NAME

yank — extract passages from a Humdrum input

SYNOPSIS

yank [**-c**] **-l -r** *range* [*inputfile* ...]
yank [**-c**] **-m** *regexp* **-r** *range* [*inputfile* ...]
yank [**-c**] **-n** *regexp* **-r** *range* [*inputfile* ...]
yank [**-c**] **-o** *regexp* [**-e** *regexp*] **-r** *range* [*inputfile* ...]
yank [**-c**] **-s** *section_label* **-r** *range* [*inputfile* ...]

DESCRIPTION

The **yank** command permits the selective extraction of segments or passages from a Humdrum input. Yanked material may consist of specified input records — addressed by absolute line number, or relative to some marker. In addition, **yank** is able to output logical segments (such as measures, phrases, or labelled sections), and is able to output material according to content. The output always consists of complete records; **yank** never outputs partial contents of a given input record.

OPTIONS

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

-c	include all comments prior to the yanked material in the output
-e *regexp*	define end-delimiter for yanked segments as *regexp*; used with **-o**
-h	displays a help screen summarizing the command syntax
-l	yank all lines whose line numbers appear in **-r** *range*
-m *regexp*	yank lines matching *regexp* listed in **-r** *range*
-n *regexp*	yank segments delineated by *regexp* according to cardinal **-r** *range*
-o *regexp*	yank segments delineated by *regexp* according to ordinal **-r** *range*
-r *range*	yank section in ranges listed in *range*; used with **-l**, **-m**, **-n**, **-o** and **-s**
-s *section*	yank section labelled *section* according to **-r** *range*

Options are specified in the command line.

The simplest operation for **yank** occurs when the **-l** option is specified. In this case **yank** echoes those lines in the input stream whose line-numbers appear in a specified *range*. The *range* consists of one of more integers separated by commas; inclusive ranges can be specified by separating two integers by a dash (-). For example, the following command selects lines, 5, 13, 23, 24, 25, and 26 from the file named `dvorak`:

```
yank -l -r 5,13,23-26 dvorak
```

The dollar sign ($) can be used to refer to the last record in the input. For example, the

following command yanks the first and last records from the file `verdi`.

```
yank -l -r '1,$' verdi
```

(Note that single quotes may be needed in regular expressions and range specifications in order to prevent the shell from misinterpreting characters such as the dollar sign or the asterisk.) Records prior to the end of the input can be specified by subtracting some value from $. For example, the following command yanks the first 20 records of the last 30 records contained in the file `ginastera`. (Notice that the dash/minus sign is used both to convey a range and as an arithmetic operator.)

```
yank -l -r '$-30-$-10' ginastera
```

In addition to the specified output lines, **yank** also outputs interpretations and comments as described below (see INTERPRETATIONS AND COMMENTS).

Alternatively, **yank** can output lines relative to some user-defined *marker*. This mode of operation can be invoked using the **-m** option. Markers are specified as regular expressions. The following command outputs the first and third data records following each occurrence of the string "XXX" in the file `wieck`.

```
yank -m XXX -r 1,3 wieck
```

The **-r** option is used to specify the range. If the value zero ("0") is specified in the range, then the record containing the marker is itself output.

Since markers are interpreted by **yank** as regular expressions, complex markers can be defined. For example, the following command yanks the first data record following all occurrences of any record in the file `franck` beginning with a letter and ending with a number:

```
yank -m '^[a-zA-Z].*[0-9]$' -r 1 franck
```

In musical applications, it is often convenient to yank material according to logical segments such as measures or phrases. In order to access such segments, the user must specify a segment *delimiter* using the **-o** or the **-o** and **-e** options. For example, common system barlines are represented by the presence of an equals-sign (=) at the beginning of a data token. Thus the user might yank specific measures from a `**kern` file by defining the appropriate barline delimiter and providing a range of (measure) numbers. Consider the following command:

```
yank -o ^= -r 1,12-13,25 joplin
```

Unlike the **-m** option, the **-o** option interprets the range list as ordinal occurrences of segments delineated by the delimiter. Whole segments are output rather than specified records as is the case with **-m.** As in the case of markers, delimiters are interpreted according to regular expression syntax. Each input record containing the delimiter is regarded as the *start* of the next logical segment. In the above command, the command-line

range specifies that the first, twelfth, thirteenth, and twenty-fifth logical segments (measures) are to be yanked from the file named `joplin`. All records starting with the delimiter record are output up to, but not including the next occurrence of a delimiter record.

Where the input stream contains data prior to the first delimiter record, this data may be addressed as logical segment "zero." For example,

```
yank -o '^=' -r 0 mahler
```

can be used to yank all records prior to the first common system barline. With the **-o** option, notice that *actual* measure numbers are irrelevant: **yank** selects segments according to their *ordinal* position in the input stream rather than according to their *cardinal* label.

When the **-n** option is invoked, however, **yank** expects a numerical value to be present in the input immediately following the user-specified delimiter. In this case, **yank** selects segments based on their numbered label rather than their ordinal position in the input. For example,

```
yank -n ^= -r 12 goldberg
```

will yank all segments begining with the label `=12` in the input file `goldberg`. If more than one segment carries the specified segment number(s), all such segment are output. Note that the dollar-sign anchor cannot be used in the range expression for the **-n** option. Note also that input tokens containing non-numeric characters appended to the number will have no effect on the pattern match. For example, input tokens such as `=12a`, `=12b`, or `=12`; will be treated as equivalent to `=12`.

As in the case of the **-o** option, the value zero ("0") addresses material prior to the first delimiter record. Like the **-o** option, the value zero may be reused for each specified input file. Thus, if `file1`, `file2` and `file3` are Humdrum files:

```
yank -n ^= -r 0 file1 file2 file3
```

will yank any leading (anacrusis) material in each of the three files.

When the **-s** option is invoked, **yank** extracts passages according to Humdrum section labels encoded in the input. Humdrum section labels are tandem interpretations that conform to the syntax:

**>label_name*

Labels are frequently used to indicate formal divisions, such as, coda, exposition, bridge, second ending, trio, minuet, etc. The following command yanks the second instance of a section labelled `First Theme` in the file `haydn08`:

```
yank -s 'First Theme' -r 2 haydn08
```

Note that with "through-composed" Humdrum files it is possible to have more than one section containing the same section-label. (See the **thru** command.)

INTERPRETATIONS AND COMMENTS

If **yank** is given a Humdrum input, it always produces a syntactically correct Humdrum output. All interpretations prior to and within the yanked material are echoed in the output. The **yank** command also appends the appropriate spine-path terminators at the end of the yanked segment.

Any comments *prior* to the yanked passage may be included in the output by specifying the **-c** option.

EXAMPLES

The following examples illustrate how the **yank** command may be used.

```
yank -l -r 1120 messiaen
```

yanks line 1120 in the file `messiaen`.

```
yank -n ^= -r 27 sinfonia
```

yanks numbered measures 27 from the `**kern` file `sinfonia`.

```
yank -n ^= -r 10-20 minuet waltz
```

yanks numbered measures 10 to 20 from *both* the `**kern` files `minuet` and `waltz`.

```
yank -o ^= -r '0,$' fugue ricercar
```

yanks any initial anacrusis material plus the final measure of both `fugue` and `ricercar`.

```
cat fugue ricercar | yank -o ^= -r '0,$'
```

yanks any initial anacrusis material from the file `fugue` followed by the final measure of `ricercar`.

```
yank -n 'Rehearsal Marking ' -r 5-7 fugue ricercar
```

yanks segments beginning with the strings `"Rehearsal Marking 5,"` `"Rehearsal Marking 6,"` and `"Rehearsal Marking 7."` Segments are deemed to end when a record is encountered containing the string `"Rehearsal Marking "`.

```
yank -o { -e } -r '1-$' webern
```

yanks all segments in the file `webern` beginning with a record containing "{" and ending with a record containing "}." The command:

```
yank -o { -e } -r '1-4,$-3-$' faure
```

yanks the first four and last four segments in the file `faure` — where segments begin with an open brace ({) and end with a closed brace (}). In the `**kern` representation, this would extract the first and last four phrases in the file.

```
yank -s Coda -r 1 stamitz
```

will yank the first occurrence of a section labelled `Coda` in the file `stamitz`.

WARNINGS

Where integer ranges are specified in the **yank** range-list, overlapping values are collapsed. For example, the command `yank -l -r 5-7,6-8` is interpreted as equivalent to `yank -l -r 5-8`; lines 6 and 7 will be echoed only once in the output stream. If the user wishes to have multiple occurrences of material in the output stream, the **yank** command can be invoked more than once (e.g. `yank -l -r 5-7 ...; yank -l -r 6-8 ...`).

Note that yanked segments cannot be output in an order other than their order in the input. For example, assuming that measure numbers in an input stream increase sequentially, **yank** is unable to output measure number 6 prior to outputting measure number 5. Once again, the order of output material can be rearranged by invoked the **yank** command more than once (e.g. `yank -l -r 100 ...; yank -l -r 99 ...; yank -l -r 98 ...`).

In the case of the **-m** option, note that range elements cannot address records more than one marker away from the current marker. For example, in a file where markers occur every 10 records, range expressions such as `'25'` and `'$-17'` will result in no output. In addition, range expressions such as `'1-25'` and `'$-17-$'` will have the same effect as `'1-10'` and `'$-9-$`. Note also that for the same input file, the range expression `'6-$-7'` will result in no output.

FILES

The files `find_reg.awk`, `findpair.awk` and `number.awk` are used by **yank.**

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

awk (UNIX), **extract** (4), **grep** (UNIX), **egrep** (UNIX), **patt** (4), **pattern** (4), **regexp** (4), **regexp** (6), **timebase** (4), **thru** (4)