

unix

COLLABORATORS

	TITLE : unix		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY		December 8, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	unix	1
1.1	main	1
1.2	intro	1
1.3	cmp	2
1.4	comm	2
1.5	compute	3
1.6	convert	4
1.7	cut	4
1.8	date	5
1.9	dc	7
1.10	extract	8
1.11	file	9
1.12	find	10
1.13	fold	11
1.14	head	12
1.15	ln	12
1.16	newform	13
1.17	nl	14
1.18	paste	16
1.19	split	17
1.20	strings	18
1.21	tail	18
1.22	tee	19
1.23	test	19
1.24	tr	20
1.25	wc	21

Chapter 1

unix

1.1 main

Intro	Introduction, distribution, disclaimer
cmp	Compare two files
comm	Select or reject lines common to two sorted files
compute	Various numeric operations on a file
convert	File conversion filter
cut	Cut out selected fields of each line of a file
date	Print and set the date
dc	Desk calculator
extract	Extract bytes from a file
file	Find file type
find	Find files
fold	Fold long lines for finite width output device
head	Give first few lines
ln	Create a link
newform	Change the format of a text file
nl	Line numbering filter
paste	Merge same lines of several files
split	Split a file into pieces
strings	Find printable strings in a binary file
tail	Deliver the last part of a file
tee	Pipe fitting
test	Condition evaluation command
tr	Translate characters
wc	Word count

1.2 intro

This is a package of some UNIX commands, ported by Denis GOUNELLE.

The original work was made in 1991, and only some minor changes occurs since then. Last modification was made 16-Apr-94, when I recompiled all the programs with SAS/C 6.51, fixed a few bugs, and change the doc to AmigaGuide format.

Any commercial usage or selling without author's written authorization is strictly forbidden. You can copy and spread this package under the following conditions:

- all the files are provided
- the files are not modified in any way
- you don't charge more than \$6 for copy fee

In spite of several tests, no warranty is made that there are no errors in these commands. YOU USE THIS PACKAGE AT YOUR OWN RISK. In no event will I be liable for any damage, direct or indirect, resulting of the use of these commands.

1.3 cmp

NAME

cmp - compare two files

SYNOPSIS

cmp [-r] [-s] file1 file2

DESCRIPTION

The two files are compared. Under default options, cmp makes no comment if the files are the same; if they differ, it announces the byte number at which the difference occurred and the two differing bytes. The differing bytes are printed (first) as an hexadecimal value and (second) as a character (replaced by a point if not printable).

If you specify - as file1 or file2, standard input is used.

Options:

- r Reverse comparison (announce bytes that are the same).
- s Print nothing, return codes only.

RETURN CODES

- 0 if all went well (files are identicals)
- 1 in case of bad argument (fatal)
- 2 if files are different

CHANGES FROM UNIX

- l option not supported
- added -r option
- print differences as hexa and character.

1.4 comm

NAME

comm - select or reject lines common to two sorted files

SYNOPSIS

```
comm [ - [ 123 ] ] file1 file2
```

DESCRIPTION

comm reads file1 and file2, which must be ordered in ASCII collating sequence, and produces a three-column output: lines only in file1; lines only in file2; and lines in both files.

If you specify - as file1 or file2, standard input is used.

Flags 1, 2, or 3 suppress printing of the corresponding column. Thus comm -12 prints only the lines common to the two files; comm -23 prints only lines in the first file but not in the second; comm -123 prints nothing.

RETURN CODES

- 0 if all went well
- 1 in case of bad argument (fatal)
- 2 if at least one file couldn't be opened (fatal)

CHANGES FROM UNIX

None

1.5 compute

NAME

compute - various numeric operations on a file

SYNOPSIS

```
compute operation -ffield [-ddelim] [ file... ]
```

DESCRIPTION

compute read each line of the named file(s) (or of standard input if no named file), extract the field-th field of this line (field separator is delim, default is tab) and perform operation on this field. When end of file is reached, the result is printed on standard output.

If you specify "-" as a file name, standard input will be used.

Operations are :

-s	summ
-p	product
-a	average
-g	find greatest value
-l	find lowest value

RETURN CODES

- 0 if all went well
- 1 in case of bad argument (fatal)
- 2 if at least one file couldn't be opened (non fatal)

CHANGES FROM UNIX

This is not an Unix command !

1.6 convert

NAME

convert - file conversion filter

SYNOPSIS

convert -src -dst

DESCRIPTION

convert is a filter that translate file that are to be tranfered from a computer to another : standard input (assumed to be in -src format) is read, converted in -dst format, and send on standard output.

The -src and -dst specifications can be :

-in9400	in9400 terminal
-amiga	Amiga, PT10 laser printer (ECMA-94)
-ibmpc	PC, PT10 laser printer (IBM-US)
-pt10rom	PT10 laser printer (ROMAN-8)
-mac	Macintosh

All lowercase stressed letters are translated, along with a few special characters. "end of line" and "end of file" marks are correctly handled and converted.

Non convertible characters are lost, so if you specify same source and destination format, some characters may disapear...

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)

CHANGES FROM UNIX VERSION

This is not an Unix command !

1.7 cut

NAME

cut - cut out selected fields of each line of a file

SYNOPSIS

cut -clist [file ...]
cut -flist [-dchar] [-s] [file ...]

DESCRIPTION

Use cut to cut out columns from a table or fields from each line of a file; in data base parlance, it implements the projection of a relation. The fields as specified by list can be fixed length, i.e., character positions as on a punched card (-c option) or the length can vary from line to

line and be marked with a field delimiter character like tab (-f option). cut can be used as a filter : if no files are given, the standard input is used. If you specify - as a file, standard input will be used.

The meanings of the options are:

list	A comma-separated list of integer field numbers (in increasing order), with optional - to indicate ranges (e.g.: 1,4,7 or 1-3,8) You can specify -y for 1-y or x- for x till end, or even - for all.
-clist	The list following -c (no space) specifies character positions (e.g., -cl-72 would pass the first 72 characters of each line).
-flist	The list following -f is a list of fields assumed to be separated in the file by a delimiter character (see -d); e.g., -f1,7 copies the first and seventh field only. Lines with no field delimiters will be passed through intact (useful for table subheadings), unless -s is specified.
-dchar	The character following -d is the field delimiter (-f option only). Default is tab. Space or other characters with special meaning to the shell must be quoted.
-s	Suppresses lines with no delimiter characters in case of -f option. Unless specified, lines with no delimiters will be passed through untouched.

Either the -c or -f option must be specified as the first command line option.

RETURN CODES

- 0 if all went well
- 1 in case of bad argument (fatal)
- 2 if at least one file couldn't be opened (non fatal)

CHANGES FROM UNIX

Doesn't complain when a list is x-y and y is greater than x, but simply exchange values.

1.8 date

NAME

date - print and set the date

SYNOPSIS

date +format
date yy-mm-dd hh:mm:ss

DESCRIPTION

If no argument is given, or if the argument begins with +, the current date and time are printed. Otherwise, the current date is set. yy is the last 2 digits of the year number; the first mm is the month number; dd is the day number in the month; hh is the hour number (24 hour system); the second mm is the minute number; ss is the seconds number. You can specify either date or time, or both :

date 87-05-14 sets date to May 14, 1987
 (time unchanged)

date 13:45:00 sets time to 13h45
 (date unchanged)

date 87-05-14 13:45:00 sets date to May 14, 1987 and
 time to 13h45

If the argument begins with +, the output of date is under the control of the user. All output fields are of fixed size (zero padded if necessary). Each field descriptor is preceded by % and will be replaced in the output by its corresponding value. A single % is encoded by %%. All other characters are copied to the output without change. The string is always terminated with a new-line character.

Field Descriptors :

n	insert a new-line character
t	insert a tab character
m	month of year - 01 to 12
d	day of month - 01 to 31
y	last 2 digits of year - 00 to 99
D	date as mm/dd/yy
H	hour - 00 to 23
M	minute - 00 to 59
S	second - 00 to 59
T	time as HH:MM:SS
j	day of year - 001 to 366
w	day of week - Sunday = 0
a	abbreviated weekday - Sun to Sat
h	abbreviated month - Jan to Dec
r	time in AM/PM notation

RETURN CODES

0 if all when well
1 in case of bad argument (fatal)
4 if date and/or time couldn't be changed (fatal)

CHANGES FROM UNIX

Format of new date and time is different

BUGS

No checking is done on new time/date values (e.g. "date 91-67-05" or "date 29:75:00" will work !)

1.9 dc

NAME

dc - desk calculator

SYNOPSIS

dc [file]

DESCRIPTION

dc is an arbitrary precision arithmetic package. Ordinarily it operates on decimal numbers, but you may specify an input base or an output base.

The overall structure of dc is a stacking (reverse Polish) calculator. If an argument is given input is taken from that file, else standard input is used.

The following constructions are recognized :

number

The value of the number is pushed on the stack. A number is an unbroken string of the digits 0-9. It may be preceded by an underscore (_) to input a negative number.

+ - / * % ^ & |

The top two values on the stack are added (+), subtracted (-), multiplied (*), divided (/), remaindered (%), powered (^), anded (&), ored(|). The two entries are popped off the stack; the result is pushed on the stack in their place.

! The top value on the stack is popped. If non-zero, 1 is pushed back on the stack, else 0 is pushed.

~ The top value on the stack is popped, inverted, and the result pushed back on the stack

sr The top of the stack is popped and stored into a register named r, where r may be any character between a and z.

lr The value in register r is pushed on the stack. The register r is not altered. All registers start with zero value.

d The top value on the stack is duplicated.

p The top value on the stack is printed. The top value

remains unchanged.

f All values on the stack are printed.

q Exits the program.

x Treats the top element of the stack as a character string and executes it as a string of dc commands.

[...] Puts the bracketed ASCII string onto the top of the stack.

c All values on the stack are popped.

i The top value on the stack is popped and used as the number radix for further input.

I Pushes the input base on the top of the stack.

o The top value on the stack is popped and used as the number radix for further output.

O Pushes the output base on the top of the stack.

k The top value on the stack and used to set the number of digit displayed after decimal point. By default precision is set to 0 : dc displays only integer part.

v The top value on the stack is replaced by its square root

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)
2 if the named file couldn't be opened (fatal)
3 if no memory could be allocated (fatal)
6 if stack was empty (fatal)
7 if stack was full (fatal)
11 in case of missing bracket (fatal)
14 if you try to compute the square root of a negative number

CHANGES FROM UNIX

P, Q, X, <r, >r, =r, z, Z, and ? operators missing
Registers are not stacks nor strings
[expr] computed when popped
Cannot handle a full expression on the same line
Doesn't read standard input when EOF reached on named file

1.10 extract

NAME

extract - extract bytes from a file

SYNOPSIS

```
extract <pos> { <len> | - } [ file ]
```

DESCRIPTION

Extract <len> bytes from the given file (or from standard input), starting to the position <pos>, to the standard output.
If you specify "-" for <len>, data is extracted until end of file.

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)
2 if file couldn't be opened (fatal)
17 if seeking to <pos> failed (fatal)

CHANGES FROM UNIX

This is not an UNIX command !

1.11 file

NAME

file - find file type

SYNOPSIS

file names

DESCRIPTION

file tries to guess the type of each named files. Recognized types are :

ABackup archive
AmigaBASIC source file
ascii file
Aztec C object file
Aztec C library module
binary file
directory
font header
empty file
executable object
IFF picture
IFF text
IFF music
IFF animation
Lattice C object file
Lharc archive
other IFF file
PowerPacker crunched data
PowerPacker crypted data
Workbench icon
ZOO archive file

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)
2 if at least one file couldn't be opened (non fatal)

3 if memory couldn't be allocated (fatal)

CHANGES FROM UNIX

File types are obviously different !

1.12 find

NAME

find - find files

SYNOPSIS

find <name> expression

DESCRIPTION

find recursively descends the directory hierarchy seeking files or directories that match a boolean expression written in the primaries given below.

If <name> is a directory name, find starts in the given directory. If <name> is a file name (or - for standard input), each line of the file is supposed to be a object name, and the boolean expression is applied on each of the names.

- | | |
|--------------|---|
| -name <name> | True if <name> matches the current objet name.
You can use "*name" or "name*" to match names
ending or beginning whith "name". |
| -perm <bits> | True if the object permission flags exactly match
given bits (valid bits are any combination of
"arwed"). If <bits> is prefixed by a "+" sign,
the given bits must be set. If <bits> is prefixed
by a "-" sign, the given bits must be cleared. |
| -type <t> | True if the type of the object is <t>, where <t> is
"f" for file and "d" for directory. Under 2.0 system
release, find also recognize "s" for soft links and
"h" for hard links. |
| -size <n> | True if the file size (in bytes) is <n>. If <n> is
prefixed by a "+" sign, returns true if the file size
greater or equal than <n>. |
| -note <note> | True if object note is <note>. Comparison is case-
insensitive, and limited to the number of characters
of <note>. |
| -exec <cmd> | cause the given command to be executed, with {}
replaced by the current object name.
False only if command couldn't be executed. |
| -fexec <cmd> | same as -exec, but false if command couldn't be
executed OR if command returned non-zero status, and
true if command returned zero. |
| -ok [<str>] | If no <str> is given, displays the current object
name followed by " (y/n)?", else displays the string |
-

	<code><str></code> , with <code>{}</code> replaced by the current object name. Then waits for user answer, and returns true if the reply was "y", and false if reply was "n" (any other answer is ignored).
<code>-print</code>	Always true; causes the current object name to be printed.
<code>-printf <fmt></code>	Always true; cause <code><fmt></code> to be displayed, with <code>{}</code> replaced by the current object name.
<code>-newer <file></code>	True if the current object has been modified more recently than the named <code><file></code> .
<code>-depth</code>	Always true; causes descent of the directory hierarchy to be done so that all entries in a directory are acted on before the directory itself.
<code>-not</code>	Reverts the result of the next primary (i.e. false if next primary is true).

The primaries are evaluated the order in which they are specified, and evaluation stops as soon as a primary return false. Multiple occurrences of each primary is supported except for `-depth`.

RETURN CODES

- 0 if all went well
- 1 in case of bad argument (fatal)
- 2 if no file matched the boolean expression (non fatal)
- 3 if some memory couldn't be allocated (fatal)

CHANGES FROM UNIX

- `-links`, `-user`, `-group`, `-atime`, `-mtime`, `-ctime`, `-cpio`, `-mount`, and `-local` switches not supported
- VERY limited pattern matching for `-name` switch
- Amiga `-fexec` switch correspond to UNIX `-exec` switch
- No `-o` (or) operators, no parenthesis in expression
- Added `-printf` and `-exec` switches
- Added `<str>` argument to `-ok` switch

NOTES

The `-exec` and `-fexec` switches will work only if the "Run" command is in your "C:" directory.
Arguments to `-printf`, `-ok`, `-exec` and `-fexec` are expanded as follow :

<code>{}</code>	current object name
<code>\n</code>	new-line character
<code>\t</code>	tabulation character
<code>\x</code>	x (if x is neither 'n' nor 't')

1.13 fold

NAME

`fold` - fold long lines for finite width output device

SYNOPSIS

```
fold [ -width ] [ -pn ] [ -tn ] [ file ... ]
```

DESCRIPTION

fold is a filter which will fold the contents of the specified files (or the standard input if no files are specified) breaking the lines to have maximum width <width>. The default for <width> is 80.

If you specify - as a file, standard input is used.

-pn cause fold to add n spaces at the right of the folded part of the line

-tn set tab stop positions to 1, n+1, 2n+1, etc...

RETURN CODES

0 if all went well

1 in case of bad argument (fatal)

2 if at least one file couldn't be opened (non fatal)

CHANGES FROM UNIX

Added -p and -t options

1.14 head

NAME

head - give first few lines

SYNOPSIS

```
head [ -count ] [ file ... ]
```

DESCRIPTION

This filter gives the first count lines of each of the specified files, or of the standard input. If you specify - as a file, standard input will be used.

If count is omitted it defaults to 10.

RETURN CODES

0 if all went well

1 in case of bad argument (fatal)

2 if at least one file couldn't be opened (non fatal)

CHANGES FROM UNIX

None

1.15 ln

NAME

ln - create a link

SYNOPSIS

```
ln [ -s ] src dst
```

```
ln -r src
```

DESCRIPTION

The first form creates a link named "src" to the file named "dst". By default it's a "hard link", but the -s switch allow to create a "soft link".
The second form examines the link "src" and tells where it point to.

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)
2 if one file couldn't be opened (fatal)
15 if system release is not 2.0 (fatal)
16 if link couldn't ne created (fatal)

CHANGES FROM UNIX

added -r option

NOTES

You must have Kickstart 2.04 or greater to use this command

1.16 newform

NAME

newform - change the format of a text file

SYNOPSIS

```
newform [-s] [-in] [-on] [-bn] [-en] [-pn] [-an] [-ck] [-ln] [-r] [-z]  
[files...]
```

DESCRIPTION

newform reads lines from the named files, or the standard input if no input file is named, and reproduces the lines on the standard output. Lines are reformatted in accordance with command line options in effect. Except for -s, command line options may appear in any order, and may be repeated. Command line options are processed in the order specified, this means that option sequences like "-e15 -l60" will yield results different from "-l60 -e15". Options are applied to all files on the command line. If you specify "-" as a file name, standard input will be used.

-s Shears off leading characters on each line up to the first tab and places up to 8 of the sheared characters at the end of the line. If more than 8 characters (not counting the first tab) are sheared, the eighth character is replaced by a * and any characters to the right of it are discarded. The first tab is always discarded.

An error message and program exit will occur if this option is used on a file without a tab on each line. The sheared off characters are saved internally until all other options specified are applied to that line. The characters are then added at the end of the processed line.

-in Input tab specification : expands tabs to spaces, according

to the given tab width. n may be 0 to stop tab expansion.

- on Output tab specification : replaces spaces by tabs, according to the given tab width. n may be 0 to stop space conversion on output.
- bn Truncate n characters from the beginning of the line when the line length is greater than the effective line length (see -l). Default is to truncate the number of characters necessary to obtain the effective line length. The default value is used when -b with no n is used.
- en Same as -bn except that characters are truncated from the end of the line.
- pn Prefix n characters (see -c) to the beginning of a line when the line length is less than the effective line length. Default is to prefix the number of characters necessary to obtain the effective line length.
- an Same as -pn except characters are appended to the end of a line.
- ck Change the prefix/append character to k. Default character is a space. If k is missing, space is assumed.
- ln Set the effective line length to n characters. If n is not entered, -l defaults to 80. The default line length without the -l option is 80 characters. Note that tabs are considered to be one character (use -i to expand tabs to spaces).
- r Delete empty lines at the end of the file.
- z Delete spaces and tabs at the beginning of a line.

RETURN CODES

- 0 if all went well
- 1 in case of bad argument (fatal)
- 2 if at least one file couldn't be opened (non fatal)
- 13 if there's no tab in an input line with -s option (fatal)

CHANGES FROM UNIX VERSION

- f option not supported
- n cannot be -- for -i and -o option
- added -z and -r options

1.17 nl

NAME

nl - line numbering filter

SYNOPSIS

```
nl [ -htype ] [ -btype ] [ -ftype ] [ -vstart ] [ -iincr ]
   [ -p ] [ -lnum ] [ -ssep ] [ -wwidth ] [ -nfmt ]
```

```
[ -dxx ] [ file... ]
```

DESCRIPTION

nl reads lines from the named files, or the standard input if no file is named, and reproduces the lines on the standard output. Lines are numbered on the left in accordance with the command options in effect. nl views the text it reads in terms of logical pages. Line numbering is reset at the start of each logical page. A logical page consists of a header, a body, and a footer section. Empty sections are valid. Different line numbering options are independently available for header, body, and footer. The start of logical page sections are signaled by input lines containing nothing but the following delimiter character(s):

Line contents:	start of:
\\: \\: \\:	header
\\: \\:	body
\\:	footer

Unless optioned otherwise, nl assumes the text being read is in a single logical page body. Command options may appear in any order. If a file name is -, standard input will be used. The options are:

-btype Specifies which logical page body lines are to be numbered. Recognized types and their meaning are:

a	number all lines
t	number lines with printable text only
n	no line numbering

Default type for logical page body is t (text lines numbered).

-htype Same as -btype except for header. Default type for logical page header is n (no lines numbered).

-ftype Same as -btype except for footer. Default for logical page footer is n (no lines numbered).

-vstart start is the initial value used to number logical page lines. Default is 1.

-iincr incr is the increment value used to number logical page lines. Default is 1.

-p Do not restart numbering at logical page delimiters.

-lnum num is the number of blank lines to be considered as one. For example, -l2 results in only the second adjacent blank being numbered (if the appropriate -ha, -ba, and/or -fa option is set). Default is 1.

-ssep sep is the character(s) used in separating the line number

and the corresponding text line. Default sep is a tab.

-wwidth width is the number of characters to be used for the line number.

Default number is 6.

-nfmt fmt is the line numbering format.

Recognized values are: ln, left justified, leading zeroes suppressed; rn, right justified, leading zeroes suppressed; rz, right justified, leading zeroes kept.

Default fmt is rn (right justified).

-dxx The delimiter characters specifying the start of a logical page section may be changed from the default characters (\:) to two user-specified characters.

RETURN CODES

0 if all went well

1 in case of bad argument (fatal)

2 if at least one file couldn't be opened (non fatal)

CHANGES FROM UNIX

Several files can be specified

Two characters must be specified for -d option

pexpr type for -h/-b/-f options is not supported

1.18 paste

NAME

paste - merge same lines of several files or subsequent lines of one file

SYNOPSIS

paste [-dlist] file...

paste -s [-dlist] file...

DESCRIPTION

In the first form, paste concatenates corresponding lines of the given input files. It treats each file as a column or columns of a table and pastes them together horizontally (parallel merging).

In the second form above, paste the function of an older command with the same name by combining subsequent lines of the input files (serial merging).

In all cases, lines are glued together with a tab, or with characters from an optionally specified list. Output is sent to the standard output.

The meanings of the options are:

-d Without this option, the new-line characters of each but the last file (or last line in case of the -s option) are replaced by a tab character. This option allows replacing the tab character by one or more alternate characters (see below).

list One or more characters immediately following -d replace the

default tab as the line concatenation character. The list is used circularly, i.e., when exhausted, it is reused. The list may contain the special escape sequences:

```

    \n      new-line
    \t      tab
    \       backslash

```

-s Merge subsequent lines rather than one from each input file. Use tab for concatenation, unless a list is specified with -d option.

- May be used in place of any file name, to read a line from the standard input.

RETURN CODES

```

0 if all went well
1 in case of bad argument (fatal)
2 if at least one file couldn't be opened (non fatal)
3 if some memory couldn't be allocated (fatal)

```

CHANGES FROM UNIX

```

output lines length not limited
\0 in list not supported
-s option works even with only one file name
result of paste with -s option is not the same (but is more logic
to my mind)

```

1.19 split

NAME

split - split a file into pieces

SYNOPSIS

```
split [ -n ] [ -c ] [ file [ name ] ]
```

DESCRIPTION

split reads file and writes it in n-line pieces (default 1000 lines) onto a set of output files. The name of the first output file is name with aa appended, and so on lexicographically, up to zz (a maximum of 676 files). If no output name is given, x is default.

If -c option is specified, split counts in characters rather than in lines (so making pieces of n-characters).

If no input file is given, or if - is given instead, then the standard input file is used.

RETURN CODES

```

0 if all went well
1 in case of bad argument (fatal)
2 if one file couldn't be opened, or too much output files (fatal)
3 if memory couldn't be allocated (fatal)

```

CHANGES FROM UNIX

name can be longer than 12 characters.

added -c option

1.20 strings

NAME

strings - find the printable strings in a binary file

SYNOPSIS

strings [-o] [-number] [file ...]

DESCRIPTION

strings looks for ascii strings in a binary file. A string is any sequence of 4 or more printing characters ending with a null. If the -o flag is given, then each string is preceded by its offset in the file (in hexadecimal). If the -number flag is given then number is used as the minimum string length rather than 4.

If no file is specified, or - is specified as a file, standard input will be used.

RETURN CODES

0 if all went well
1 in case of bad arguments (fatal)
2 if at least one file couldn't be opened (non fatal)

CHANGES FROM UNIX

-a and - options not supported
offset in hexadecimal (instead of octal)

1.21 tail

NAME

tail - deliver the last part of a file

SYNOPSIS

tail [{+|-}number] [file ...]

DESCRIPTION

tail copies the named files to the standard output beginning at a designated place. If no file is named, the standard input is used. If you specify - as a file name, standard input will be used.

Copying begins at distance +number lines from the beginning, or -number lines from the end of the input (default distance is -10).

Tails relative to the end of the file can require a lot of memory.

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)
2 if at least one file couldn't be opened (non fatal)
3 if memory couldn't be allocated (fatal)

CHANGES FROM UNIX

No -f option
Distance can't be specified in blocs nor in characters.
Several files to tail can be named

1.22 tee

NAME

tee - pipe fitting

SYNOPSIS

tee [-a] [file] ...

DESCRIPTION

tee transcribes the standard input to the standard output and makes copies in the files.
The -a option causes the output to be appended to the files rather than overwriting them.

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)
2 if at least one file couldn't be opened (non fatal)
3 if memory couldn't be allocated (fatal)

CHANGES FROM UNIX

-i option not supported.

1.23 test

NAME

test - condition evaluation command

SYNOPSIS

test expr

DESCRIPTION

test evaluates the given expression and, if its value is true, returns a zero (true) exit status; otherwise, a non-zero (false) exit status is returned.

All operators, operands and flags, must be separate arguments to the test command; normally these items are separated by spaces.

The following primitives are used to construct test :

-r objects	true if all objects exist and are readable.
-w objects	true if all objects exist, and are writable AND deletable.

<code>-x objects</code>	true if all objects exist and are executable.
<code>-f objects</code>	true if all objects exists and are files.
<code>-d objects</code>	true if all objects exists and are directories.
<code>-s objects</code>	true if all objects exists and have a size greater than zero.
<code>object1 -nt object2</code>	true if object1 is newer than object2.
<code>object1 -ot object2</code>	true if object1 is older than object2.
<code>-z string</code>	true if the length of string is zero.
<code>-n string</code>	true if the length of the string is not zero.
<code>s1 = s2</code>	true if strings s1 and s2 are identical.
<code>s1 != s2</code>	true if strings s1 and s2 are not identical.
<code>n1 -eq n2</code>	true if the integers n1 and n2 are equal. Any of the comparisons <code>-ne</code> , <code>-gt</code> , <code>-ge</code> , <code>-lt</code> , and <code>-le</code> may be used in place of <code>-eq</code> .

RETURN CODES

0 if expression was true
 1 in case of bad argument (fatal)
 2 if expression was false
 3 if memory couldn't be allocated (fatal)

CHANGES FROM UNIX

`-c`, `-b`, `-p`, `-u`, `-g`, `-k`, and `-t` switches not supported.
`!`, `-a`, `-o` operators, and parentheses not supported.
 several objects can be specified for `-r`, `-w`, `-x`, `-f`, `-d`, `-s` switches

1.24 tr

NAME

`tr` - translate characters

SYNOPSIS

`tr [-c] [-d] [-s] [string1 [string2]]`

DESCRIPTION

`tr` copies the standard input to the standard output with substitution or deletion of selected characters. Input characters found in `string1` mapped into the corresponding characters of `string2` (or deleted if `string2` is shorter than `string1`).

`-c` Complements `string1` by the characters between space (32) and tilde (126).

`-d` Deletes all input characters in `string1`.

-s Squeezes all strings of repeated output characters that are in string2 to single characters.

The following abbreviation conventions may be used to introduce ranges of characters or repeated characters into the strings:

[a-z] Stands for the string of characters whose ASCII codes run from character a to character z, inclusive.

[a*n] Stands for n repetitions of a. If the first digit of n is 0, n is considered octal; otherwise, n is taken to be decimal.

The escape character \ followed by 1, 2, or 3 octal digits stands for the character whose ASCII code is given by those digits.

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)

CHANGES FROM UNIX

n in [a*n] expressions cannot be 0 nor missing
\ cannot be used to remove special meaning of characters [,], etc...

1.25 wc

NAME

wc - word count

SYNOPSIS

wc [-lwc] [names]

DESCRIPTION

wc counts lines, words, and characters in the named files, or in the standard input if no name appear. It also keeps a total count for all named files. A word is a maximal string of characters delimited by spaces, tabs, or newl-ines.

The options l, w, and c may be used in any combination to specify what subset of lines, words, and characters counts are to be reported. The default is -lwc.

When names are specified on the command line, they will be printed along with the counts. If you specify - as a name, standard input will be used.

RETURN CODES

0 if all went well
1 in case of bad argument (fatal)
2 if at least one file couldn't be opened (non fatal)

CHANGES FROM UNIX

None
