NAME

bibsort − sort a BibTeX bibliography file

SYNOPSIS

bibsort [**−byday** or **−byvolume** or **−byyear**] [optional **sort**(1) switches] < infile >outfile
or
bibsort [**−byday** or **−byvolume** or **−byyear**] [optional **sort**(1) switches] BibTeXfile(s) >outfile

DESCRIPTION

bibsort filters a BibTeX bibliography, or bibliography fragment, on its standard input, printing on standard output a sorted bibliography.

Sorting is normally by BibTeX citation label name, or by *@String* macro name, and letter case is always ignored in the sorting.

OPTIONS

Except for the switches described below, command-line words beginning with a hyphen are assumed to be options to be passed to **sort**(1).

All remaining command-line words are assumed to be input files. Should such a filename begin with a hyphen, it must be disguised by a leading absolute or relative directory path, e.g. */tmp/-foo.bib* or *./-foo.bib*.

The **sort**(1) **−f** switch to ignore letter case differences is always supplied. The **−r** switch reverses the order of the sort. The **−u** switch removes duplicate bibliography entries from the input stream; however, such entries must match exactly, including all white space.

−byday This switch is intended for use with bibliographies of publications containing day, month, and year data, such as technical reports, newspapers, and magazines. It causes entries to be sorted by year, month, day, and citation label, so that the entries appear in their original publication order.

With -byday sorting, a day keyword is recognized (it will be standard in BibTeX 1.0), but for backward compatibility, month entries of the form

"daynumber " # monthname
"daynumber~" # monthname
{daynumber } # monthname
{daynumber~} # monthname
monthname # "daynumber "
monthname # "daynumber~"
monthname # {daynumber }
monthname # {daynumber~}

are also recognized, and will yield both a day and a month. If a day number is not available, 99 is assumed, which will sort the entry after others that have day values in the same year and month.

−byvolume This switch is intended for use with bibliographies of single journals. It causes entries to be sorted by journal, year, volume, number, page, year, and citation label, so that the entries appear in their original publication order. The journal name is included in the sort key, so that in a bibliography with multiple journals, output entries for each journal are kept together.

With **−byvolume** sorting, warnings are issued for any entry in which any of these fields are missing, and a value of the missing field is supplied that will sort higher than any printable value.

Because **−byvolume** sorting is first on journal name, it is essential that there be only one form of each journal name; the best way to ensure this is to always use @String{...} abbreviations for them. Order **-byvolume** is convenient for checking a bibliography against the original journal, but less convenient for a bibliography user.

−byyear If this switch is given, then the entry year value is prefixed to the sort key, so that sorting is first by year, then by citation label. This is useful for keeping a bibliography in approximate chronological order, ordered by citation label within each year.

BIBTEX FILE PARTS

The input stream is conceptually divided into five parts, any of which may be absent.

1. Introductory material such as comments, file headers, and edit logs that are ignored by BIBTEX. No line in this part begins with an at-sign, ‘‘@’’.
2. Preamble material delineated by ‘‘@Preamble{’’ and a matching closing ‘‘}’’, intended to be processed by TEX. Normally, there is only one such entry in a bibliography file, although BIBTEX, and **bibsort**, permit more than one.
3. Macro definitions (abbreviations) of the form ‘‘@String{...}’’. Any single @String specification may span multiple lines, and there are usually several such definitions.
4. Bibliography entries such as ‘‘@Article{...}’’, ‘‘@Book{...}’’, ‘‘@InProceedings{...}’’, and so on, provided that their citation labels have not already been encountered in a *crossref* assignment in a preceding entry. For **bibsort**, any line that begins with an ‘‘@’’ followed by letters and digits and an open brace is considered to be such an entry. Optional spaces and tabs may surround the ‘‘@’’, and precede the first open brace; these spaces and tabs will be deleted from the output to help standardize the appearance.
5. ‘‘@Proceedings{...}’’ bibliography entries, which are likely to be cross-referenced by ‘‘@InProceedings{...}’’ entries, and any other bibliography entries for which a crossref assignment was met before the entry itself.

An unfortunate implementation limitation of the current BIBTEX requires cross-referenced entries to appear *after* all other entries that cross-reference them, although this limitation works to the advantage of **bibsort**, allowing single-pass processing.

The order of these parts is preserved in the output stream. Part 1 will be unchanged, but parts 2–5 will be sorted within themselves.

The sort key of ‘‘@Preamble’’ entries is their initial line, of ‘‘@String’’ entries, the abbreviation name. For all other BIBTEX entries, the sort key is citation label between the open curly brace and the trailing comma, unless the sort key is prefixed with additional fields as requested by **−byvolume** or **−byyear** switches.

bibsort will correctly handle UNIX files with LF line terminators, as well as IBM PC DOS files with CR LF line terminators; the essential requirement is that input lines be delineated by LF characters. Thus, files from the Apple Macintosh, which uses bare CR to terminate lines, would first have to be converted to UNIX or PC DOS line format before giving them to **bibsort**.

CAVEATS

BIBTEX has loose syntactical requirements that the current simple implementation of **bibsort** does not support. In particular, outer parentheses may *not* be used in place of braces following ‘‘@keyword’’ patterns. If you have such a file, you can use **bibclean**(1) to prettyprint it into a form that **bibsort** can handle successfully.

The user must be aware that sorting a bibliography is not without peril, for at least these reasons:

1. BIBTEX has a requirement that entry labels given in *crossref = label* pairs in a bibliography entry *must* refer to entries defined *later*, rather than earlier, in the bibliography file. This regrettable implementation limitation of the current (pre-1.0) BIBTEX prevents arbitrary ordering of entries when *crossref* values are present. To partially solve this problem, **bibsort** will place ‘‘@Proceedings’’ entries last, since they are frequently cross-referenced by ‘‘@InProceedings’’ entries. However, it is also possible for ‘‘@Book’’, ‘‘@InBook’’, and ‘‘@InCollection’’ entries to cross-reference ‘‘@Book’’ entries, and for ‘‘@Article’’ entries to cross-reference other ‘‘@Article’’ entries. Neither of these cases are dealt with by **bibsort**, except that ‘‘@Book’’ entries that contain a ‘‘booktitle’’ assignment, and entries that are explicitly cross-referenced before their definition, are sorted with ‘‘@Proceedings’’,

2. If the BibTeX file contains interspersed commentary between "@keyword{...}" entries, this material will be considered part of the *preceding* entry, and will be sorted with it. Leading commentary is more common, and will be moved elsewhere in the file.

   This is normally not a problem for the part 1 material before the "@Preamble", since it is kept together at the beginning of the output stream.

3. Some kinds of bibliography files should be kept in a different order than alphabetically by citation labels. Good examples are a bibliography file with the contents of a journal, or a personal publication list, for both of which chronological publication order is likely to be preferred.

While a much more sophisticated implementation of **bibsort** could deal with the first point, and the **−byvolume** switch provides a partial solution to the third point, in general, a satisfactory solution requires human intelligence and natural language understanding that computers lack.

bibsort uses octal ASCII control characters 001 through 007, 0177, and 0377, for temporary modifications of the input stream. If any of these are already present in the input, they will be altered on output. This is unlikely to be a problem, because those characters have neither a printable representation, nor are they conventionally used to mark line or page boundaries in text files.

PROGRAMMING NOTES

Some text editors permit application of an arbitrary filter command to a region of text. For example, in GNU **emacs**(1), the command *C-u M-x shell-command-on-region*, or equivalently, *C-u M-*|, can be used to run **bibsort** on a region of the buffer that is devoid of cross references and other material that cannot be safely sorted.

Some implementations of BibTeX editing support in GNU **emacs**(1) have a *sort-bibtex-entries* command that is functionally similar to **bibsort**. However, the file size that can be processed by **emacs**(1) is limited, while **bibsort** can be used on arbitrarily large files, since it acts as a filter, processing a small amount of data at a time. The sort stage needs the entire data stream, but fortunately, the UNIX **sort**(1) command is clever enough to deal with very large inputs.

The current implementation of **bibsort** follows the UNIX tradition of combining simple already-available tools. A six-stage pipeline of **egrep**(1), **nawk**(1), **sort**(1), and **tr**(1) accomplishes the job in one pass with about 500 lines of heavily-commented shell script, about 225 lines of which is a **nawk**(1) program for insertion of sort keys. The initial prototype of **bibsort** was written and tested on several large bibliographies in a couple of hours, and after considerable use, was later extended with advanced sorting capabilities and cross-reference recognition in a couple of days of work. By contrast, **bibtex**(1) is more than 11 000 lines of code and documentation, and **bibclean**(1) is more than 15 000 lines long; both took months to develop, implement, and test.

BUGS

bibsort may fail on some UNIX systems if their **sort**(1) implementations cannot handle very long lines, because for sorting purposes, each complete bibliography entry is temporarily folded into a single line. You may be able to overcome this problem by adding a **−z***nnnnn* switch to the **sort**(1) command (passed via the command line to **bibsort**) to increase the maximum line size to some larger value of *nnnnn* bytes. According to their documentation, some UNIX **sort**(1) implementations require a space after **−z**, others forbid it, and still others do not support it at all. If a space is required, you must quote the pair, to prevent the *nnnnn* value from being interpreted as a filename by **bibsort**.

SEE ALSO

bibcheck(1), **bibclean**(1), **bibdup**(1), **bibextract**(1), **bibjoin**(1), **biblabel**(1), **biblex**(1), **biborder**(1), **bibparse**(1), **bibtex**(1), **bibunlex**(1), **citesub**(1), **egrep**(1), **emacs**(1), **nawk**(1), **sort**(1), **tr**(1).

AUTHOR

Nelson H. F. Beebe, Ph.D.
Center for Scientific Computing
Department of Mathematics
University of Utah

Salt Lake City, UT 84112
Tel: +1 801 581 5254
FAX: +1 801 581 4148
Email: <beebe@math.utah.edu>
WWW URL: http://www.math.utah.edu/~beebe