ("clean output") option is given, the appearance of the output is made visually cleaner (but harder to post-process) by changing it to:

```
-strip+add<tab>count<tab>bytes
```

where *strip*, *add*, *count*, and *bytes* are as before, and *<tab>* represents the ASCII tab character.

The method used to generate possible affixes will also generate longer affixes which have common headers or trailers. For example, the two words "moth" and "mother" will generate not only the obvious substitution "+er" but also "-h+her" and "-th+ther" (and possibly even longer ones, depending on the value of *min*). To prevent cluttering the output with such affixes, any affix pair that shares a common header (or, for prefixes, trailer) string longer than *elim* characters (default 1) will be suppressed. You may want to set "elim" to a value greater than 1 if your language has string characters; usually the need for this parameter will become obvious when you examine the output of your *findaffix* run.

Normally, the affixes are sorted according to the estimate of bytes saved. The **-f** switch may be used to cause the affixes to be sorted by frequency of appearance.

To save output file space, affixes which occur fewer than 10 times are eliminated; this limit may be changed with the **-l** switch. The **-M** switch specifies a maximum affix length (default 8). Affixes longer than this will not be reported. (This saves on temporary disk space and makes the script run faster.)

Affixes which generate stems shorter than 3 characters are suppressed. (A stem is the word after the *strip* string has been removed, and before the *add* string has been added.) This reduces both the running time and the size of the output file. This limit may be changed with the **-m** switch. The minimum stem length should only be set to 1 if you have a *lot* of free time and disk space (in the range of many days and hundreds of megabytes).

The *findaffix* script requires a non-blank field-separator character for internal use. Normally, this character is a slash ("/"), but if the slash appears as a character in the input word list, a different character can be specified with the **-t** switch.

IsPELL dictionaries should be expanded before being fed to *findaffix*; in addition, characters that are not in the English alphabet (if any) should be translated to lowercase.

tryaffix

The *tryaffix* shell script is used to estimate the effectiveness of a proposed prefix (**-p** switch) or suffix (**-s** switch, the default) with a given *expanded-file*. Only one affix can be tried with each execution of *tryaffix*, although multiple arguments can be used to describe varying forms of the same affix flag (e.g., the **D** flag for English can add either *D* or *ED* depending on whether a trailing E is already present). Each word in the expanded dictionary that ends (or begins) with the chosen suffix (or prefix) has that suffix (prefix) removed; the dictionary is then searched for root words that match the stripped word. Normally, all matching roots are written to standard output, but if the **-c** (count) flag is given, only a statistical summary of the results is written. The statistics given are a count of words the affix potentially applies to and an estimate of the number of dictionary bytes that a flag using the affix would save. The estimate will be high if the flag generates words that are currently generated by other affix flags (e.g., in English, *bathers* can be generated by either *bath/X* or *bather/S*).

The dictionary file, *expanded-file*, must already be expanded (using the **-e** switch of *ispell*) and sorted, and things will usually work best if uppercase has been folded to lower with 'tr'.

The *affix* arguments are things to be stripped from the dictionary file to produce trial roots: for English, *con* (prefix) and *ing* (suffix) are examples. The *addition* parts of the argument are letters that would have been stripped off the root before adding the affix. For example, in English the affix *ing* normally strips *e* for words ending in that letter (e.g., *like* becomes *liking*) so we might run:

```
tryaffix ing ing+e
```

to cover both cases.

All of the shell scripts contain documentation as commentary at the beginning; sometimes these comments

contain useful information beyond the scope of this manual page.

It is possible to install *ispell* in such a way as to only support ASCII range text if desired.

icombine

The *icombine* program is a helper for *munchlist*. It reads a list of words in dictionary format (roots plus flags) from the standard input, and produces a reduced list on standard output which combines common roots found on adjacent entries. Identical roots which have differing flags will have their flags combined, and roots which have differing capitalizations will be combined in a way which only preserves important capitalization information. The optional *aff-file* specifies a language file which defines the character sets used and the meanings of the various flags. The **-T** switch can be used to select among alternative string character types by giving a dummy suffix that can be found in an **altstringtype** statement.

ijoin

The *ijoin* program is a re-implementation of *join*(1) which handles long lines and 8-bit characters correctly. The **-s** switch specifies that the *sort*(1) program used to prepare the input to *ijoin* uses signed comparisons on 8-bit characters; the **-u** switch specifies that *sort*(1) uses unsigned comparisons. All other options and behaviors of *join*(1) are duplicated as exactly as possible based on the manual page, except that *ijoin* will not handle newline as a field separator. See the *join*(1) manual page for more information.

ENVIRONMENT

DICTIONARY

Default dictionary to use, if no **-d** flag is given.

WORDLIST

Personal dictionary file name

INCLUDE_STRING

Code for file inclusion under the **-A** option

TMPDIR

Directory used for some of *munchlist*'s temporary files

TEXSKIP1

List of TeX keywords that have a single argument that *ispell* should ignore.

TEXSKIP2

List of TeX keywords that have two arguments that *ispell* should ignore.

HTMLIGNORE

List of HTML keywords that delimit text that should not be spell-checked.

HTMLCHECK

List of HTML fields that should always be spell-checked, even inside a tag.

FILES

`c:/usr/local/lib/english.hash`

Hashed dictionary (may be found in some other local directory, depending on the system).

`c:/usr/local/lib/english.aff`

Affix-definition file for *munchlist*

`/usr/dict/web2` or `/usr/dict/words`

For the Lookup function (depending on the **WORDS** compilation option).

`$HOME/.ispell_hashfile`

User's private dictionary

`.ispell_hashfile`

Directory-specific private dictionary

SEE ALSO

spell(1), *egrep*(1), *look*(1), *join*(1), *sort*(1), *sq*(1L), *tib*(1L), *ispell*(4L), *english*(4L)

BUGS

It takes several to many seconds for *ispell* to read in the hash table, depending on size.

When all options are enabled, *ispell* may take several seconds to generate all the guesses at corrections for a misspelled word; on slower machines this time is long enough to be annoying.

The hash table is stored as a quarter-megabyte (or larger) array, so a PDP-11 or 286 version does not seem likely.

Ispell should understand more *troff* syntax, and deal more intelligently with contractions.

Although small personal dictionaries are sorted before they are written out, the order of capitalizations of the same word is somewhat random.

When the `-x` flag is specified, *ispell* will unlink any existing .bak file.

There are too many flags, and many of them have non-mnemonic names.

Munchlist does not deal very gracefully with dictionaries which contain "non-word" characters. Such characters ought to be deleted from the dictionary with a warning message.

Findaffix and *munchlist* require tremendous amounts of temporary file space for large dictionaries. They do respect the TMPDIR environment variable, so this space can be redirected. However, a lot of the temporary space needed is for sorting, so TMPDIR is only a partial help on systems with an uncooperative *sort*(1). ("Cooperative" is defined as accepting the undocumented -T switch). At its peak usage, *munchlist* takes 10 to 40 times the original dictionary's size in Kb. (The larger ratio is for dictionaries that already have heavy affix use, such as the one distributed with *ispell*). *Munchlist* is also very slow; munching a normal-sized dictionary (15K roots, 45K expanded words) takes around an hour on a small workstation. (Most of this time is spent in *sort*(1), and *munchlist* can run much faster on machines that have a more modern *sort* that makes better use of the memory available to it.) *Findaffix* is even worse; the smallest English dictionary cannot be processed with this script in a mere 50Kb of free space, and even after specifying switches to reduce the temporary space required, the script will run for over 24 hours on a small workstation.

AUTHOR

Pace Willisson (pace@mit-vax), 1983, based on the PDP-10 assembly version. That version was written by R. E. Gorin in 1971, and later revised by W. E. Matson (1974) and W. B. Ackerman (1978).

Collected, revised, and enhanced for the Usenet by Walt Buehring, 1987.

Table-driven multi-lingual version by Geoff Kuenning, 1987-88.

Large dictionaries provided by Bob Devine (vianet!devine).

A complete list of contributors is too large to list here, but is distributed with the *ispell* sources in the file "Contributors".

VERSION

The version of *ispell* described by this manual page is International Ispell Version 3.1.20, 10/10/95.