-

NAME

azel − satellite predictions

SYNOPSIS

azel [**−d**] [**−l**] satellite1 [**−d**] [**−l**] satellite2 ...

DESCRIPTION

Azel predicts, in convenient form, the apparent trajectories of Earth satellites whose orbital elements are given in the argument files. If a given satellite name cannot be read, an attempt is made to find it in a directory of satellites maintained by the programs's author. The **−d** option causes *azel* to ask for a date and read line 1 data (see below) from the standard input. The **−l** option causes *azel* to ask for the observer's latitude, west-longitude, and height above sea level.

For each satellite given the program types its full name, the date, and a sequence of lines each containing a time, an azimuth, an elevation, a distance, and a visual magnitude. Each such line indicates that: at the indicated time, the satellite may be seen from Murray Hill (or provided location) at the indicated azimuth and elevation, and that its distance and apparent magnitude are as given. Predictions are printed only when the sky is dark (sun more than 5 degrees below the horizon) and when the satellite is not eclipsed by the earth's shadow. Satellites which have not been seen and verified will not have had their visual magnitude level set correctly.

All times input and output by *azel* are GMT (Universal Time).

The satellites for which elements are maintained are:

sla,b,e,f,k Skylab A through Skylab K. Skylab A is the laboratory; B was the rocket but it has crashed. A and probably K have been verified.

cop Copernicus I. Never verified.

oao Orbiting Astronomical Observatory. Seen and verified.

pag Pageos I. Seen and verified; fairly dim (typically 2nd-3rd magnitude), but elements are extremely accurate.

exp19 Explorer 19; seen and verified, but quite dim (4th-5th magnitude) and fast-moving.

c103b, c156b, c184b, c206b, c220b, c461b, c500b
Various of the USSR Cosmos series; none seen.

7276a Unnamed (satellite # 72-76A); not seen.

The element files used by *azel* contain five lines. The first line gives a year, month number, day, hour, and minute at which the program begins its consideration of the satellite, followed by a number of minutes and an interval in minutes. If the year, month, and day are 0, they are taken to be the current date (taken to change at 6 A.M. local time). The output report starts at the indicated epoch and prints the position of the satellite for the indicated number of minutes at times separated by the indicated interval. This line is ended by two numbers which specify options to the program governing the completeness of the report; they are ordinarily both ‘‘1’’. The first option flag suppresses output when the sky is not dark; the second supresses output when the satellite is eclipsed by the earth's shadow. The next line of an element file is the full name of the satellite. The next three are the elements themselves (including certain derivatives of the elements).

FILES

/usr/jfo/el/* − orbital element files

SEE ALSO

sky (VI)

AUTHOR

J. F. Ossanna

BUGS

-

NAME

bj – the game of black jack

SYNOPSIS

/usr/games/bj

DESCRIPTION

Bj is a serious attempt at simulating the dealer in the game of black jack (or twenty-one) as might be found in Reno. The following rules apply:

The bet is $2 every hand.

> A player 'natural' (black jack) pays $3. A dealer natural loses $2. Both dealer and player naturals is a 'push' (no money exchange).
>
> If the dealer has an ace up, the player is allowed to make an 'insurance' bet against the chance of a dealer natural. If this bet is not taken, play resumes as normal. If the bet is taken, it is a side bet where the player wins $2 if the dealer has a natural and loses $1 if the dealer does not.
>
> If the player is dealt two cards of the same value, he is allowed to 'double'. He is allowed to play two hands, each with one of these cards. (The bet is doubled also; $2 on each hand.)
>
> If a dealt hand has a total of ten or eleven, the player may 'double down'. He may double the bet ($2 to $4) and receive exactly one more card on that hand.
>
> Under normal play, the player may 'hit' (draw a card) as long as his total is not over twenty-one. If the player 'busts' (goes over twenty-one), the dealer wins the bet.
>
> When the player 'stands' (decides not to hit), the dealer hits until he attains a total of seventeen or more. If the dealer busts, the player wins the bet.
>
> If both player and dealer stand, the one with the largest total wins. A tie is a push.

The machine deals and keeps score. The following questions will be asked at appropriate times. Each question is answered by **y** followed by a new line for 'yes', or just new line for 'no'.

? (means, "do you want a hit?")
Insurance?
Double down?

Every time the deck is shuffled, the dealer so states and the 'action' (total bet) and 'standing' (total won or lost) is printed. To exit, hit the interrupt key (DEL) and the action and standing will be printed.

BUGS

NAME

cal − print calendar

SYNOPSIS

cal [month] year

DESCRIPTION

Cal prints a calendar for the specified year. If a month is also specified, a calendar just for that month is printed. *Year* can be between 1 and 9999. The *month* is a number between 1 and 12. The calendar produced is that for England and her colonies.

Try September 1752.

BUGS

The year is always considered to start in January even though this is historically naive.

-

NAME

chess – the game of chess

SYNOPSIS

/usr/games/chess

DESCRIPTION

Chess is a computer program that plays class D chess. Moves may be given either in standard (descriptive) notation or in algebraic notation. The symbol '+' is used to specify check; 'o-o' and 'o-o-o' specify castling. To play black, type 'first'; to print the board, type an empty line.

Each move is echoed in the appropriate notation followed by the program's reply.

FILES

/usr/lib/book opening 'book'

DIAGNOSTICS

The most cryptic diagnostic is 'eh?' which means that the input was syntactically incorrect.

WARNING

Over-use of this program will cause it to go away.

BUGS

Pawns may be promoted only to queens.

NAME

col − filter reverse line feeds

SYNOPSIS

col

DESCRIPTION

Col reads the standard input and writes the standard output. It performs the line overlays implied by reverse line feeds (ascii code ESC-7). *Col* is particularly useful for filtering multicolumn output made with the '.rt' command of *nroff.*

SEE ALSO

nroff (I)

BUGS

Can't back up more than 102 lines.

-

NAME

cubic − three dimensional tic-tac-toe

SYNOPSIS

/usr/games/cubic

DESCRIPTION

Cubic plays the game of three dimensional 4×4×4 tic-tac-toe. Moves are given by the three digits (each 1-4) specifying the coordinate of the square to be played.

WARNING

Too much playing of the game will cause it to disappear.

BUGS

NAME

factor − discover prime factors of a number

SYNOPSIS

factor

DESCRIPTION

When *factor* is invoked without an argument, it waits for a number to be typed in. If you type in a positive number less than 2^{56} (about 7.2×10^{16}) it will factor the number and print its prime factors; each one is printed the proper number of times. Then it waits for another number. It exits if it encounters a zero or any non-numeric character.

If *factor* is invoked with an argument, it factors the number as above and then exits.

Maximum time to factor is proportional to $\sqrt{n}$ and occurs when n is prime or the square of a prime. It takes 1 minute to factor a prime near 10^{13}.

DIAGNOSTICS

'Ouch.' for input out of range or for garbage input.

BUGS

-

NAME

fed – edit form letter memory

SYNOPSIS

fed

DESCRIPTION

Fed is used to edit a form letter associative memory file, **form.m,** which consists of named strings. Commands consist of single letters followed by a list of string names separated by a single space and ending with a new line. The conventions of the Shell with respect to '*' and '?' hold for all commands but **m**. The commands are:

e name ...
Fed writes the string whose name is *name* onto a temporary file and executes *ed.* On exit from the *ed* the temporary file is copied back into the associative memory. Each argument is operated on separately. Be sure to give an editor *w* command (without a filename) to rewrite *fed's* temporary file before quitting out of *ed.*

d [name ...]
deletes a string and its name from the memory. When called with no arguments **d** operates in a verbose mode typing each string name and deleting only if a **y** is typed. A **q** response returns to *fed*'s command level. Any other response does nothing.

m name1 name2 ...
(move) changes the name of name1 to name2 and removes previous string name2 if one exists. Several pairs of arguments may be given. Literal strings are expected for the names.

n [name ...]
(names) lists the string names in the memory. If called with the optional arguments, it just lists those requested.

p name ...
prints the contents of the strings with names given by the arguments.

q
returns to the system.

c [**p**] [**f**]
checks the associative memory file for consistency and reports the number of free headers and blocks. The optional arguments do the following:

p causes any unaccounted-for string to be printed.

f fixes broken memories by adding unaccounted-for headers to free storage and removing references to released headers from associative memory.

FILES

/tmp/ftmp?	temporary
form.m	associative memory

SEE ALSO

form (VI), ed (I), sh (I)

WARNING

It is legal but unwise to have string names with blanks, '*' or '?' in them.

BUGS

-

NAME

form − form letter generator

SYNOPSIS

form proto arg ...

DESCRIPTION

Form generates a form letter from a prototype letter, an associative memory, arguments and in a special case, the current date.

If *form* is invoked with the *proto* argument *x*, the associative memory is searched for an entry with name *x* and the contents filed under that name are used as the prototype. If the search fails, the message '[*x*]:' is typed on the console and whatever text is typed in from the console, terminated by two new lines, is used as the prototype. If the prototype argument is missing, '{letter}' is assumed.

Basically, *form* is a copy process from the prototype to the output file. If an element of the form [*n*] (where *n* is a digit from 1 to 9) is encountered, the *n*-th *arg* is inserted in its place, and that argument is then rescanned. If [0] is encountered, the current date is inserted. If the desired argument has not been given, a message of the form '[*n*]:' is typed. The response typed in then is used for that argument.

If an element of the form [*name*] or {*name*} is encountered, the *name* is looked up in the associative memory. If it is found, the contents of the memory under this *name* replaces the original element (again rescanned). If the *name* is not found, a message of the form '[*name*]:' is typed. The response typed in is used for that element. The response is entered in the memory under the name if the name is enclosed in [ ]. The response is not entered in the memory but is remembered for the duration of the letter if the name is enclosed in { }. Brackets and braces may be nested.

In both of the above cases, the response is typed in by entering arbitrary text terminated by two new lines. Only the first of the two new lines is passed with the text.

If one of the special characters [{]}\ is preceded by a \, it loses its special character.

If a file named 'forma' already exists in the user's directory, 'formb' is used as the output file and so forth to 'formz'.

The file 'form.m' is created if none exists. Because form.m is operated on by the disc allocator, it should only be changed by using *fed,* the form letter editor, or *form.*

FILES

form.m associative memory
form? output file (read only)

SEE ALSO

fed (VI), roff (I)

BUGS

An unbalanced] or } acts as an end of file but may add a few strange entries to the associative memory.

-

NAME

graph − draw a graph

SYNOPSIS

graph [option] ... | plotter

DESCRIPTION

Graph with no options takes pairs of numbers from the standard input as abscissas and ordinates of a graph. The graph is written on the standard output to be piped to the plotter program for a particular device; see *plot* (VI). These plotters exist: *gsip,* for the GSI and other Diablo terminals; *tek,* for the Tektronix 4014 terminal; and *vt0* for the on-line storage scope.

The following options are recognized, each as a separate argument.

a Supply abscissas automatically (they are missing from the input); spacing is given by the next argument, or is assumed to be 1 if next argument is not a number. A second optional argument is the starting point for the automatic abscissa.

c Place character string given by next argument at each point.

d Omit connections between points. (Disconnect.)

g*n* Grid style:
n=0, no grid
n=1, axes only
n=2, complete grid (default).

s Save screen, don't erase before plotting.

x Next 1 (or 2) arguments are lower (and upper) *x* limits.

y Next 1 (or 2) arguments are lower (and upper) *y* limits.

h Next argument is fraction of space for height

w Next argument is fraction of space for width.

r Next argument is fraction of space to move right before plotting.

u Next argument is fraction of space to move up before plotting.

Points are connected by straight line segments in the order they appear in input. If a specified lower limit exceeds the upper limit, or if the automatic increment is negative, the graph is plotted upside down. Automatic abscissas begin with the lower *x* limit, or with 0 if no limit is specified. Grid lines and automatically determined limits fall on round values, however roundness may be subverted by giving an inappropriately rounded lower limit. Plotting symbols specified by **c** are placed so that a small initial letter, such as + o x, will fall approximately on the plotting point.

SEE ALSO

spline (VI), plot (VI)

BUGS

A limit of 1000 points is enforced silently.

NAME

gsi − interpret extended character set on GSI terminal

SYNOPSIS

gsi

DESCRIPTION

Gsi interprets special characters understood by the Model 37 Teletype terminal and turns them into the escape sequences understood by the GSI and other Diablo-based terminals. The things interpreted include vertical motions and extended graphic characters. It is most often used in a pipeline like

neqn file ... | nroff | gsi

SEE ALSO

greek (V)

BUGS

Some funny characters can’t be correctly printed in column 1 because you can’t move to the left from there.

NAME

m6 − general purpose macroprocessor

SYNOPSIS

m6 [name]

DESCRIPTION

M6 copies the standard input to the standard output, with substitutions for any macro calls that appear. When a file name argument is given, that file is read before the standard input.

The processor is as described in the reference with these exceptions:

#def,arg1,arg2,arg3: causes *arg1* to become a macro with defining text *arg2* and (optional) built-in serial number *arg3*.

#del,arg1: deletes the definition of macro *arg1*.

#end: is not implemented.

#list,arg1: sends the name of the macro designated by *arg1* to the current destination without recognition of any warning characters; *arg1* is 1 for the most recently defined macro, 2 for the next most recent, and so on. The name is taken to be empty when *arg1* doesn't make sense.

#warn,arg1,arg2: replaces the old warning character *arg1* by the new warning character *arg2*.

#quote,arg1: sends the definition text of macro *arg1* to the current destination without recognition of any warning characters.

#serial,arg1: delivers the built-in serial number associated with macro *arg1*.

#source,arg1: is not implemented.

#trace,arg1: with *arg1* = ‘1’ causes a reconstruction of each later call to be placed on the standard output with a call level number; other values of *arg1* turn tracing off.

The built-in ‘warn’ may be used to replace inconvenient warning characters. The example below replaces ‘#’ ‘:’ ‘<’ ‘>’ by ‘[’ ‘]’ ‘{’ ‘}’.

```
#warn,<#>,[:
[warn,<:>,]:
[warn,[substr,<<>>,1,1;,{]
[warn,[substr,{{>>,2,1;,}]
[now,{calls look like this}]
```

Every built-in function has a serial number, which specifies the action to be performed before the defining text is expanded. The serial numbers are: 1 gt, 2 eq, 3 ge, 4 lt, 5 ne, 6 le, 7 seq, 8 sne, 9 add, 10 sub, 11 mpy, 12 div, 13 exp, 20 if, 21 def, 22 copy, 23 warn, 24 size, 25 substr, 26 go, 27 gobk, 28 del, 29 dnl, 32 quote, 33 serial, 34 list, 35 trace. Serial number 0 specifies no built-in action.

SEE ALSO

A. D. Hall, M6 Reference Manual. Computer Science Technical Report #2, Bell Laboratories, 1969.

DIAGNOSTICS

Various table overflows and “impossible” conditions result in comment and dump. There are no diagnostics for poorly formed input.

AUTHOR

M. D. McIlroy

BUGS

Provision should be made to extend tables as needed, instead of wasting a big fixed core allocation. You get what the PDP11 gives you for arithmetic.

-

NAME

moo – guessing game

SYNOPSIS

/usr/games/moo

DESCRIPTION

Moo is a guessing game imported from England. The computer picks a number consisting of four distinct decimal digits. The player guesses four distinct digits being scored on each guess. A ‘cow’ is a correct digit in an incorrect position. A ‘bull’ is a correct digit in a correct position. The game continues until the player guesses the number (a score of four bulls).

BUGS

NAME

plot: tek, gsip, vt0 – graphics filters

SYNOPSIS

source **tek**
source **gsip**
source **vt0**

DESCRIPTION

These commands produce graphical output on the Tektronix 4014 terminal, the GSI or other Diablo-mechanism terminals, and the on-line storage scope respectively. They read the standard input to obtain plotting instructions, which are usually generated by a program calling the graphics subroutines described in *plot* (VII). Each instruction consists of an ASCII letter usually followed by binary information. A plotting coordinate is transmitted as four bytes representing the *x* and *y* values; each value is a signed number transmitted low-order byte first. The assumed plotting space is set by request. The instructions are taken from

m move: the next four bytes specify the coordinates of a point to move to. This is used before writing a label.

p point: the next four bytes specify the coordinates at which a point is drawn.

l line: the next eight bytes are taken as two pairs of coordinates specifying the endpoints of a line to be drawn.

t label: the bytes up to a new-line are written as ASCII starting at the last point drawn or moved to.

a arc: the first four bytes specify the center, the next four specify the starting point, and the last four specify the end point of a circular arc. The least significant coordinate of the end point is used only to determine the quadrant. The arc is drawn counter-clockwise. This command is not necessarily implemented on all (or even any) of the output devices.

c circle: The first four bytes specify the center of the circle, the next two the radius.

e erases the screen

f linemod: takes the following string as the type for all future lines. The types are 'dotted,' 'solid,' 'longdashed,' 'shortdashed,' and 'dotdashed.' This instruction is effective only with the Tektronix terminal.

d dotline: takes the first four bytes as the coordinates of the beginning of a dotted line. The next two are a signed x-increment, and the next two are a word count. Following are the indicated number of byte-pairs representing words. For each bit in this list of words a point is plotted which is visible if the bit is '1,' invisible if not. Each point is offset rightward by the x-increment. The instruction is effective only on the vt0 scope.

SEE ALSO

plot (VII), graph (VI)

BUGS

-

NAME

primes − print all primes larger than somewhat

SYNOPSIS

primes

DESCRIPTION

When *primes* is invoked, it waits for a number to be typed in. If you type in a positive number less than 2^{56} (about 7.2×10^{16}) it will print all primes greater than or equal to this number.

DIAGNOSTICS

'Ouch.' for input out of range or for garbage input.

BUGS

-

NAME

quiz − test your knowledge

SYNOPSIS

quiz [**−i** file] [**−t**] [category1 category2]

DESCRIPTION

Quiz gives associative knowledge tests on various subjects. It asks items chosen from *category1* and expects answers from *category2.* If no categories are specified, *quiz* gives instructions and lists the available categories.

Quiz tells a correct answer whenever you type a bare newline. At the end of input, upon interrupt, or when questions run out, *quiz* reports a score and terminates.

The **−t** flag specifies ‘tutorial’ mode, where missed questions are repeated later, and material is gradually introduced as you learn.

The **−i** flag causes the named file to be substituted for the default index file. The lines of these files have the syntax:

```
line        = category newline | category ‘:’ line
category    = alternate | category ‘|’ alternate
alternate   = empty | alternate primary
primary     = character | ‘[’ category ‘]’ | option
option      = ‘{’ category ‘}’
```

The first category on each line of an index file names an information file. The remaining categories specify the order and contents of the data in each line of the information file. Information files have the same syntax. Backslash ‘\’ is used as with *sh* (I) to quote syntactically significant characters or to insert transparent newlines into a line. When either a question or its answer is empty, *quiz* will refrain from asking it.

FILES

/usr/lib/quiz/index
/usr/lib/quiz/*

BUGS

NAME

sky − obtain ephemerides

SYNOPSIS

sky [**−l**]

DESCRIPTION

Sky predicts the apparent locations of the Sun, the Moon, the planets out to Saturn, stars of magnitude at least 2.5, and certain other celestial objects. *Sky* reads the standard input to obtain a GMT time typed on one line with blanks separating year, month number, day, hour, and minute; if the year is missing the current year is used. If a blank line is typed the current time is used. The program prints the azimuth, elevation, and magnitude of objects which are above the horizon at the ephemeris location of Murray Hill at the indicated time. The '−l' flag causes it to ask for another location.

Placing a "1" input after the minute entry causes the program to print out the Greenwich Sidereal Time at the indicated moment and to print for each body its topographic right ascension and declination as well as its azimuth and elevation. Also, instead of the magnitude, the semidiameter of the body, in seconds of arc, is reported.

A "2" after the minute entry makes the coordinate system geocentric.

The effects of atmospheric extinction on magnitudes are not included; the brightest magnitudes of variable stars are marked with "*".

For all bodies, the program takes into account precession and nutation of the equinox, annual (but not diurnal) aberration, diurnal parallax, and the proper motion of stars. In no case is refraction included.

The program takes into account perturbations of the Earth due to the Moon, Venus, Mars, and Jupiter. The expected accuracies are: for the Sun and other stellar bodies a few tenths of seconds of arc; for the Moon (on which particular care is lavished) likewise a few tenths of seconds. For the Sun, Moon and stars the accuracy is sufficient to predict the circumstances of eclipses and occultations to within a few seconds of time. The planets may be off by several minutes of arc.

There are lots of special options not described here, which do things like substituting named star catalogs, smoothing nutation and aberration to aid generation of mean places of stars, and making conventional adjustments to the Moon to improve eclipse predictions.

For the most accurate use of the program it is necessary to know that it actually runs in Ephemeris time.

FILES

/usr/lib/startab, /usr/lib/moontab

SEE ALSO

azel (VI)
American Ephemeris and Nautical Almanac, for the appropriate years; also, the *Explanatory Supplement to the American Ephemeris and Nautical Almanac.*

AUTHOR

R. Morris

BUGS

-

NAME

sno − Snobol interpreter

SYNOPSIS

sno [file]

DESCRIPTION

Sno is a Snobol III (with slight differences) compiler and interpreter. *Sno* obtains input from the concatenation of *file* and the standard input. All input through a statement containing the label ‘end’ is considered program and is compiled. The rest is available to ‘syspit’.

Sno differs from Snobol III in the following ways.

There are no unanchored searches. To get the same effect:

```
a ** b          unanchored search for b
a *x* b = x c   unanchored assignment
```

There is no back referencing.

```
x = "abc"
a *x* x         is an unanchored search for ‘abc’
```

Function declaration is different. The function declaration is done at compile time by the use of the label ‘define’. Thus there is no ability to define functions at run time and the use of the name ‘define’ is preempted. There is also no provision for automatic variables other than the parameters. For example:

define f( )

or

define f(a,b,c)

All labels except ‘define’ (even ‘end’) must have a non-empty statement.

If ‘start’ is a label in the program, program execution will start there. If not, execution begins with the first executable statement. ‘define’ is not an executable statement.

There are no builtin functions.

Parentheses for arithmetic are not needed. Normal precedence applies. Because of this, the arithmetic operators ‘/’ and ‘*’ must be set off by space.

The right side of assignments must be non-empty.

Either ´ or " may be used for literal quotes.

The pseudo-variable ‘sysppt’ is not available.

SEE ALSO

Snobol III manual. (JACM; Vol. 11 No. 1; Jan 1964; pp 21)

BUGS

NAME

speak − word to voice translator

SYNOPSIS

speak [−**efpsv**] [vocabulary [output]]

DESCRIPTION

Speak turns a stream of words into utterances and outputs them to a voice synthesizer, or to the specified *output.* It has facilities for maintaining a vocabulary. It receives, from the standard input

- working lines: text of words separated by blanks
- phonetic lines: strings of phonemes for one word preceded and separated by commas. The phonemes may be followed by comma-percent then a ‘replacement part’ − an ASCII string with no spaces. The phonetic code is given in *vs* (V).
- empty lines
- command lines: beginning with **!.** The following command lines are recognized:

!r file	replace coded vocabulary from file
!w file	write coded vocabulary on file
!p	print phonetics for working word
!l	list vocabulary on standard output with phonetics
!c word	copy phonetics from working word to specified word
!d	print decomposition of working word into substrings
!f *n*	turn off (or on) English preprocessing rule number *n* (see listing for meaning of *n)*

Each working line replaces its predecessor. Its first word is the ‘working word’. Each phonetic line replaces the phonetics stored for the working word. In particular, a phonetic line of comma only deletes the entry for the working word. Each working line, phonetic line or empty line causes the working line to be uttered. The process terminates at the end of input.

Unknown words are pronounced by rules, and failing that, are spelled. For the builtin part of the rules, see the reference. Spelling is done by taking each character of the word, prefixing it with ‘*’, and looking it up. Unspellable words burp.

Words not found verbatim in the vocabulary are pronounced piecewise. First the word is bracketed by sharps: ‘#...#’. The vocabulary is then searched for the longest fragment that matches the beginning of the word. The phonetic part of the phonetic string is uttered, and the matched fragment is replaced by the replacement part of the phonetic string, if any. The process is repeated until the word is exhausted. A fragment is entered into the vocabulary as a working word prefixed by ‘%’.

Speak is initialized with a coded vocabulary stored in file */usr/lib/speak.m.* The vocabulary option substitutes a different file for */usr/lib/speak.m.* Other vocabularies, to be used with option −**e**, exist in /usr/vs/latin.m and /usr/vs/polish.m.

A set of single letter options may appear in any order preceded by **−.** Their meanings are:

e	suppress English preprocessing
f	equivalent to ‘f1, f2,...’
p	suppress pronunciation by rule
s	suppress spelling
v	suppress voice output

The following input will reconstitute a coded vocabulary, ‘speak.m’, from an ascii listing, ‘speak.v’, that was created using **!l**.

```
(cat speak.v; echo !w speak.m) | speak −v /dev/null
```

FILES

/usr/lib/speak.m

SEE ALSO

M. D. McIlroy, ‘‘Synthetic English Speech by Rule,’’ Computing Science Technical Report #14, Bell Laboratories, 1973
vs (V), vs (IV)

BUGS

Excessively long words cause dumps.
Space is not reclaimed from changed entries; use **!w** and **!r** to effect reclamation.
!p doesn’t always work as advertised.

NAME

spline − interpolate smooth curve

SYNOPSIS

spline [option] ...

DESCRIPTION

Spline takes pairs of numbers from the standard input as abcissas and ordinates of a function. It produces a similar set, which is approximately equally spaced and includes the input set, on the standard output. The cubic spline output (R. W. Hamming, *Numerical Methods for Scientists and Engineers,* 2nd ed., 349ff) has two continuous derivatives, and sufficiently many points to look smooth when plotted, for example by *plot* (I).

The following options are recognized, each as a separate argument.

a Supply abscissas automatically (they are missing from the input); spacing is given by the next argument, or is assumed to be 1 if next argument is not a number.

k The constant k used in the boundary value computation

$$y_0'' = k y_1'', \quad y_n'' = k y_{n-1}''$$

is set by the next argument. By default $k = 0$.

n Space output points so that approximately n points occur between the lower and upper x limits. (Default $n = 100$.)

p Make output periodic, i.e. match derivatives at ends. First and last input values should normally agree.

x Next 1 (or 2) arguments are lower (and upper) x limits. Normally these limits are calculated from the data. Automatic abcissas start at lower limit (default 0).

SEE ALSO

plot (I)

AUTHOR

M. D. McIlroy

BUGS

A limit of 1000 input points is enforced silently.

-

NAME

tbl − format tables for nroff or troff

SYNOPSIS

tbl [files] ...

DESCRIPTION

Tbl is an nroff (I) or troff(I) preprocessor for formatting tables. The input files are copied to the standard output, except for lines between .TS and .TE command lines, which are assumed to describe tables and reformatted. The first line after .TS specifies the various columns: it consists of a list of column describers separated by blanks or tabs. Each column describer is a character string made up of the letters ‘n’, ‘r’, ‘c’, ‘l’ and ‘s’, which mean:

c center within the column

r right-adjust

l left-adjust

n numerical adjustment: the units digits of numbers are aligned.

s span the previous entry over this column.

The column describer may be followed by an integer giving the number of spaces between this column and the next; 3 is default. The describer ‘ccr5’ indicates that the first two lines in this column are centered; the third and remaining lines are right-adjusted; and the column should be separated from the column to the right by 5 spaces. Letting \t represent a tab (which must be typed as a genuine tab) the input

```
.TS
cccl sccn sscn
Household Population
Town\tHouseholds
\tNumber\tSize
Bedminster\t789\t3.26
Bernards Twp.\t3087\t3.74
Bernardsville\t2018\t3.30
Bound Brook\t3425\t3.04
Branchburg\t1644\t3.49
Bridgewater\t7897\t3.81
Far Hills\t240\t3.19
.TE
```

yields

Household Population		
Town	Households	
	Number	Size
Bedminster	789	3.26
Bernards Twp.	3087	3.74
Bernardsville	2018	3.30
Bound Brook	3425	3.04
Branchburg	1644	3.49
Bridgewater	7897	3.81
Far Hills	240	3.19

If no arguments are given, *tbl* reads the standard input, so it may be used as a filter. When it is used with *eqn* or *neqn* the *tbl* command should be first, to minimize the volume of data passed through pipes.

BUGS

No column describer may end with ‘s’.

NAME

tmg − compiler-compiler

SYNOPSIS

tmg name

DESCRIPTION

Tmg produces a translator for the language whose parsing and translation rules are described in file *name***.t**. The new translator appears in a.out and may be used thus:

a.out input [output]

Except in rare cases input must be a randomly addressable file. If no output file is specified, the standard output file is assumed.

FILES

*name***.s**: assembly language version of *name***.t**
/usr/lib/tmg: the compiler-compiler
/usr/lib/tmg[abc], /lib/libs.a: libraries
alloc.d: scratch file for table storage

SEE ALSO

A Manual for the Tmg Compiler-writing Language, internal memorandum.

DIAGNOSTICS

Syntactic errors result in "???" followed by the offending line.
Situations such as space overflow with which the Tmg processor or a Tmg-produced processor can not cope result in a descriptive comment and a dump.

AUTHOR

M. D. McIlroy

BUGS

Footnote 1 of Section 9.2 of Tmg Manual is not enforced, causing trouble.
Restrictions (7.) against mixing bundling primitives should be lifted.
Certain hidden reserved words exist: gpar, classtab, trans, goto, alt, salt.
Octal digits include 8=10 and 9=11.

NAME

ttt – the game of tic-tac-toe

SYNOPSIS

/usr/games/ttt

DESCRIPTION

Ttt is the X and O game popular in the first grade. This is a learning program that never makes the same mistake twice.

Although it learns, it learns slowly. It must lose nearly 80 games to completely know the game.

FILES

/usr/games/ttt.k learning file

BUGS

-

NAME

units – conversion program

SYNOPSIS

Units converts quantities expressed in various standard scales to their equivalents in other scales. It works interactively in this fashion:

You have: inch
You want: cm
** 2.54000e+00*
/ 3.93701e−01

A quantity is specified as a multiplicative combination of units optionally preceded by a numeric multiplier. Powers are indicated by suffixed positive integers, division by the usual sign:

You have: 15 pounds force/in2
You want: atm
** 1.02069e+00*
/ 9.79730e−01

Units only does multiplicative scale changes. Thus it can convert Kelvin to Rankine, but not Centigrade to Fahrenheit. Most familiar units, abbreviations, and metric prefixes are recognized, together with a generous leavening of exotica and a few constants of nature including:

pi	ratio of circumference to diameter
c	speed of light
e	charge on an electron
g	acceleration of gravity
force	same as g
mole	Avogadro's number
water	pressure head per unit height of water
au	astronomical unit

'Pound' is a unit of mass. Compound names are run together, e.g. 'lightyear'. British units that differ from their US counterparts are prefixed thus: 'brgallon'. For a complete list of units, 'cat /usr/lib/units'.

FILES

/usr/lib/units

BUGS

NAME

wump – the game of hunt-the-wumpus

SYNOPSIS

/usr/games/wump

DESCRIPTION

Wump plays the game of ‘‘Hunt the Wumpus.’’ A Wumpus is a creature that lives in a cave with several rooms connected by tunnels. You wander among the rooms, trying to shoot the Wumpus with an arrow, meanwhile avoiding being eaten by the Wumpus and falling into Bottomless Pits. There are also Super Bats which are likely to pick you up and drop you in some random room.

The program asks various questions which you answer one per line; it will give a more detailed description if you want.

This program is based on one described in *People's Computer Company, 2,* 2 (November 1973).

BUGS

It will never replace Space War.