**NAME**

azel – obtain satellite predictions

**SYNOPSIS**

**azel** satellite ...

**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.

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 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, ... sll — Skylab A through Skylab L. Skylabs A and B are the laboratory and its rocket respectively; the remainder are various other objects attendant upon its launch and subsequent activities. A, B, and probably K have been sighted and 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). The author should be consulted for more information.

**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**

Be careful of the random number generator.

**NAME**

cal – print calendar

**SYNOPSIS**

**cal** [ month ] year

**DESCRIPTION**

*Cal* will print 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 and is not required; '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 and the elapsed time in seconds.

**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 has been known to cause it to go away.

**AUTHOR**

K. Thompson

**BUGS**

Pawns may be promoted only to queens.

-



**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, it types out 'Enter:' at you. If you type in a positive number less than $2^{56}$ (about $7.2\times10^{16}$) it will repeat the number back at you and then its prime factors each one printed the proper number of times. Then it says 'Enter:' again. To exit, feed it an EOT or a delete.

Maximum time to factor is proportional to $\sqrt{n}$ and occurs when *n* is 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**

hyphen – find hyphenated words

**SYNOPSIS**

**hyphen** file ...

**DESCRIPTION**

It finds all of the words in a document which are hyphenated across lines and prints them back at you in a convenient format.

If no arguments are given, the standard input is used. Thus *hyphen* may be used as a filter.

**BUGS**

Yes, it gets confused, but with no ill effects other than spurious extra output.

**NAME**

m6 – general purpose macro processor

**SYNOPSIS**

**m6** [ –**d** arg1 ] [ arg2 [ arg3 ] ]

**DESCRIPTION**

*M6* takes input from file arg2 (or standard input if arg2 is missing) and places output on file arg3 (or standard output). A working file of definitions, ‘‘m.def’’, is initialized from file arg1 if that is supplied. *M6* differs from the standard [1] in these respects:

#trace:, #source: and #end: are not defined.

#meta,arg1,arg2: transfers the role of metacharacter arg1 to character arg2. If two metacharacters become identical thereby, the outcome of further processing is not guaranteed. For example, to make [ ]{} play the roles of #:<> type

```
\#meta,<\#>,[:
[meta,<:>,]:
[meta,[substr,<<>>,1,1;,{]
[meta,[substr,{{>>,2,1;,}]
```

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

#save: and #rest: save and restore the definition table together with the current metacharacters on file m.def.

#def,arg1,arg2,arg3: works as in the standard with the extension that an integer may be supplied to arg3 to cause the new macro to perform the action of a specified builtin before its replacement text is evaluated. Thus all builtins except #def: can be retrieved even after deletion. Codes for arg3 are:

0 – no function
1,2,3,4,5,6 – gt,eq,ge,lt,ne,le
7,8 – seq,sne
9,10,11,12,13 – add,sub,mpy,div,exp
20 – if
21,22 – def,copy
23 – meta
24 – size
25 – substr
26,27 – go,gobk
28 – del
29 – dnl
30,31 – save,rest

**FILES**

| | |
|---|---|
| m.def | working file of definitions |
| /usr/lang/mdir/m6a | m6 processor proper (/usr/bin/m6 is only an initializer) |
| /usr/lang/mdir/m6b | default initialization for m.def |
| /bin/cp | used for copying initial value of m.def |

**SEE ALSO**

[1] A. D. Hall, The M6 Macroprocessor, Bell Telephone Laboratories, 1969

**DIAGNOSTICS**

‘‘err’’ – a bug, an unknown builtin or a bad definition table
‘‘oprd’’–can’t open input or initial definitions
‘‘opwr’’–can’t open output
‘‘ova’’ – overflow of nested arguments
‘‘ovc’’ – overflow of calls
‘‘ovd’’ – overflow of definitions

‘‘Try again’’ – no process available for copying m.def

**AUTHOR**

M. D. McIlroy

**BUGS**

Characters in internal tables are stored one per word. They really should be packed to improve capacity. For want of space (and because of unpacked formats) no file arguments have been provided to #save: or #rest:, and no check is made on the actual opening of file m.def. Again to save space, garbage collection makes calls on #save: and #rest: and so overwrites m.def.

Since the program is written in the defunct language B it is currently unavailable. Expressions of interest may make a C version appear.

-



**NAME**

maze – generate a maze problem

**SYNOPSIS**

**maze**

**DESCRIPTION**

*Maze* will ask a few questions and then print out a maze.

**BUGS**

Some mazes (especially small ones) have no solutions.

-



**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**

Watch out for the random number generator.

**NAME**

ov – overlay pages

**SYNOPSIS**

**ov** [ file ]

**DESCRIPTION**

*Ov* is a postprocessor for producing double column formatted text when using nroff(I). *Ov* literally overlays successive pairs of 66-line pages.

If the file argument is missing, the standard input is used. Thus *ov* may be used as a filter.

**SEE ALSO**

nroff(I), pr(I)

**BUGS**

**NAME**

ptx – permuted index

**SYNOPSIS**

**ptx** [ –**t** ] input [ output ]

**DESCRIPTION**

*Ptx* generates a permuted index from file *input* on file *output.* It has three phases: the first does the permutation, generating one line for each keyword in an input line. The keyword is rotated to the front. The permuted file is then sorted. Finally the sorted lines are rotated so the keyword comes at the middle of the page.

*Input* should be edited to remove useless lines. The following words are suppressed: ‘a’, ‘an’, ‘and’, ‘as’, ‘is’, ‘for’, ‘of’, ‘on’, ‘or’, ‘the’, ‘to’, ‘up’.

The optional argument –**t** causes *ptx* to prepare its output for the phototypesetter.

The index for this manual was generated using *ptx.*

**FILES**

/bin/sort

**NAME**

sfs – structured file scanner

**SYNOPSIS**

**sfs** filename [ – ]

**DESCRIPTION**

*Sfs* provides an interactive program for scanning and pactching a structured file. If the second argument is supplied, the file is block addressed.

Some features of *sfs* include.

1. It provides interactive and preprogramed operation.
2. It provides expression evaluation (32 bit precision) and branching.
3. It provides the ability to assimulate a large set of heirarchical structure definitions.
4. It provides the ability to locate, to dump, and to patch specific instances of structure in the file. Furthermore, in the dump and patch operations the external form of the structure is selected by the user.
5. It provides the ability to escape to the UNIX command level to allow the use of other UNIX debugging aids.

**SEE ALSO**

‘‘SFS reference manual’’ (internal memorandum)

**BUGS**

**NAME**

sky – obtain ephemerides

**SYNOPSIS**

**sky**

**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 including comet Kohoutek and M31. *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.

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 right ascension and declination as well as its azimuth and elevation. Also, instead of the magnitude, the geocentric distance of the body, in units the program considers convenient, is printed. (For planets the unit is essentially A. U.)

The magnitudes of Solar System bodies are not calculated and are given as 0. The effects of atmospheric extinction are not included; the mean 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 (but not annual parallax). 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.

Information about the program may be obtained from its author.

**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

**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 Engineers and Scientists*, 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.

**n** Output approximately *n* points, where *n* is given by the next argument. (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.

**SEE ALSO**

plot(I)

**AUTHOR**

M. D. McIlroy

**BUGS**

A limit of 1000 input points is enforced silently.

**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**

| | |
|---|---|
| /sys/tmg/tmgl.o | the compiler-compiler |
| /sys/tmg[abc] | libraries |
| alloc.d | 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**

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

**NAME**

ttt – 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**

ttt.k learning file

**BUGS**

**NAME**

wump – 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.

**NAME**

yacc – yet another compiler-compiler

**SYNOPSIS**

**yacc** [ grammar ]

**DESCRIPTION**

*Yacc* converts a context-free grammar into a set of tables for a simple automaton which executes an LR(1) parsing algorithm.

For complete information, see the author.

**SEE ALSO**

"LR Parsing", by A. V. Aho and S. C. Johnson.

**AUTHOR**

S. C. Johnson

**BUGS**