NAME

xargs − construct argument list(s) and execute command

SYNOPSIS

xargs [flags] [command [initial-args]]

DESCRIPTION

Xargs combines the fixed *initial-args* with args read from standard input to execute the specified *command* one or more times. The *command* can either be executed for each line of args read, with all args read for each automatically-determined group of (at most *size* characters of) args read, or for each user-specifiable *number* of args read.

Specifically, *xargs* reads the standard input for arguments, using them to construct one or more arg lists with *initial-args* (if any), and executes *command* with each such constructed argument list; the directory containing *command*, which may also be a Shell file, must be in one's *.path* file. If *command* is omitted, **/bin/echo** is used. Excepting the use of the insert option (**−i** flag, see below), arguments read in from standard input are defined to be contiguous strings of characters delimited by one or more blanks, tabs, or newlines; however, quoted strings (including embedded blanks or tabs) may also form all or part of an argument.

Excepting the **−i** option, each argument list will be constructed starting with the *initial-args*, followed by an appropriate number of arguments read from standard input. Flags **−i, −l,** and **−n** modify how args are selected for each command invocation; when none of these flags are coded, each arg list is built from the continuously-read args from standard input, up to *size* characters per list maximum, until there are no more args. When there are flag conflicts (e.g., **−l** vs. **−n**), the last flag has precedence. Flag values are:

−x Causes *xargs* to terminate if any arg list would be greater than *size* characters; **−x** is forced by the options **−i** and **−l.** When neither of the options **−i, −l,** or **−n** are coded, the total length of all args must be within the *size* limit.

−l *Command* is executed for each non-null line of args from standard input. A line is considered to end with the first newline *unless* the last character of the line is a blank or a tab; in either of these cases, the blank/tab signals continuation through the next non-null line. Option**−x** is forced.

−ireplstr Insert mode: *command* is executed for each line from standard input, taking the entire line as one entity, inserting it in *initial-args* for each occurrence of *replstr.* A maximum of 5 args in *initial-args* may each contain one or more instances of *replstr.* Blanks and tabs at the beginning of each line are thrown away, as are empty lines. Constructed args may not grow larger than 255 characters, and option **−x** is also forced. '{ }' is assumed for *replstr* if not specified.

−nnumber Execute *command* using as many standard input args as possible, up to *number* args maximum. Fewer args will be used if their total size is greater than *size* characters, and for the last invocation if there are fewer than *number* args remaining. If option **−x** is also coded, each *number* args must fit in the *size* limitation, else *xargs* terminates execution.

−t Trace mode: the *command* and each constructed arg list are echoed to file descriptor 2 just prior to their execution.

−p Prompt mode: the user is asked whether to execute *command* each invocation. Trace mode (**−t**) is turned on to print the command instance to be executed, followed by the prompt '**?...**'. A reply of **y** (optionally followed by anything) will execute the command; anything else, including just a carriage return, skips that particular invocation of *command.*

−ssize The maximum total size of each arg list is set to *size* characters; *size* must be a positive integer less than or equal to 470. If **−s** is not coded, 470 is taken as the default. Note that

the character count for *size* includes one extra character for each arg and the count of characters in the command name.

−eeofstr *Eofstr* is taken as the logical end-of-file string. Underbar (_) is assumed for the logical EOF string if **−e** is not coded. **−e** with no *eofstr* coded turns off the logical EOF string capability (underbar is taken literally). *Xar gs* reads standard input until either end-of-file or the logical EOF string is encountered.

In args read from standard input, characters may be escaped (by a '\') outside of quoted strings; quoted strings are stripped of the delimiting quotes, with the contents taken literally.

Xargs will terminate if either it receives a return code of minus one from, or if it cannot execute, *command.*

EXAMPLES

The following will copy all files from directory $1 to directory $2, and echo each move command just before doing it:

```
ls $1 | xargs −i −t mv $1/{ } $2/{ }
```

The following will combine the output of the parenthesized commands onto one line, which is then echoed to the file *log*:

```
(logname; date; echo $0 $*) | xargs >>log
```

The user is asked which files in the current directory are to be archived and archives them into *arch* (1.) one at a time, or (2.) many at a time.

```
1. ls | xargs −p −l ar r arch
2. ls | xargs −p −l | xargs ar r arch
```

The following will execute *com* with successive pairs of args originally typed as Shell arguments:

```
echo $* | xargs −n2 com
```

DIAGNOSTICS

arg list too long
command not executed or returned −1
Missing quote? <string>
too many args with *replstr*
insert-buffer overflow
max arg size with insertion via *replstr* exceeded
unknown option: <option>
0 < max-line-size <= 470: <−s option as coded>
#args must be positive int: <−n option as coded>
can't read from tty for −p