NAME

humsed — stream editor for Humdrum inputs

SYNOPSIS

humsed [**-E**] '*sed_instruction[;sed_instruction;...]*' [*inputfile ...*] [> *outputfile*]
humsed [**-E**] [**-f** *scriptfile*] [*inputfile ...*] [> *outputfile*]

DESCRIPTION

The **humsed** command provides a stream-editor for Humdrum data records. A stream-editor is a non-interactive editor that automatically processes a given input according to a user-specified set of editing instructions. Possible editing operations include substitution, deletion, transliteration, file-read, and file-write. By default the output is sent to the display, however the output can be routed to a file using file redirection (> *outputfile).*

The **humsed** command is fashioned after the UNIX **sed** stream editor. In contrast to **sed, humsed** editing instructions are applied only to Humdrum data records; Humdrum *interpretations* and *comments* are not affected by **humsed.**

The **humsed** command accepts one or more **sed** instructions. Instructions are specified on the command-line within a pair of single quotes. Where more than one editing instruction is specified, successive instructions are separated by a semicolon. Alternatively, instructions may be executed from a *scriptfile* using the **-f** option. If instructions are provided both on the command-line as well as via a *scriptfile,* the command line instructions are performed prior to the *scriptfile* instructions.

Permissible instructions include **s** for substitution, **y** for transliteration, **d** for deletion, **i** for insertion, **a** for append, **r** for file-read, and **w** for file-write. Each instruction may be preceded by an optional *regular expression* that limits the scope of the editing instruction only to those data records matching the regular expression. For example, the user may replace all occurrences of 'X' with 'Y' — provided the signifier 'Z' also occurs in the same data record. In the case of the delete **(d)** instruction, failing to specify a preceding regular expression will result in the deletion of all data records in the input.

For further information concerning the syntax and use of **humsed** editing instructions, refer to the documentation for the UNIX **sed** command.

OPTIONS

The **humsed** command provides the following options:

-h	displays a help screen summarizing the command syntax
-E	invoke Extended Regular Expression syntax
-f *scriptfile*	execute editing instructions from the file *scriptfile*

Options are specified in the command line.

With the **-E** option, **humsed** invokes the "extended" regular expression syntax, rather than the normal or "basic" regular expression syntax. With extended regular expressions, the following additional operations are supported: one-or-more (+), zero-or-one (?), logical OR (|), precedence grouping ( ), and alphanumeric token start and end anchors < >.

Note that not all systems support extended regular expressions for the **sed** command; on such systems the **-E** option for **humsed** is ineffective and may result in an error.

The **-f** options allows the user to specify a *scriptfile* that contains a set of editing instructions. Instructions in *scriptfile* are executed after any command-line editing scripts.

EXAMPLES

The following examples illustrate the substitution, transliteration, deletion, file-read and file-write instruction provided by **humsed.**

Simple substitution:

```
humsed 's/A/X/g' ragtime
```

The above command replaces the upper-case letter A by the upper-case letter X. Without the `g` (global) modifier, only the first occurrence of an "A" in each data record would be modified. The use of `g` applies the substitution instruction to *all* occurrences in a data record.

Substitution commands can be preceded by another regular expression that limits the selection of records that are affected by the substitution. For example, the following command eliminates all measure numbers in a `**kern` representation:

```
humsed '/=/s/[0-9]*//g' jellyroll
```

Rather than simply eliminating all numerical data, the initial regular expression `(/=/)` limits the substitution operation to those data records contain the `**kern` barline signifier `(=)`.

More complicated substitutions may involve compound (two or more) instructions. Instructions are separated by a semicolon, and are executed in succession for each data record. Consider the following command:

```
humsed 's/4[A-G]/8&/g;s/84/8/g' chicago > fastbass
```

This command changes all quarter-note pitches (in a `**kern` representation) below middle

C to eighth-note durations, while leaving quarter-notes above middle C unchanged. The first substitution instruction (`s/4[A-G]/8&/g`) searches for all strings beginning with the number 4, followed by one of the upper-case letters A to G. It then prepends the number 8; thus the token 4F will be replaced by 84F. (Note that the ampersand (&) in the substitution denotes the matched string found by the target regular expression.) The second substitution (`s/84/8/g`) replaces the string 84 by the string 8. In short, tokens such as 4F and 4CC# will be modified to 8F and 8CC# respectively — whereas tokens such as 2F and 4cc# will remain unmodified.†

The transliteration instruction (**y**) provides a short-cut for multiple single-character substitutions. For example, the following command replaces A with 0, B with 1, C with 2, etc. for the letters A to J:

```
humsed 'y/ABCDEFGHIJ/0123456789/' dixieland
```

Substitutions are organized by mapping each element in the first character string with the corresponding element in the second string. The first and second character strings must contain the same number of characters.

The delete instruction is preceded by a regular expression, followed by the single letter **d.** The following command deletes all data records containing the lower-case letter "r".

```
humsed '/r/d' swing
```

The file-write instruction (**w**) provides a way of copying selected material to a specified output file. Consider the following command:

```
humsed '/;/w pauses' bigband
```

This command identifies all data records in the file "bigband" that contain a semicolon (the `**kern` pause signifier) and copies them into the file "pauses." Recall that **humsed** operates only on Humdrum data records, so the **w** command will cause only data records to be outputted. Hence the resulting file "pauses" will not be a valid Humdrum file. (If the user wishes the extracted material to be in a valid Humdrum format, this could be done using the Humdrum **yank** command:`yank -m ';' 0 bigband > pauses`.)

The **humsed** command can also be used to read (**r**) material from a specified file whenever a certain condition occurs in the input stream. For example, the following command could be used to search for `**kern` pause signifiers (;) and add a global comment indicating the presence of a pause.

```
humsed '/;/r comment' bebop
```

---

† Note that this command is inadequate if 24th notes (thirty-second note triplets) are present in the input — since they will be transformed to 28th notes.

— where the file "comment" contains the following global comment:

```
!! A pause.
```

PORTABILITY

Any system supporting the UNIX-style **sed** command. Note that the **-E** option is a non-POSIX extension currently supported only by the MKS toolkit. It is hoped that in the future, other systems will support extended regular expression syntax for **sed.**

SEE ALSO

awk (UNIX), **regexp** (4), **regexp** (6), **rid** (4), **sed** (UNIX), **vi** (UNIX), **yank** (4)

WARNINGS

In the process of modifying text, it is possible to transform inadvertently Humdrum data records into interpretation records or comments. Particular caution should be exercised when inserting exclamation marks or asterisks.

In addition, it is possible to disrupt the spine structure by inserting or deleting tabs. Substitutions may result in empty lines or extra spaces that render the file no longer consistent with the Humdrum syntax.

LIMITS

Note that the extended regular expression mode (**-E**) is not available on most UNIX systems.