# PDP-1 LISP

A program has been written for the PDP-1 providing a subset of the features of the LISP interpreter for the IBM 709/7090. This program, which contains no known bugs, will run on any PDP-1 with automatic divide. On machines with more than 4K of memory, it must be run in memory field 0.

It is assumed that the reader is familiar with 709 LISP in general and with the LISP 1.5 Programmer's Manual in particular.

## Input/Output

Input comes from the typewriter if SS 5 is up and from the reader otherwise. Output is normally on the typewriter: however, SS 3 up causes punching (with correct parity) and SS 6 up independently suppresses typeout.

Each S-expression typed in will be evaluated and its value printed out. Unlike 709 LISP, arguments of functions are also evaluated on the top level: thus, to evaluate

cons[A;B]

it is necessary to write

(cons (quote a) (quote b))

In the input:

Tab, space, and comma are equivalent;

Carriage return is ignored;

Backspace causes deletion of everything typed since the last control character (parenthesis, space/tab/comma, or period);

An extra space must be typed to terminate the entire expression;

Upper and lower case shifts will be noted but not necessarily inserted into the symbol at that point (for example, the sequence u.c., l.c., u.c., A, space, produces a symbol with print name u.c., A, l.c.);

It is very advisable to stick to "printout" format for all input since the READ routine is not guaranteed to work on any other form, although it may;

"-" is a letter and does not negate a following number;

All numbers are octal integers: to input the number -1 it is necessary to type 777776;

There is no limit on the length of a print name;

The character "‾" or "|" will cause the next character to be inserted in the print name and considered a letter, regardless of what it actually is (the "‾" or "|" itself does not appear in the print name): thus atoms may be generated for output formatting purposes with names such as "tab" or "space".

In the output:

A carriage return is automatically generated after any 100(octal) characters not containing a carriage return;

Unlike the 709 LISP output, no spaces are provided before and after the "." of concatenation (since there are no floating point numbers to worry about).

## Available Functions

The following functions are available and identical to their 709 counterparts:

| | | |
|---|---|---|
| ATOM | CAR | CDR |
| COND | CONS | GENSYM |
| LIST | MINUS | NUMBERP |
| QUOTIENT | RPLACA | RPLACD |
| PROG | RETURN | GO |
| SETQ | SASSOC | READ |
| EVAL | QUOTE | NULL |
| TERPRI | | |

The following functions are available and somewhat different from the corresponding 709 functions:

EQ also works on numbers;<br>
GREATERP tests for x>y, not x≥y;<br>
STOP is equivalent to 709 PAUSE and takes a numeric argument which appears in the AC when the machine halts;<br>
PRINT has value NIL.

The following function has no counterpart on the 709:

PRIN1[x] prints the atom x without the extra space at the end: value is NIL

The following special forms are available and are the same as the corresponding forms on the 709:

| | |
|---|---|
| LAMBDA | LABEL |

The following permanent objects exist in the system:

OBLIST the current atomic symbol list;<br>
ALIST the current bound variable pair-list;<br>
NIL<br>
T<br>
F has been replaced by NIL<br>
EXPR<br>
SUBR<br>
FSUBR<br>
APVAL

## Operation of the System

READ IN the binary tape. Do nothing until the machine has come to a halt. If it stops anywhere except at 4, pull the tape back a block check for missing holes, and CONTINUE When it stops at 4, CONTINUEing begins the READ-EVAL-PRINT cycle. STARTing at 4 at any time and CONTINUEing is safe: indeed, it is the only way to annull most typing errors. If the system "drops dead", dump as much of core between 4 and 775' as is possible and try STARTing at 4. If further difficulties ensue (which is likely), notify Donald Eastlake (MIT. room 26-260, extension 4'67) at once and provide him with your name, the dump, and all pertinent typed or punched material

## Error Halts

Error halts cause identification of the error in red on the typewriter, regardless of the settings of SS 3 and 6, and usually send the system to 4. A list follows.

"icd" Illegal COND: returns value NIL and continues.
"uss" Unbound symbol in SETQ: returns NIL and continues.
"tma" Too many arguments for a SUBR (more than 3): ignores extras and proceeds.
"uas" Unbound atomic symbol (followed by form currently being evaluated).
"ilp" Illegal parity: halts with character in AC: CONTINUE ignores character, but SS 5 may be turned up to provide a replacement if desired.
"lts" LAMBDA variable list too short.
"ats" Argument list (paired with LAMBDA list) too short.
"sce" Storage capacity exceeded. CONTINUEing is not advisable, as it will probably provoke the same error again in short order.
"pce" Pushdown capacity exceeded.
"nna" Non-numeric argument for arithmetic, followed by argument in question: returns value zero and proceeds.
"ana" Argument not atom (for PRIN1): returns NIL as usual and proceeds.
"ovf" Division overflow: returns zero and proceeds.

## Odds and Ends

PNAMEs are not part of property lists. A quick examination of listings of the system will show exactly where they are.

Doing a CDR of an atom is quite ok and will get the atom's property list. Doing a CAR of an atom may very easily wreck the system.

Do not forget that FEXPR does not exist.

QUOTE should be used in place of FUNCTION. This may require a bit of extra care in defining functions with functional arguments.

It is advisable to use PROG to avoid recursion wherever possible, even though it may take more space. The pushdown list will permit recursion to about 60(decimal) levels. If this is intolerably small, reassemble the system.

The system has about 900 LISP registers of free storage and 250 registers of half-word space (replaces full-word space).

Questions should be addressed first to the listings of the system then to Donald Eastlake, then, if all else fails, to me.

Peter Deutsch

The following additions have been made to PDP-1 Lisp:

1. plus (fsubr, uses octal)
2. times (fsubr, uses octal)
3. logand (fsubr)
4. logor (fsubr)
5. fexpr (works except that the second variable of the lambda is not bound to the alist)