NAME

mkkey – create keys (for indexing) from text files

SYNOPSIS

mkkey [**-C** *configfile*] [**-d** *delimiter*] [**-af** *argfile*] [**-c** *eigenfile*] [**-n** *eigenwords*] [**-l** *minwordlen*] [**-m** *maxwordlen*] [**-k** *maxkeys*] [**-w**] [**-s**] [**-a**] [**-V**] [**-?**] [**-HELP**] [**-D**] [**-v***[=level]] [***-i** *ignorechars]* [*file(s) ...*]

DESCRIPTION

Introduction

This program is similar to the MKEY program that is, or was, distributed with most UNIXes (back from the old V7 days), but this program can optionally (normally) pass all useful words through as keys instead of filtering out useful words like the original MKEY program did. This program takes many of the same argument options as the original but some of these options can also be specified in a configuration file.

Output

The output of this program are the keys extracted from the input text files and is normally written to the standard output of the program, unless the `-of outfile` option was used, in which case the output is written to that file specified. The output keys are by default written in key-file format unless the `-s` option was specified, in which case only the keys are written without the file-record tag prefix. Note that although the output key file is in ASCII (or ISO Latin-1) characters, the lines are generally too long to be viewed with any normal viewing program. The key file data is really only meant to be processed by other programs expecting key-file data.

OPTIONS

Several options can be given to specified the type of text input files expected and the nature of the keys extracted. Most of these are optional. By default the output keys (in key-file format) are written to standard output.

-C *configfile* This is used to specify a configuration file containing some options.

-af *argfile* This specifies the name of a file that serves as a list of input files to process. The *argfile* value can also be the minus ('-') character to specify standard input as the input file to be processed.

-of *outfile* Normally, the output (in key-file format) is written to standard output. This option redirects standard output to the file specified.

-d *delimiter* This is the most important option and must be specified for record-delimited files where records span more than a single input line (the default assumed). The value of the option should consist of those characters that will be taken to be a line that separates records. A blank line should be specified with a delimiter consisting of a single space (remember to quote it from the shell if necessary). Other common delimiters are the percent ('%') character that is used for "cookie" files or one or more pound ('#') characters used for other common record-oriented files.

-w This option specifies that each whole input file should be considered a single record. The `-d delimiter` option is ignored if this option is present.

-c *eigenfile* By default, the program searches some standard locations for an EIGEN file (a file of the most commonly used English words) and uses that by default. If this option is used, the specified file is used as the EIGEN file instead.

-n *eigenwords* By default, a maximum of ten thousand eigenwords are considered for key filtering. This option can change the number of eigenwords considered.

-l *minlen* This option specifies the minimum length of a word to be used as a key. The default is three. Keys that would have less characters than the minimum are ignored.

- m *maxlen* This option specifies the maximum length of a word to be used as a key. The default is six. Keys that have more characters than the maximum in effect are truncated to the maximum. This maximum has implications for what keys can be queried for when a database is eventually prepapred. Query keys should always be at least as long as this maximum and are only allowed to be shorter than the actual word in the original input text if prefix matching is used during the query operation.

- k *maxkeys* This option specifies the maximum number of keys to be created for each record in the input file(s). The default is all keys found that match the other criteria for key selection. Note that if a maximum number of keys is specified for a record then queries may take longer time since more false hits will need to be eliminated with actual key traversals rather than though the faster database joining of the other keys that were used for indexing.

- s This option specifies that tag labels should be removed from the key- file output. The result is that the keys produced do not have tag labels identifying which original text record they were from. This option is provied for flexibility but all standard database creation mechanisms require the tag labels to be present in the key file.

- a This specifies that output keys (in key- file format) should be appended to an existing key file. This option only needs to be used (if desired) when the `-of out-file` option is also specified.

- V This option writes the version of the program on standard error and then has the program exit without any further processing.

- HELP This option searches for a help file for the present program and prints it only standard output. The program then exits without any additional processing.

OPERANDS

All of the invocation operands constitute source text input files for which keys are to be extracted.

files(s) All other positional arguments are taken to be source text input files to be processed. The keys for all files are concatenated into the output (in key- file format). An input file specified as a single minus character ('- ') is taken to mean standard input.

USAGE

EXAMPLES

☞ extracts keys from the file `test.txt` and put they keys into file `test.key` :

```
mkkey test.txt > test.key
```

ENVIRONMENT

MKKEY_PROGRAMROOT
this is used to find the program root

LOCAL this is used to find the program root if all of above are not available

PROGRAMROOT
this is used to find the program root if all of above are not available

MKKEY_NAME this is used as the program search- name if none is specified as an invocation argument

FILES

EXIT STATUS

0	program completed successfully
!=0	program encountered an error in processing

ATTRIBUTES

SEE ALSO

mkinv(1), mkquery(1), mkey(1), inv(1), hunt(1), lineindex(1)

DIAGNOSTICS

PATH TO

This program is currently located in `/usr/add-on/local/bin` or possibly where ever "local" programs are stored on your system. This is often at `${LOCAL}/bin` on some systems.

NOTES

CAVEATS

Be careful to put at one space between all option key letters and the associated key letter parameter.

ACKNOWLEDGEMENTS

The idea for this program was taken from the original UNIX `mkey(1)` program.

AUTHOR

David A.D. Morano