

NAME

`gperf` – generate a perfect hash function from a key set

SYNOPSIS

`gperf` [*OPTION*]... [*INPUT-FILE*]

DESCRIPTION

GNU `'gperf'` generates perfect hash functions.

If a long option shows an argument as mandatory, then it is mandatory for the equivalent short option also.

Output file location:

`--output-file=FILE` Write output to specified file.

The results are written to standard output if no output file is specified or if it is `-`.

Input file interpretation:

`-e, --delimiters=DELIMITER-LIST`

Allow user to provide a string containing delimiters used to separate keywords from their attributes. Default is `" , "`.

`-t, --struct-type`

Allows the user to include a structured type declaration for generated code. Any text before `%%` is considered part of the type declaration. Key words and additional fields may follow this, one group of fields per line.

`--ignore-case`

Consider upper and lower case ASCII characters as equivalent. Note that locale dependent case mappings are ignored.

Language for the output code:

`-L, --language=LANGUAGE-NAME`

Generates code in the specified language. Languages handled are currently C++, ANSI-C, C, and KR-C. The default is ANSI-C.

Details in the output code:

`-K, --slot-name=NAME`

Select name of the keyword component in the keyword structure.

`-F, --initializer-suffix=INITIALIZERS`

Initializers for additional components in the keyword structure.

`-H, --hash-function-name=NAME`

Specify name of generated hash function. Default is `'hash'`.

`-N, --lookup-function-name=NAME`

Specify name of generated lookup function. Default name is `'in_word_set'`.

`-Z, --class-name=NAME`

Specify name of generated C++ class. Default name is `'Perfect_Hash'`.

`-7, --seven-bit`

Assume 7-bit characters.

`-l, --compare-lengths`

Compare key lengths before trying a string comparison. This is necessary if the keywords contain NUL bytes. It also helps cut down on the number of string comparisons made during the lookup.

`-c, --compare-strncmp`

Generate comparison code using `strncmp` rather than `strcmp`.

`-C, --readonly-tables`

Make the contents of generated lookup tables constant, i.e., readonly.

- E, --enum**
Define constant values using an enum local to the lookup function rather than with defines.
- I, --includes**
Include the necessary system include file <string.h> at the beginning of the code.
- G, --global-table**
Generate the static table of keywords as a static global variable, rather than hiding it inside of the lookup function (which is the default behavior).
- P, --pic**
Optimize the generated table for inclusion in shared libraries. This reduces the startup time of programs using a shared library containing the generated code.
- Q, --string-pool-name=NAME**
Specify name of string pool generated by option **--pic**. Default name is 'stringpool'.
- null-strings**
Use NULL strings instead of empty strings for empty keyword table entries.
- constants-prefix=PREFIX**
Specify prefix for the constants like TOTAL_KEYWORDS.
- W, --word-array-name=NAME**
Specify name of word list array. Default name is 'wordlist'.
- length-table-name=NAME**
Specify name of length table array. Default name is 'lengthtable'.
- S, --switch=COUNT**
Causes the generated C code to use a switch statement scheme, rather than an array lookup table. This can lead to a reduction in both time and space requirements for some keyfiles. The COUNT argument determines how many switch statements are generated. A value of 1 generates 1 switch containing all the elements, a value of 2 generates 2 tables with 1/2 the elements in each table, etc. If COUNT is very large, say 1000000, the generated C code does a binary search.
- T, --omit-struct-type**
Prevents the transfer of the type declaration to the output file. Use this option if the type is already defined elsewhere.

Algorithm employed by gperf:

- k, --key-positions=KEYS**
Select the key positions used in the hash function. The allowable choices range between 1-255, inclusive. The positions are separated by commas, ranges may be used, and key positions may occur in any order. Also, the meta-character '*' causes the generated hash function to consider ALL key positions, and \$ indicates the "final character" of a key, e.g., \$,1,2,4,6-10.
- D, --duplicates**
Handle keywords that hash to duplicate values. This is useful for certain highly redundant keyword sets.
- m, --multiple-iterations=ITERATIONS**
Perform multiple choices of the **-i** and **-j** values, and choose the best results. This increases the running time by a factor of ITERATIONS but does a good job minimizing the generated table size.
- i, --initial-asso=N**
Provide an initial value for the associate values array. Default is 0. Setting this value larger helps inflate the size of the final table.
- j, --jump=JUMP-VALUE**
Affects the "jump value", i.e., how far to advance the associated character value upon collisions. Must be an odd number, default is 5.

-n, --no-strlen

Do not include the length of the keyword when computing the hash function.

-r, --random

Utilizes randomness to initialize the associated values table.

-s, --size-multiple=N

Affects the size of the generated hash table. The numeric argument N indicates "how many times larger or smaller" the associated value range should be, in relationship to the number of keys, e.g. a value of 3 means "allow the maximum associated value to be about 3 times larger than the number of input keys". Conversely, a value of 1/3 means "make the maximum associated value about 3 times smaller than the number of input keys". A larger table should decrease the time required for an unsuccessful search, at the expense of extra table space. Default value is 1.

Informative output:**-h, --help**

Print this message.

-v, --version

Print the gperf version number.

-d, --debug

Enables the debugging option (produces verbose output to the standard error).

AUTHOR

Written by Douglas C. Schmidt and Bruno Haible.

REPORTING BUGS

Report bugs to <bug-gperf@gnu.org>.

COPYRIGHT

Copyright © 1989-2017 Free Software Foundation, Inc. License GPLv3+: GNU GPL version 3 or later <<http://gnu.org/licenses/gpl.html>>

This is free software: you are free to change and redistribute it. There is NO WARRANTY, to the extent permitted by law.

SEE ALSO

The full documentation for **gperf** is maintained as a Texinfo manual. If the **info** and **gperf** programs are properly installed at your site, the command

info gperf

should give you access to the complete manual.