NAME

strophe — selectively extract strophic data

SYNOPSIS

strophe [**-s** *strophe_list* | **-x** *strophe_label*] [*inputfile ...*]

DESCRIPTION

The **strophe** command is used to isolate or extract selective strophic data. Strophic data encode alternative concurrent information for a given passage. Examples of alternative concurrent information might include the texts for different verses of a song, alternative renditions of the same passage (such as *ossia* passages), or differing editorial interpretations of a given note or sequence of notes.

The **strophe** command permits the user to extract selected information paths (called *strophes*) present in a Humdrum input.

Structurally, strophic data must begin from a single common spine, split apart into two or more alternative spines, and then rejoin to form a single spine. Since the strophes split from a common spine, all strophic data necessarily begin by sharing the same data type. Different exclusive interpretations may be introduced in the strophic passage — provided all strophic spines end up sharing the same data type just prior to being rejoined.

The beginning of a strophic passage is signalled by the presence of a *strophic passage initiator* — a single asterisk followed by the keyword "strophe" (`*strophe`). The end of a strophic passage is signalled by the *strophic passage terminator* — a single asterisk followed by the upper-case letter 'S' followed by a minus sign (`*S-`). Each spine within the strophic passage begins with a *strophe label* and ends with a *strophe end indicator* (`*S/fin`).

Strophe labels may consist of either alphanumeric names, or numbers. Numerical labels should be used when the strophic data imply some sort of order, such as verses in a song. Alphanumeric labels are convenient for distinguishing different editions or *ossia* passages. The following example encodes a melodic phrase containing four numbered verses from "Das Wandern" from *Die Schöne Müllerin* by Schubert.

```
!! Franz Schubert, 'Das Wandern' from "Die Schoene Muellerin"
**kern          **text
*>[V,V,V,V]     *>[V,V,V,V]
*>V             *>V
*k[b-e-]        *Deutsch
*               *solo
*               *strophe
*               *^
*               *^              *^
*               *S/1            *S/2        *S/3        *S/4
8f              Das             Vom         Das         Die
=5              =5              =5          =5          =5
8f              Wan-            Was-        sehn        Stei-
8b-             -dern           -ser        wir         -ne
8a              ist             ha-         auch        selbst ,
8ee-            des             -ben        den         so
=6              =6              =6          =6          =6
(16dd           Mül-            wir's       Rä-         schwer
)16ff           |               |           |           |
(16dd           -lers           ge-         -dern       sie
)16b-           |               |           |           |
8f              Lust ,          -lernt ,    ab ,        sind ,
8dd             das             vom         den         die
=7              =7              =7          =7          =7
(8.cc           Wan-            Was-        Rä-         Stei-
)16a            |               |           |           |
8b-             -dern !         -ser !      -dern !     -ne !
8r              %               %           %           %
*               *S/fin          *S/fin      *S/fin      *S/fin
*               *v              *v          *v          *v
*               *S-
*-              *-
```

Notice that this file contains a single section labelled 'V' (verse) and that an expansion list occurs near the beginning of the file that indicates the section is to be repeated 4 times in total.

The strophic passage in the above example pertains only to the spine marked `**text`. Following the strophic passage indicator (`*strophe`), the spine is split apart until the required number of verses are generated. Then each spine is labelled with its own strophe label. Since the verses have an order, it is appropriate to label them with numbers:`*S/1`, `*S/2`, and so on. The individual verses are terminated with strophe end indicators (`*S/fin`), the spines rejoin, and then a strophic passage terminator (`*S-`) marks the end of the strophic passage.

Executing the command:

```
strophe -s 4
```

produces the following output:

```
!! Franz Schubert, 'Das Wandern' from "Die Schoene Muellerin"
**kern          **text
*>[V,V,V,V]     *>[V,V,V,V]
*>V             *>V
*k[b-e-]        *Deutsch
*               *solo
8f              Die
=5              =5
8f              Stei-
8b-             -ne
8a              selbst ,
8ee-            so
=6              =6
(16dd           schwer
)16ff           |
(16dd           sie
)16b-           |
8f              sind ,
8dd             die
=7              =7
(8.cc           Stei-
)16a            |
8b-             -ne !
8r              %
*-              *-
```

Strophic encodings are nearly always encoded in *abbreviated* rather than *through-composed* file formats. Abbreviated encodings employ *section labels* and *expansion-lists* in order to identify how various passages are repeated and ordered.

When extracting a single strophe, either the abbreviated or through-composed versions can be used as input. However, when using the **strophe** command to select more than one strophe for output, it is essential that the input first be expanded to a through-composed version, via the **thru** command. For example, in order to select the first and third verses in the above passage by Schubert, the user would need to execute the following command pipeline:

```
thru wandern | strophe -s 1,3
```

The list following the **-s** option can contain individual strophes separated by commas. For example, the following command extracts verses 1, 3 and 4 in succession:

```
thru wandern | strophe -s 1,3,4
```

Strophes may also be output in non-numeric order as in the following command invocation:

```
thru wandern | strophe -s 4,3,2,1
```

If the **-x** option is invoked, **strophe** outputs only a single strophe whose string *label* is specified as an option. Strophe names need not be numerical. E.g.

```
strophe -x ossia
```

If the **strophe** command is invoked without any option, then all strophes are expanded in the output in numerical order beginning with strophe 1. Missing numerical strophes (such as a missing strophe `S/3` in a four-strophe encoding) will cause an error to be generated and terminate the **strophe** command.

Note that the **strophe** command allows strophe numbers containing a single decimal point, such as strophe `*S/4.2`. Having extracted the verse `*S/1`, the **strophe** command will output verse `*S/1.1` in preference to `*S/2` — if the decimal strophe is present. This feature allows more than one strophic passage to be encoded within a single abbreviated format file. This feature might prove useful, for example, in a musical work that contains a brief refrain in the middle of each verse.

The various strophe-related tandem interpretations are summarized below:

*strophe	strophic passage initiator
*S-	strophic passage terminator
*S/*string*	strophe name label
*S/*n[.n]*	strophe number label
*S/fin	strophe end indicator

Types of Strophe Interpretations

Note that for each strophic passage, all strophe labels must appear on the same record. See EXAMPLES below.

OPTIONS

The *strophe* command provides the following options:

-h	displays a help screen summarizing the command syntax
-s *strophe_list*	output numbered strophes according to *strophe_list*
-x *strophe_label*	output only strophes labelled *strophe_label*

Options are specified in the command line.

EXAMPLES

The following example is concocted to illustrate the operation of the **strophe** command. Consider the following Humdrum input:

```
!! 'strophe' example.
**example          **bar
*>[A,V,V,Coda]     *>[A,V,V,Coda]
*>A                *>A
A                  i
*>V                *>V
*                  **foo
*                  *strophe
*                  *^
*                  *S/1            *S/2
B                  1               2
*                  *S/fin          *S/fin
*                  *v              *v
*                  *S-
*                  **bar
C                  refrain
*                  *strophe
*                  *^
*                  *S/1.1          *S/2.1
B                  1               2
*                  *S/fin          *S/fin
*                  *v              *v
*                  *S-
*>Coda             *>Coda
*                  **foo
E                  i
*-                 *-
```

Since this file is in abbreviated format, we must first expand it to through-composed form using the **thru** command. The resulting output is:

```
!! 'strophe' example.
**example          **bar
*thru              *thru
*>A                *>A
A                  i
*>V                *>V
*                  **foo
*                  *thru
*                  *strophe
*                  *^
*                  *S/1            *S/2
B                  1               2
*                  *S/fin          *S/fin
```

```
*                   *v                  *v
*                   *S-
*                   **bar
*                   *thru
C                   refrain
*                   *strophe
*                   *^
*                   *S/1.1              *S/2.1
B                   1                   2
*                   *S/fin              *S/fin
*                   *v                  *v
*                   *S-
*>V                 *>V
*                   **foo
*                   *thru
*                   *strophe
*                   *^
*                   *S/1                *S/2
B                   1                   2
*                   *S/fin              *S/fin
*                   *v                  *v
*                   *S-
*                   **bar
*                   *thru
C                   refrain
*                   *strophe
*                   *^
*                   *S/1.1              *S/2.1
B                   1                   2
*                   *S/fin              *S/fin
*                   *v                  *v
*                   *S-
*>Coda              *>Coda
*                   **foo
*                   *thru
E                   i
*-                  *-
```

The command:

```
strophe file
```

will produce the following output:

```
!! 'strophe' example.
**example   **bar
*thru       *thru
*>A         *>A
A           i
*>V         *>V
*           **foo
*           *thru
B           1
*           **bar
*           *thru
C           refrain
B           1
*>V         *>V
*           **foo
*           *thru
B           2
*           **bar
*           *thru
C           refrain
B           2
*>Coda      *>Coda
*           **foo
*           *thru
E           i
*-          *-
```

PORTABILITY

DOS 2.0 and up, with the MKS Toolkit. OS/2 with the MKS Toolkit. UNIX systems supporting the *Korn* shell or *Bourne* shell command interpreters, and revised *awk* (1985).

SEE ALSO

extract (4), **thru** (4), ***strophe** (2), **yank** (4)