To: Distribution

From: Bill Silver
Jerry Stern

Date: May 23, 1977

Subject: WORDPRO Dictionaries

M. West said:

"If you've got it, flaunt it."

INTRODUCTION

This memorandum describes proposed new standard command and subroutine interfaces for hyphenation and spelling error detection. The full power and flexibility of Multics is used to solve the problems associated with these word processing functions, problems that other computer systems can only solve poorly, or not at all.

Tools that perform hyphenation and spelling error detection exist today on Multics. They are used now to produce high quality documents. However, they are not currently standard Multics products. They are installed in the Multics TOOLS library or they are private commands.

The proposed new commands will become standard Multics products. This will make them better known and easier to use. They are very much like the existing nonstandard commands they replace. Thus it will be easy to learn to use the new commands and only a minimal development effort will be needed.

This memorandum contains the following sections:

- Hyphenation
- Spelling
- WORDPRO dictionaries
- Wordlist segments
- Summary of commands
- MPM documentation

Please send all comments and suggestions on this memorandum to the authors.

Send mail to: Bill Silver
Honeywell Information Systems
575 Tech. Sq.
Cambridge, Mass. 02139

or send Multics mail at M.I.T or System M to:

Silver.Multics

or call: (617) 492-9306
HVN 261-9306

HYPHENATION

Hyphenation is the process of connecting or separating a word by a hyphen. The Multics WORDPRO text formatter "runoff" can perform hyphenation when the next word to be output does not fit in the space remaining in a line.

Why Hyphenation is Needed

It is the process of text formatting that makes hyphenation necessary. Text formatting involves the right justification of text on a line, with accompanying padding of the line with blanks. This often generates unsightly rivers of vertical white space. If right justification is not performed, then the right-hand margin often becomes very ragged.

Since hyphenation allows more characters to be put on a line, it can help solve both of these problems. Adding more characters to a line reduces the text padding needed when right justification is performed. When right justification is not performed, hyphenation results in smoother right-hand margins.

New Standard Hyphenation Interface

Currently, the actual hyphenation of a word is performed by a user supplied subroutine, hyphenate_word_, that is called by runoff. In conjunction with a new implementation of runoff (see MTB-337), the subroutine hyphenate_word_ will be supplied as a standard Multics product. (See the MPM documentation for hyphenate_word_ presented in this memorandum.)

Hyphenation Problems Solved by the New Runoff

The many problems of hyphenation are solved by WORDPRO through the cooperation of runoff and the hyphenate_word_ subroutine. The main hyphenation task of runoff is to identify when hyphenation is required and then call hyphenate_word_ to obtain the correct hyphenation. The new runoff provides additional hyphenation capabilities. They are:

- the ability to explicitly turn hyphenation on or off within a document.
- the ability to specify the minimum necessary number of character positions remaining in an output line before hyphenation is performed.
- the ability to strip punctuation from the word being hyphenated.
- the ability to specify the maximum allowed number of permissible hyphenated lines to appear in sequence (not available with initial release of the new runoff).
- the ability to list or flag output lines that contain too much padding (not available with initial release of the new runoff).

Hyphenation Techniques

The most important hyphenation problem is, of course, how to hyphenate a word. Before describing how WORDPRO performs hyphenation, it may be interesting to describe how other word processing systems perform hyphenation. Three common techniques are described:

1. Hot Zone: Hyphenation is performed by the terminal operator. Usually an 8 to 10 character "hot zone" is defined at the right-hand margin of the terminal. As the system outputs formatted text, any word that starts before the hot zone and extends into the hot zone, causes the system to stop typing in midword as soon as the hot zone is reached. The terminal operator then allows typing to proceed character-by-character until a suitable hyphenation point is reached. The terminal operator then types a hyphen and returns the system to automatic output mode. The "hot zone" technique is used by many stand-alone word processing systems, for example, Honeywell/Base Ultra-Text, Redactron, Wang, etc.

2. Exception Dictionary: This hyphenation technique involves an arbitrarily complex set of hyphenation rules and a small online dictionary of correctly hyphenated words that would not be hyphenated satisfactorily by these rules. The first action performed when hyphenating a word is to search through the exception dictionary to ascertain whether or not the word exists there. If it does, the hyphenation for the word is taken from the dictionary. If the word does not appear in the exception dictionary then its hyphenation is determined by the hyphenation rules. Several user supplied hyphenate_word_ subroutines use the exception dictionary technique. A typical exception dictionary contains from 7,000 to 10,000 words. Documentation for the hyphenation rules average from 10 to 30 pages of text.

3. Total Dictionary: This technique is similar to the exception dictionary technique except that finding a word in the dictionary is the rule rather than the exception. No logical hyphenation rules are used. The Atex-8000 photocomposition system uses this technique. The dictionaries used contain from 80,000 to 150,000 words.

Multiple Dictionaries

WORDPRO will use a "multiple dictionary" hyphenation technique. It is similar to the "total dictionary" technique except that more than one dictionary may be used.

A user may dynamically specify an ordered list of dictionaries to be searched during hyphenation. This list of dictionaries is specified in the "dict" search list. (See MTB-336 for a description of the new Multics search facility.) The default "dict" search list will specify one standard WORDPRO dictionary. This dictionary will be a standard Multics product.

An important point about the multiple dictionary technique is that the dictionaries are searched in order. If the word being hyphenated is not found in the first dictionary searched, then the next dictionary in the list is searched. When a word is found in a dictionary it is hyphenated as specified in that dictionary. No further searching for that word is performed. If a word is not found in any of the specified dictionaries, then that word is not hyphenated.

The multiple dictionary technique offers the WORDPRO user viable solutions to the problems of hyphenation. A list of the most important advantages of this technique is presented below:

- Hyphenation is automatic. No terminal operator interaction is required. In situations where output is directed to a file, or the formatting of a document is performed by an absentee process, any requirement for online human interaction is unacceptable. It is also highly unlikely that any terminal operator can quickly and consistently make accurate hyphenation decisions.

- No complicated hyphenation rules are used. It is easy to understand how WORDPRO performs hyphenation. The WORDPRO hyphenation technique can be summarized in a few words:

  "A word is hyphenated as specified in the first dictionary in which it is found."

- Each user has complete control over hyphenation. Any Multics user may specify his own set of dictionaries. By adding a private dictionary to the beginning of the "dict" search list, a user may specify how words contained in that dictionary are hyphenated. If a user does not like the hyphenation specified for a word in the standard WORDPRO dictionary, then that word may be added to a private dictionary and hyphenated differently or not at all. If a user wants a word hyphenated that is not contained in the standard WORDPRO dictionary, then that word may be added to a private dictionary.

- The problem of homographs can be solved. (1) In the standard WORDPRO dictionary no hyphenation will be specified for homographs. The reason is that it is better not to hyphenate a word than to hyphenate it incorrectly. A user that wants a particular instance of a homograph hyphenated can specify the hyphenation within the document, the only place that the correct hyphenation is known. This can be done by using a runoff control to make external calls to the search list and dictionary commands. For example, calls can be made to add a temporary dictionary to the "dict" search list, to add the homograph to this dictionary, and then to delete the temporary dictionary from the "dict" search list after the homograph is processed.

---

(1) A homograph is a word with the same spelling as another but with different meaning, possibly different pronunciation, and possibly different hyphenation. For example, the verb re-sume (to begin again) and the noun re-su-me (a summing up).

- The multiple dictionary technique fits the needs of all users. For the average user that does not demand perfection, the standard WORDPRO dictionary alone is sufficient. No knowledge of dictionaries or search lists is required. Other users may have a second or third dictionary set up for use each time they log in and thus they also need not be concerned with how this technique works. However, other users may require absolute and dynamic control over hyphenation. For these users, absolute and dynamic control is available.

SPELLING

Spelling is the process of forming words by putting letters together. WORDPRO provides tools that identify, find, and correct words in a document that were put together or spelled incorrectly.

The ability to eliminate misspelled, mistyped, and unwanted words from a document provides WORDPRO users with the means to produce high quality documents.

Spelling Technique

The WORDPRO technique for identifying misspelled words involves making a list of the unique words in a document and then looking up in a dictionary each word in the list. If a word is found in a dictionary, it is deleted from the list. The result is a list of the words in the document that cannot be found in a dictionary.

The search for a word in a dictionary is performed in the same way as described for hyphenation. Multiple dictionaries can be used. As with hyphenation, the list of dictionaries to be used is specified in the "dict" search list. Thus the default dictionary used for spelling is the same standard WORDPRO dictionary used for hyphenation.

The multiple dictionary technique allows one or more private dictionaries to be used. If a user identifies a correctly spelled word that is not contained in the standard WORDPRO dictionary, then that word may be added to a private dictionary.

Unwanted Words

In addition to the need to identify misspelled words in a document, there is the need to identify unwanted words. An unwanted word may be some normally acceptable, correctly spelled word that should not appear in a particular document; at least not without a careful check of the context in which the word is used.

An example of an unwanted word is the word "basic" within Honeywell Multics documentation. This word, when used in computer documentation, may be confused with the "basic" command used to invoke the BASIC compiler. Thus every instance of this word in a Honeywell Multics document must be identified and approved.

The problem with unwanted words is that they are very likely contained in the standard WORDPRO dictionary and thus are deleted from any list of misspelled words. It is not acceptable to solve this problem by requiring a user to maintain a private dictionary containing all of the words in the standard WORDPRO dictionary except the few words that are unwanted.

Instead, it will be possible to add a word to a dictionary (presumably a private dictionary that is searched before the standard dictionary) and to specify that this word is unwanted and therefore should not be trimmed (deleted) from any list of misspelled words.

WORDPRO DICTIONARIES

The WORDPRO dictionaries that will be used for hyphenation and spelling will be implemented as indexed files and accessed via the "vfile" I/O module.

A dictionary file will contain no records. The words in the dictionary will be the index keys. The hyphenation and other control information (for example, the no-trim indicator) will be kept in the record descriptor associated with each key (word).

Implementation Advantages

The implementation of WORDPRO dictionaries as keyed vfiles allows WORDPRO users to take full advantage of this powerful and flexible Multics I/O interface. The most important advantages of this implementation are:

- Capacity: A WORDPRO dictionary could contain billions of words. The number of words in a dictionary is limited only by disk storage and quota. Dictionaries of one half million words are both possible and reasonable.

- Efficiency: Very large dictionaries can be used efficiently. The average cost (in processor time, memory usage, and paging) for a direct access of a word in an eight million word dictionary is only about 50% greater than the cost of a direct access of a word in a two hundred word dictionary. In addition, WORDPRO dictionary users will benefit from future vfile performance improvements.

- Access Control: It is possible to have complete control over who may use and who may update a dictionary.

- Sharing: Dictionaries may be shared by multiple concurrent users. A dictionary may even be updated (words added or deleted) at the same time it is used for hyphenation or spelling.

- Integrity: WORDPRO dictionaries are fully protected from damage during update. Any update operation that is interrupted (due to system failure or process termination) will be automatically completed or backed out. A dictionary is never left in an inconsistent state.

Standard WORDPRO Dictionary

In order to make hyphenation and spelling standard WORDPRO features, a WORDPRO dictionary will be provided as a standard Multics product. The system default for the "dict" search list will reference only the standard WORDPRO dictionary.

The initial version of the standard WORDPRO dictionary contains about 50,000 words. This dictionary was obtained from the Rome Air Force Data Center. The words in this dictionary will be hyphenated according to rules specified in the U. S. Government Printing Office Style Manual. Words identified as homographs will not be hyphenated. In general, no technical, scientific, medical, foreign language, or other specialized words will be included in this dictionary. However, all runoff control words (.un, .hy, etc.) will be included so that they will not be identified as unknown words in a runoff input segment.

It is hoped that later versions of this dictionary will contain significantly more words. A dictionary of 200,000 to 400,000 words is obtainable. It is also hoped that other dictionaries of specialized words can be obtained and offered to WORDPRO users.

WORDLIST SEGMENTS

A wordlist segment is a segment (with the entryname suffix ".wl") that contains an alphabetically ordered list of unique words. Words in a wordlist segment are separated by newline characters.

The format of a wordlist segment, and the commands that add and list words in a dictionary, have been designed so dictionaries may be updated from wordlist segments and wordlist segments may be generated from dictionaries.

Wordlist segments are used during spelling error detection. Most WORDPRO users will use a command that lists unknown words without explicitly using a wordlist segment. Thus, most WORDPRO users will never have to deal with wordlist segments. However, for advanced WORDPRO users, a full set of commands will be available to create, print, and trim wordlist segments.

SUMMARY OF COMMANDS

This section presents a summary of the WORDPRO commands that perform hyphenation, spelling, dictionary maintainence, and wordlist segment operations. The commands are grouped according to function.

Hyphenation

hyphenate_words	prints words with their hyphenation.
runoff	formats a file with hyphenation if the -hyphenate control argument is specified. (See MTB-337.)

Spelling

correct_words	replaces with a correction all occurrences of a given word in a text segment.
find_words	finds all occurrences of a given word in a text segment.
list_unknown_words	lists all words in a text segment that are <u>not</u> found in a selected set of WORDPRO dictionaries. These unknown words are likely to be misspelled, mistyped, or unwanted.

Dictionaries

add_dict_words	adds words to a WORDPRO dictionary.
delete_dict_words	deletes words from a WORDPRO dictionary.
list_dict_words	lists words in a WORDPRO dictionary.

Wordlist Segments

create_wordlist	creates a wordlist segment from a text segment.
print_wordlist	prints the words in a wordlist segment.
trim_wordlist	deletes all words in a wordlist segment except those words that are not found in a specified list of dictionaries.

MPM DOCUMENTATION

The remainder of this memorandum presents draft MPM documentation for the commands and subroutine that implement the proposed new WORDPRO hyphenation and spelling facilities.

Name: add_dict_words, adw

The add_dict_words command adds one or more words to a WORDPRO dictionary.

Usage

add_dict_words path {words} {-control_args}

where:

1. path
   is the pathname of the dictionary to which the words are added. If the entryname suffix ".dict" is not specified, then it is added. If the dictionary does not exist, then it is created.

2. words
   are words to add to the dictionary. At least one word is required unless -input_file is specified. If a word is already in the dictionary, then it is not replaced and a warning message is issued. (See the notes below for a description of the verification performed before a word is added to the dictionary, and for a description of how unwanted words and hyphenation are specified.)

3. -control_args
   can be chosen from the following:

   -brief, -bf
   suppresses the warning message usually given when the word cannot be added to the dictionary because it is already in the dictionary or because it contains invalid characters.

   -force, -fc
   allows a word already in the dictionary to be replaced. This feature may be used to change the capitalization, no-trim attribute, or hyphenation of a word in the dictionary.

   -input_file path, -if path
   adds to the dictionary words contained in the segment specified by path. Words in this segment should be separated by newlines.

add_dict_words

-no_verify, -nv
specifies that no verification is performed on words added to the dictionary.

Notes

Normally, only words composed entirely of lowercase letters are added to the dictionary. By design, the add_dict_words command does not add words containing uppercase letters, hyphens, apostrophes, numbers, and other nonlowercase characters. A warning message is issued for each word rejected because of this verification. This verification is performed to help users avoid filling a dictionary with "garbage" words. However, if a user wants to add such a word, he can use the -no_verify control argument.

Only one form of capitalization is allowed for each word in the dictionary. If a word is already in the dictionary, then an attempt to add this word with a different capitalization is treated as an attempt to replace the original word. The user should be aware of the effect capitalization has when searching for a word in a dictionary. (See the trim_wordlist command.)

It is possible to add a word to the dictionary and specify that the word not be trimmed (deleted) by the trim_wordlist or list_unknown_words commands. This no-trim attribute is denoted by preceding the word with the ASCII circumflex character "^" (Octal 136).

The correct hyphenation of a word can be specified when it is added to the dictionary. Imbedded hyphens indicate the hyphenation points. If no hyphenation points are specified, it is assumed that the word cannot be hyphenated.

If a word is spelled with a hyphen, then that hyphen must be preceded by another hyphen or an ASCII circumflex character. The character sequence "--" indicates that the word contains a hyphen and that hyphenation may be performed at (after) the hyphen. The character sequence "^-" indicates that the word contains a hyphen, but the word may not be hyphenated at the hyphen.

Examples

adw good_words test

The word "test" is added to the dictionary "good_words.dict" in the working directory. The word will not be hyphenated.

adw >udd>Project_id>Webster ex-am-ple

The word "example" is added to the dictionary "Webster.dict" in the user's project directory. It will be hyphenated at the specified hyphenation points.

adw [home_dir]>my_words ^bas-ic

The word "basic" is added to the dictionary "my_words.dict" in the user's home directory. It will be hyphenated, but it will not be trimmed (deleted) by the trim_wordlist or list_unknown_words commands.

adw good_words in--house Multics

No words are added to the dictionary "good_words.dict". The presence of an actual hyphen in the word "in-house" and the capital letter in the word "Multics" cause these words to be rejected.

adw good_words in--house Multics -no_verify

The use of the "no_verify" control argument permits the user to add these words to the dictionary. The word "in-house" will be hyphenated at the point of its actual hyphen.

adw good_words right^-hand -no_verify

The word "right-hand" is added to the dictionary "good_words.dict". The word will not be hyphenated.

adw good_words -input_file document.wl -brief

This command attempts to add to the dictionary "good_words.dict" all words in the wordlist segment "document.wl". Any words that are already in the dictionary are not added, but all warning messages are suppressed.

Name: correct_words, cow

The correct_words command replaces all occurrences of a given word within a specified text segment with a new word called the "correction". The user can specify more than one word to be corrected.

Usage

correct_words path word1 corr1 ... {wordn corrn} {-control_args}

where:

1. path
   is the pathname of the text segment.

2. wordi
   is a word in the text segment to be corrected.

3. corri
   is the correction, i.e., the replacement for wordi.

4. control_args
   can be chosen from the following:

   -brief, -bf
   suppresses printing of the number of corrections for each wordi.

   -from n, -fm n
   corrections are made in the text segment starting from the line number specified by n. If this control argument is not specified, then the text segment is processed starting from the first line.

   -header, -he
   prints the pathname of the text segment.

   -lines n, -li n
   for each correction made, the lines (and line numbers) starting n lines before, through n lines after the line containing the correction are printed. Thus if n is "1" three lines are printed.

   -long, -lg
   for each word corrected, the line (and line number) where the correction is made is printed.

-to n
corrections are made in the text segment up to and including the line number specified by n. If this control argument is not specified, then the text segment is processed to the last line.

-word word corr
specifies that the value following this control argument is a word to be corrected. This control argument can be used to correct a word that appears to be a control argument.

Notes

Words are found in the text segment in the same way as described for the create_wordlist command. (See the create_wordlist command.) Words are separated by white space characters and punctuation is removed. Underscored words are corrected. Numbers may also be corrected.

Examples

```
correct_words document.runoff typpoo typo
```

The misspelled word "typpoo" is replaced wherever it occurs in document.runoff by the word "typo". If there are two occurrences of "typpoo", then the command prints the message:

```
2 corrections for "typpoo"
```

Name: create_wordlist, cwl

The create_wordlist command produces an alphabetized list of all distinct words found in the specified text segment. This list is saved in a wordlist segment that is created in the working directory. The wordlist segment is created with the entryname of the text segment with the suffix ".wl" added. The total number of words in the text segment and the number of words put into the wordlist segment are printed.

Usage

create_wordlist path {-control_args}

where:

1. path
   is the pathname of the text segment.

2. control_args
   can be chosen from the following:

   -brief, -bf
   suppresses the printing of the total number of words in the text segment and the number of words put into the wordlist segment.

   -from n, -fm n
   words are processed in the text segment starting from the line number specified by n. If this control argument is not specified, then the text segment is processed starting from the first line.

   -header, -he
   prints the pathname of the text segment.

   -no_sort, -ns
   specifies that the words in the wordlist segment are not to be sorted into alphabetical order. They are put into the wordlist segment in the order in which they are found in the text segment and duplications are not eliminated. This control argument is intended for special application and should not be used for normal wordlist segment creation.

-number, -nb
specifies that words consisting of all numeric characters are not automatically trimmed (deleted) from the wordlist segment. (See the notes below.)

-to n
words are processed in the text segment up to and including the line number specified by n. If this control argument is not specified, then the text segment is processed to the last line.

Notes

Words longer than 95 characters are truncated to 94 characters with a question mark "?" appended.

Words in the text segment are separated by the following delimiter (white space) characters:

space
horizontal tab
vertical tab
newline
form feed
pad (octal 177)

Punctuation characters are removed from the word. The characters "([{ are removed from the left side of the word. The characters ")]}.,;:?! are removed from the right side of the word.

Additional special processing is performed on each word after all punctuation is removed. A summary of this special processing is given below:

- if the entire word is underscored, then the underscoring is removed. If only part of a word is underscored, then the underscores remain.
- if the word is a number, i.e., consists of all numeric characters or the special characters .,-+/ then the word is automatically trimmed (deleted) from the wordlist segment. The -number control argument disables the automatic trimming of numbers.

Examples

The table below shows examples of how punctuation is removed from a word and how special processing is performed.

WORD BEFORE PROCESSING	WORD IN WORDLIST
example	example
"example"	example
example.)	example
{example}	example
example	example
exam.ple	exam.ple
)example(	)example(
100	"trimmed"
1,000	"trimmed"
7-5.2	"trimmed"
+1	"trimmed"
1/2	"trimmed"
1A	1A
1(2)	1(2)

Name: delete_dict_words, ddw

The delete_dict_words command deletes one or more words from a WORDPRO dictionary.

Usage

delete_dict_words path {words} {-control_args}

where:

1. path
   is the pathname of the dictionary. If the entryname suffix ".dict" is not specified, then it is added.

2. words
   are words to be deleted from the dictionary. At least one word is required unless -input_file is specified. If a word is not found in the dictionary, then a warning message is issued.

3. control_args
   can be chosen from the following:

   -brief, -bf
   suppresses the warning message usually given when a word is not found in the dictionary.

   -input_file path, -if path
   deletes from the dictionary the words contained in the segment specified by path. Words in this segment should be separated by newlines.

Notes

A word to be deleted from the dictionary must be capitalized exactly the way it is capitalized in the dictionary. The word must not contain the special characters used by the add_dict_words command to specify the no-trim attribute and hyphenation points.

Examples

```
ddw >udd>Project_id>Webster example
ddw [home_dir]>my_words basic
ddw good_words test in-house Multics
```

Name: find_words, fiw

The find_words command finds all occurrences of a given word within a specified text segment. The user can specify more than one word to be found. For each occurrence of a given word within the text segment, the number of the line containing the word is printed.

Usage

find_words path {words} {-control_args}

where:

1. path
   is the pathname of the text segment.

2. words
   are words to be found in the text segment.

3. control_args
   can be chosen from the following:

   -from n, -fm n
   the text segment is searched starting from the line number specified by n. If this control argument is not specified, then the text segment is searched starting from the first line.

   -header, -he
   prints the pathname of the text segment.

   -input_file path, -if path
   finds the words contained in the segment specified by path. Words in this segment should be separated by newlines.

   -lines n, -li n
   for each occurrence of a given word, the lines (and line numbers) starting n lines before, through n lines after the line containing the word are printed. Thus if n is "1", three lines are printed.

   -long, -lg
   for each occurrence of a given word, the line (and line number) of that word is printed.

-to n
the text segment is searched up to and including the line number specified by n. If this control argument is not specified, then the text segment is searched to the last line.

-word word
specifies that the value following this control argument is a word to be found. This control argument can be used to find a word that appears to be a control argument.

Notes

Words are found in the text segment in the same way as described for the create_wordlist command. (See the create_wordlist command.) Words are separated by white space characters and punctuation is removed. Underscored words are found. Numbers may also be found.

Name: hyphenate_words, hyw

The hyphenate_words command hyphenates a word and prints the word with imbedded hyphens to indicate the hyphenation points. Double hyphens are used to denote the presence of an actual hyphen in the word. The user can specify more than one word to be hyphenated and printed.

Usage

hyphenate_words {words} {-control_arg}

where:

1. words
   are words to be hyphenated.

2. control_arg
   can be the following:

   -input_file path, -if path
   hyphenates the words contained in the segment specified by path. Words in this segment should be separated by newlines.

Notes

In order to perform the hyphenation, this command uses one or more WORDPRO dictionaries. The dictionaries used are specified in the "dict" search list. When searching for a word, capitalization is processed as described for the trim_wordlist command.

This command can be used to test the hyphenation of a word using the dictionaries currently defined in the "dict" search list.

Name: list_dict_words, ldw

The list_dict_words command prints a list of words in a WORDPRO dictionary.

Usage

list_dict_words path {words} {-control_args}

where:

1. path
   is the pathname of the dictionary to be listed. If the entryname ".dict" is not specified, then it is added.

2. words
   are words to be listed. If no words are specified, and if -input_file is not specified, then all words in the dictionary are listed.

3. control_args
   can be chosen from the following:

   -brief, -bf
   suppresses the warning message usually given when a word is not found in the dictionary.

   -input_file path, -if path
   lists of the words contained in the segment specified by path. Words in this segment should be separated by newlines.

   -raw
   prints the words without indicating the no-trim attribute or hyphenation points. If this control argument is not specified, then a ASCII circumflex character "^" (Octal 136) precedes each word listed that has the no-trim attribute, and hyphens are imbedded in each word listed in order to indicate the hyphenation points of the word.

Name: list_unknown_words, luw

The list_unknown_words command lists all words in the specified segment that are not found in a selected set of WORDPRO dictionaries. These unknown words are likely to be misspelled, mistyped, or unwanted.

Usage

list_unknown_words path {-control_args}

where:

1. path
   is the pathname of of the segment to be processed.

2. control_args
   can be chosen from the following:

   -from n, -fm n
   lists words in the text segment starting from the line number specified by n. If this control argument is not specified, then the text segment is processed starting from the first line.

   -number, -nb
   specifies that words consisting of all numeric characters are not automatically trimmed (deleted) from the list of unknown words. If this control argument is not specified, then no numbers appear in this list.

   -to n
   lists words in the text segment up to and including the line number specified by n. If this control argument is not specified, then the text segment is processed to the last line.

list_unknown_words

Notes

This command first produces an alphabetized list of all distinct words found in the specified segment. (See the create_wordlist command.) It then trims (deletes) all words in this list that are found in any WORDPRO dictionary specified in the "dict" search list. (See the trim_wordlist command.) The remaining words in the list are printed. If no words remain in the list (all of the words in the specified segment are found in the dictionaries) then the following message is printed:

"No unknown words."

Name: print_wordlist, pwl

The print_wordlist command arranges a wordlist segment (see the create_wordlist command) into a multiple column format suitable for either terminal printing or printing on a line printer.

Usage

print_wordlist path {-control_args}

where:

1. path
   is the pathname of a wordlist segment. If the entryname suffix ".wl" is not specified, then it is added.

2. control_args
   can be chosen from the following:

   -columns n, -cols n
   specifies that the output is to contain n columns where 1<=n<=6. (See the notes below.)

   -output_file path, -of path
   directs the output to the segment specified by path.

Notes

If the -output_file control argument is specified, the default number of columns is 6. Otherwise, the default number of columns is the maximum number of columns (not exceeding 6 that fit within the line length defined for terminal output. If no line length is defined for terminal output (for example, when file_output is used), the default number of columns is 3.

Words are arranged into columns 20 character positions wide. However, a word may extend beyond the end of a column up to a maximum of 31 character positions. If a word exceeds 31 character positions, it is truncated to 31 followed by a question mark "?". This implies that the rightmost column is actually 32 characters wide. Therefore, for a line length L, the maximum number of columns that can be accommodated is ((L-32)/20)+1.

Name: trim_wordlist, twl

The trim_wordlist command trims (deletes) all words in the specified wordlist segment that are found in any WORDPRO dictionary specified in the "dict" search list. The trimmed wordlist segment replaces the original wordlist segment. The number of words in the trimmed wordlist segment is printed.

Usage

trim_wordlist path {-control_args}

where:

1. path
   is the pathname of the wordlist segment to be trimmed. If the entryname suffix ".wl" is not specified, then it is added.

2. control_args
   can be chosen from the following:

   -brief, -bf
   suppresses printing of the number of words in the trimmed wordlist segment.

   -dictionary path, -dict path
   the WORDPRO dictionary specified by path is used instead of the WORDPRO dictionaries specified in the "dict" search list. This control argument may be specified more than once, thus multiple dictionaries may be used. They will be used in the order specified.

Notes

For each word processed, the dictionaries are searched in the order defined in the "dict" search list. Normally, when a word is found in a dictionary, it is trimmed. However, if the word found has the no-trim attribute, then the word is not trimmed and no more dictionaries are searched for this word.

When searching for a word in a dictionary, special processing of capital letters is performed. Most words in a dictionary consist of all lowercase letters. These words match any representations of themselves that are either all lowercase letters, all lowercase letters with a leading capital letter, or all capital letters. Words in a dictionary that have a leading capital letter only match representations of themselves that have a leading capital letter or are all capital letters. Words in a dictionary that consist of all capital letters or mixed lowercase and capital letters only match representations of themselves that have the identical capitalization.

The table below shows examples of different ways a word in a dictionary may be capitalized. It also shows which representations of these words match and which do not match.

WORD	MATCH	NO MATCH
example	example Example EXAMPLE	ExAmple
Multics	Multics MULTICS	multics MULTics
WORDPRO	WORDPRO	wordpro Wordpro WordPro
TermiNet	TermiNet	terminet Terminet TERMINET

Name: hyphenate_word_

The hyphenate_word_ subroutine determines where, if at all, a given word may be hyphenated in order to best fit the word into a given amount of space at the end of an output line.

Usage

```
dcl  hyphenate_word_ entry (char(*) unaligned, fixed bin(17),
     fixed bin(17, fixed bin(35));

call hyphenate_word_ (word, space, hpoint, code)
```

where:

1. word (Input)
   is the text word to be hyphenated.

2. space (Input)
   is the number of character positions remaining in the output line.

3. hpoint (Output)
   is the position of the character on the "left" of the selected hyphenation point. The word may be broken after this character and a hyphen appended. A value of 0 indicates the word may not be hyphenated.

4. code (Output)
   is a standard status code. A value of 0 indicates the hyphenation proceeded normally. The special value of error_table_$hyphen indicates the hyphenation point selected is at an actual hyphen character and thus no additional hyphen need be appended. All other values indicate an error accessing a WORDPRO dictionary.

Notes

In order to perform the hyphenation, this entry point uses one or more WORDPRO dictionaries. The dictionaries used are specified by the "dict" search list. When searching for a word, capitalization is processed as described for the trim_wordlist command.

In order for the hyphenation to be performed correctly, the caller should remove all unwanted punctuation characters from the beginning and end of the word being hyphenated.