NAME

apl – APL interpreter

SYNOPSIS

apl

DESCRIPTION

Apl is an interpreter for the language APL described in the reference. The interpreter maintains its workspace on disk rather than in core. This has two consequences: there is the potential of a million byte workspace; it takes a week to access that much data.

Not Implemented (never)

1. Lamination (except for scalar,scalar)
2. 0 div 0 is a domain error.
3. 0 mod x is a domain error.
4. No function definition – use ‘)edit fname’ to enter the system editor; type ‘‘w’’ when done editing to write the function out in a place where apl can pick it up. Type ‘‘w file’’ to save it.
5. Indexing is off in character vectors containing overstrikes.

Under Implementation (later)

1. Negative numbers raised to fractional powers are handled incorrectly.
2. No trace or SI.
3. Incomplete set of I-beams and system calls.

Over Implemented (over zealous)

1. Ravel[i] – obvious extension of cat.
2. Grade up and grade down extend to matrices.
3. Arbitrary overstriking is allowed in characters.

FILES

/usr/lib/apl/* programs
alloc.d workspace
apl_ed editor intermediate

SEE ALSO

IBM GH20-0906-1 ‘‘APL User’s Manual’’
/usr/pub/apl ASCII APL character set

BUGS

NAME

azel – obtain 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 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).

FILES

/usr/jfo/el/* – orbital element files

SEE ALSO

sky (VI)

AUTHOR

J. F. Ossanna

-

BUGS

NAME

bas – basic

SYNOPSIS

bas [file]

DESCRIPTION

Bas is a dialect of Basic. If a file argument is provided, the file is used for input before the console is read. *Bas* accepts lines of the form:

statement
integer statement

Integer numbered statements (known as internal statements) are stored for later execution. They are stored in sorted ascending order. Non-numbered statements are immediately executed. The result of an immediate expression statement (that does not have '=' as its highest operator) is printed.

Statements have the following syntax:

expression
The expression is executed for its side effects (assignment or function call) or for printing as described above.

done
Return to system level.

draw expression expression expression
A line is drawn on the Tektronix 611 display '/dev/vt0' from the current display position to the XY co-ordinates specified by the first two expressions. The scale is zero to one in both X and Y directions. If the third expression is zero, the line is invisible. The current display position is set to the end point.

display list
The list of expressions and strings is concatenated and displayed (i.e. printed) on the 611 starting at the current display position. The current display position is not changed.

dump
The name and current value of every variable is printed.

erase
The 611 screen is erased.

for name = expression expression statement
for name = expression expression
...
next
The *for* statement repetitively executes a statement (first form) or a group of statements (second form) under control of a named variable. The variable takes on the value of the first expression, then is incremented by one on each loop, not to exceed the value of the second expression.

goto expression
The expression is evaluated, truncated to an integer and execution goes to the corresponding integer numbered statment. If executed from immediate mode, the internal statements are compiled first.

if expression statement
The statement is executed if the expression evaluates to non-zero.

list [expression [expression]]
is used to print out the stored internal statements. If no arguments are given, all internal statements are printed. If one argument is given, only that internal statement is listed. If two arguments are given, all internal statements inclusively between the arguments are printed.

print list
The list of expressions and strings are concatenated and printed. (A string is delimited by " characters.)

prompt list
Prompt is the same as *print* except that no newline character is printed.

return [expression]
The expression is evaluated and the result is passed back as the value of a function call. If no expression is given, zero is returned.

run
The internal statements are compiled. The symbol table is re-initialized. The random number generator is reset. Control is passed to the lowest numbered internal statement.

save [expression [expression]]
Save is like *list* except that the output is written on the *file* argument. If no argument is given on the command, **b.out** is used.

Expressions have the following syntax:

name
A name is used to specify a variable. Names are composed of a letter followed by letters and digits. The first four characters of a name are significant.

number
A number is used to represent a constant value. A number is written in Fortran style, and contains digits, an optional decimal point, and possibly a scale factor consisting of an **e** followed by a possibly signed exponent.

(expression **)**
Parentheses are used to alter normal order of evaluation.

_ expression
The result is the negation of the expression.

expression operator expression
Common functions of two arguments are abbreviated by the two arguments separated by an operator denoting the function. A complete list of operators is given below.

expression **(** [expression [**,** expression] ...] **)**
Functions of an arbitrary number of arguments can be called by an expression followed by the arguments in parentheses separated by commas. The expression evaluates to the line number of the entry of the function in the internally stored statements. This causes the internal statements to be compiled. If the expression evaluates negative, a builtin function is called. The list of builtin functions appears below.

name **[** expression [**,** expression] ... **]**
Each expression is truncated to an integer and used as a specifier for the name. The result is syntactically identical to a name. **a[1,2]** is the same as **a[1][2].** The truncated expressions are restricted to values between 0 and 32767.

The following is the list of operators:

=
= is the assignment operator. The left operand must be a name or an array element. The result is the right operand. Assignment binds right to left, all other operators bind left to right.

& |
& (logical and) has result zero if either of its arguments are zero. It has result one if both its arguments are non-zero. | (logical or) has result zero if both of its arguments are zero. It has result one if either of its arguments are non-zero.

< <= > >= == <>
: The relational operators (< less than, <= less than or equal, > greater than, >= greater than or equal, == equal to, <> not equal to) return one if their arguments are in the specified relation. They return zero otherwise. Relational operators at the same level extend as follows: a>b>c is the same as a>b&b>c.

+ –
: Add and subtract.

* /
: Multiply and divide.

^
: Exponentiation.

The following is a list of builtin functions:

arg(i)
: is the value of the *i* -th actual parameter on the current level of function call.

exp(x)
: is the exponential function of *x*.

log(x)
: is the natural logarithm of *x*.

sqr(x)
: is the square root of *x*.

sin(x)
: is the sine of *x* (radians).

cos(x)
: is the cosine of *x* (radians).

atn(x)
: is the arctangent of *x*. Its value is between $-\pi/2$ and $\pi/2$.

rnd( )
: is a uniformly distributed random number between zero and one.

expr( )
: is the only form of program input. A line is read from the input and evaluated as an expression. The resultant value is returned.

int(x)
: returns *x* truncated to an integer.

FILES

/tmp/btm?	temporary
b.out	save file

DIAGNOSTICS

Syntax errors cause the incorrect line to be typed with an underscore where the parse failed. All other diagnostics are self explanatory.

BUGS

Has been known to give core images.

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

catsim – phototypesetter simulator

SYNOPSIS

catsim

DESCRIPTION

Catsim will interpret its standard input as codes for the phototypesetter (cat). The output of *catsim* is output to the display (vt).

About the only use of *catsim* is to save time and paper on the phototypesetter by the following command:

troff –t files ... | catsim

FILES

/dev/vt0

SEE ALSO

troff (I), cat (IV), vt (IV)

BUGS

Point sizes are not correct. The vt character set is restricted to one font of ASCII.

-

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.

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.

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, it types out 'Enter:' at you. If you type in a positive number less than 2^{56} (about 7.2×10^{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 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

graf – draw graph on GSI terminal

SYNOPSIS

graf | nroff | gsi

DESCRIPTION

Graf is a preprocessor to *nroff* (q.v.) for producing plots imbedded in documents. The standard input is copied to the standard output except for plots, which are inserted whenever a line beginning ‘‘.GR’’ is found. The remainder of the line should be the arguments to *plog*, normally including a ‘‘<file’’ to point off to the data for the graph. *Graf* itself reads its standard input.

Graf and *neqn* can be used together:

neqn files | graf | nroff | gsi

produces a memo with figures and equations and text all intermixed. There is no typesetter equivalent of *graf*, nor are there any plans for one.

SEE ALSO

plog (VI), neqn (I), nroff (I), gsi (VI)

BUGS

Same as *plog* (VI). Axes and labels are not scaled down correctly for smaller graphs. It should recognize .so commands.

NAME

gsi – interpret extended character set on GSI terminal

SYNOPSIS

gsi

DESCRIPTION

Gsi interprets commands specific to the GSI terminal. It converts half line forward and reverse motions into the right vertical motion. It also attempts to draw Greek letters and other special symbols of the Model 37 extended character set. (These are normally preceded by shift-out and followed by shift-in.) *Gsi* is most often used to print equations neatly, in the sequence

neqn file ... | nroff | gsi

Gsi also interprets the plot control characters ACK and BEL. This makes it useful in the sequence

graf | nroff | gsi

FILES

SEE ALSO

neqn (I), graf (VI), greek (VII)

BUGS

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

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

ibm – submit off-line job to HO IBM 370

SYNOPSIS

ibm [–**j**] file ...

DESCRIPTION

Ibm arranges to have the 201 data phone daemon submit a job to the IBM 370 at Holmdel via the Murray Hill H6070. Normally the job is submitted with enough ‘‘JCL’’ (the IBM version of the shell) to return the output to your box at Murray Hill. You can supply your own if you dare– the –**j** option suppresses all JCL.

If there are no arguments, the standard input is read and submitted. Thus *ibm* may be used as a filter.

FILES

/usr/dpd/*	spool area
/etc/passwd	personal ident cards
/etc/dpd	daemon

SEE ALSO

dpd (VIII), passwd (V)

BUGS

Stuff is sent on 6-bit cards, so lower case vanishes, as do some of the special characters.

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

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

NAME

npr – print file on Spider line-printer

SYNOPSIS

npr file ...

DESCRIPTION

Npr prints files on the line-printer in the Spider room. sending them over the Spider loop.

If there are no arguments, the standard input is read and submitted. Thus *npr* may be used as a filter.

FILES

/dev/tiu/d2 tiu to loop

BUGS

NAME

plog – make a graph on the gsi terminal

SYNOPSIS

plog [option] ...

DESCRIPTION

Plog is almost the same as *plog* (q.v.) but the plot is written on the standard output using the control sequences for the GSI terminal. The following changes have been made:

- The default for grid is no grid at all.
- The ‘a’ option can be followed by two arguments; the second is the starting point for automatic abscissas.
- There is a new option ‘h’ which must be followed by a numerical argument: it specifies the height desired for the plot.
- There is a new option ‘w’ similar to ‘h’, except that the width is specified. If only on of ‘h’ and ‘w’ is given, the plot is made square of the indicated size. If neither is given, the plot is made six inches square.
- There is a new option ‘r’ to be followed by a number which loates the plot that many inches to the right on the page.

SEE ALSO

plot (VI)

BUGS

Same as plot (VI). Drawing lines is not yet done exactly right. If you store the output in a file, before printing with cat you must turn off delays and turn off CR-NL echo (e.g. ‘‘stty -delay nl’’);

NAME

plot – make a graph

SYNOPSIS

plot [option] ...

DESCRIPTION

Plot takes pairs of numbers from the standard input as abscissas and ordinates of a graph. The graph is plotted on the storage scope, /dev/vt0.

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.

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.

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.

FILES

/dev/vt0

SEE ALSO

spline (VI), plog (VI)

BUGS

A limit of 1000 points is enforced silently.

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 preprogrammed operation.
2. It provides expression evaluation (32 bit precision) and branching.
3. It provides the ability to define a large set of hierarchical 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 *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.

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 [–**epsv**] [vocabulary [output]]

DESCRIPTION

Speak turns a stream of words into utterances and outputs them to a voice synthesizer, or to the specified output file. 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 bs (VII).
- 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 into substrings

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. Spelling is done by taking each character of the word, prefixing it with '*', and looking it up. Unspellable words burp.

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.*

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

- **–e** suppress English preprocessing
- **–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**. 'Null' names a nonexistent vocabulary file.

```
cat speak.v – | speak –v null
!w speak.m
```

FILES

/usr/lib/speak.m

SEE ALSO

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

BUGS

Excessively long words cause dumps.
Space is not reclaimed from changed entries; use **!w** and **!r** to effect reclamation.
The first phoneme is sometimes dropped when **!p** is used after **!d**.

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

$$y_0'' = ky_1'', \quad y_n'' = ky_{n-1}''$$

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

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

/usr/games/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 [–**v**] [grammar]

DESCRIPTION

Yacc converts a context-free grammar into a set of tables for a simple automaton which executes an LR(1) parsing algorithm. The grammar may be ambiguous; specified precedence rules are used to break ambiguities.

The output is *y.tab.c*, which must be compiled by the C compiler and loaded with any other routines required (perhaps a lexical analyzer) and the Yacc library:

```
cc y.tab.c other.o –ly
```

If the –**v** flag is given, the file *y.output* is prepared, which contains a description of the parsing tables and a report on conflicts generated by ambiguities in the grammar.

SEE ALSO

''LR Parsing'', by A. V. Aho and S. C. Johnson, Computing Surveys, June, 1974. ''The YACC Compiler-compiler'', internal memorandum.

AUTHOR

S. C. Johnson

FILES

y.output	
y.tab.c	
/lib/liby.a	runtime library for compiler

DIAGNOSTICS

The number of reduce-reduce and shift-reduce conflicts is reported on the standard output; a more detailed report is found in the *y.output* file.

BUGS