```
************************************************
***** CHAPTER 10 ***** FUNCTIONS ON STRINGS *****
************************************************
```

THESE FUNCTIONS PERFORM VARIOUS OPERATIONS ON STRINGS. WHERE STRINGS OR NUMBERS ARE REQUIRED AS ARGUMENTS, ERRORS WILL OCCUR IF SUCH REQUIREMENTS ARE NOT MET.

CATENATE (LAMBDA X ---

CATENATE DOES FOR STRINGS WHAT APPEND DOES FOR LISTS; THE ARGUMENTS ARE STRUNG TOGETHER AND MADE INTO A NEW STRING.

```
EXAMPLES:       (CATENATE ,THIS IS A STRING, ,AND THIS IS TOO,)
                          = ,THIS IS A STRINGAND THIS IS TOO,
                (CATENATE ,BARF , ,QUUX , ,FOOBARZ,) = ,BARF QUUX FOOBARZ,
                (CATENATE ,ONLY ONE,) = ,ONLY ONE,
                (CATENATE) = ,,
```

REVSTR (LAMBDA (X) ---

REVSTR DOES FOR STRINGS WHAT REVERSE DOES FOR LISTS; THE CHARACTERS ARE REVERSED AND MADE INTO A NEW STRING.

```
EXAMPLE:        (REVSTR ,GUY THE GREAT QUUX,) = ,XUUQ TAERG EHT YUG,
```

STRLENGTH (LAMBDA (X) ---

STRLENGTH DOES FOR STRINGS WHAT LENGTH DOES FOR LISTS; THE NUMBER OF CHARACTERS IN THE STRING IS RETURNED.

```
EXAMPLES:       (STRLENGTH ,VIOLINIST,) = 9
                (STRLENGTH ,,) = 0
```

STRINDEX (LAMBDA (X Y) ---

STRINDEX SEARCHES FOR AN OCCURRENCE OF THE STRING Y IN THE STRING X. IF SUCH AN OCCURRENCE IS FOUND, THE NUMBER OF THE FIRST MATCHING CHARACTER IN X IS RETURNED; OTHERWISE 0 IS RETURNED.

```
EXAMPLES:       (STRINDEX ,THIS IS A STRING, ,IS,) = 6
                (STRINDEX ,THIS IS A STRING, , THIS,) = 0
                (STRINDEX , THIS IS A STRING, , THIS,) = 1
                (STRINDEX ,THIS IS A STRING, ,QUUX,) = 0
                (STRINDEX ,, X) = 0                FOR ANY STRING X.
```

SUBSTR (LAMBDA X Y . Z) ---

SUBSTR TAKES EITHER TWO OR THREE ARGUMENTS; THE FIRST IS A STRING, THE OTHER(S) ARE NUMBERS. IF ONLY TWO ARGUMENTS ARE GIVEN, SUBSTR CHOPS THE FIRST (Y-1) CHARACTERS OFF X AND RETURNS THE RESULT AS A STRING. IF THREE ARE GIVEN, SUBSTR CHOPS OFF THE FIRST (Y-1) CHARACTERS OFF X, THEN TAKES THE NEXT Z CHARACTERS AND RETURNS THEM AS A STRING. IF Y>(STRLENGTH X) THEN THE NULL STRING ,, IS RETURNED. IF (PLUS Y Z -1)>(STRLENGTH X) THEN

ONLY AS MANY CHARACTERS AS ARE PRESENT ARE RETURNED. IF Y<1 THEN Y IS ASSUMED TO BE 1. IF Z<1 THEN THE NULL STRING ,, IS RETURNED.

```
EXAMPLES:       (SUBSTR ,THIS IS A STRING, 8) = ,A STRING,
                (SUBSTR ,THIS IS A STRING, 8 1) = ,A,
                (SUBSTR ,THIS IS A STRING, 8 5) = ,A STR,
                (SUBSTR ,THIS IS A STRING, 5 43) = ,IS A STRING,
                (SUBSTR ,THIS IS A STRING, 43 6) = ,,
                (SUBSTR ,THIS IS A STRING, Ø 12) = ,THIS IS A ST,
                (SUBSTR ,THIS IS A STRING, 6 -1) = ,,
```

```
PNAME      (LAMBDA (X) ---
```

X SHOULD BE AN IDENTIFIER (THIS IS NOT CHECKED). THE PRINT NAME OF X IS RETURNED AS A STRING.

```
EXAMPLES:       (PNAME 'FOOBAR) = ,FOOBAR,
                (PNAME '&(QUUX&)) = ,(QUUX),
                (PNAME 'ATOM& NAME) = ,ATOM NAME,
                (PNAME '&'SPEED& FREAK& &&& VIOLINIST&')
                          = ,'SPEED FREAK & VIOLINIST',
*************************************
***** CHAPTER 11 ***** PROGRAMS *****
*************************************
```

THE LISP "PROGRAM FEATURE" OR "PROG FEATURE" ALLOWS THE USER TO WRITE ALGOL-LIKE SEQUENCES OF STATEMENTS WITH PROGRAM VARIABLES AND LABELS.

PROG (NLAMBDA (VARS . S) ---

PROG IS A FUNCTION WHICH TAKES AS ARGUMENTS A LIST OF VARIABLES AND A STATEMENT BODY OF TAGES (WHICH ARE IDENTIFIERS) AND STATEMENTS (WHICH ARE NON-ATOMIC S-EXPRESSIONS). PROG BINDS EACH OF THE VARIABLES TO THE VALUE NIL (SAVING THEIR OLD VALUES) AND EVALUATES THE STATEMENTS SEQENTIALLY UNTIL A CALL TO THE "GO" OR "RETURN" FUNCTION IS MADE, OR UNTIL THE STATEMENT BODY IS EXHAUSTED, IN WHICH CASE THE VALUE OF THE PROG EXPRESSION IS NIL.

RETURN (LAMBDA (X) ---

RETURN FORCES THE PROG CONTAINING IT TO EXIT IMMEDIATELY WITH THE VALUE X. IF THERE IS NO CONTAINING PROG AN ERROR OCCURS.

GO (NLAMBDA (X) ---

GO CAUSES THE PROG CONTAINING IT TO TRANSFER CONTROL WITHIN THE STATEMENT BODY TO THE STATEMENT IMMEDIATELY FOLLOWING THE TAG EQ TO THE ARGUMENT OF THE GO, AND TO CONTINUE STATEMENT EVALUATION FROM THAT POINT. ONLY TAGS IN THE IMMEDIATELY CONTAINING PROG ARE EXAMINED. IF THE TAG DOES NOT EXIST IN THE CONTAINING PROG, OR IF THERE IS NO CONTAINING PROG, AN ERROR OCCURS. IF THE ARGUMENT OF GO IS NON-ATOMIC, IT IS EVALUATED REPEATEDLY UNTIL AN ATOMIC VALUE IS OBTAINED.

NOTE: BOTH GO AND RETURN MAY BE EVALUATED AT ANY POINT WITHIN A PROG, BUT IT IS SAFEST AND MOST USEFUL TO DO SO ONLY AT THE TOP LEVEL OF A PROG, OR IN COMPOSITIONS OF "COND", "AND", "OR", AND "NOT" AT THE TOP LEVEL OF A PROG.

EXAMPLE: THE FUNCTION "LENGTH" MAY BE DEFINED AS FOLLOWS:

```
(SETQQ LENGTH (LAMBDA (X)
     (PROG (N)
          (SETQ N 0)
     Q    (COND ((NULL X) (RETURN N)))
          (SETQ N (ADD1 N))
          (SETQ X (CDR X))
          (GO Q) ) ))
```

PROG2 (LAMBDA (X Y . Z) Y)

PROG2 EVALUATES ALL OF ITS ARGUMENTS AND RETURNS THE VALUE OF THE SECOND.

EXAMPLE: (PROG2 (PGSKP 2) (PRINC 2 (PLUS 2 3)) (PGSKP 2))

```
PUNCH:    /*/*/
PUNCH:    5
PUNCH:    /*/*/
RESULT:   5
```

PROGN   (LAMBDA X (CAR (LAST X)))

PROGN EVALUATES ALL OF ITS ARGUMENTS AND RETURNS THE VALUE OF THE LAST ARGUMENT. NOTE THAT (PROGN) = NIL. NOTE ALSO THAT LAMBDA EXPRESSIONS ARE SIMILAR TO PROGN IN THAT THEY TOO EVALUATE ANY NUMBER OF EXPRESSIONS AND RETURN THE VALUE OF THE LAST ONE.

****

***** CHAPTER 12 ***** INPUT/OUTPUT *****

****

1130 LISP INPUT/OUTPUT IS PERFORMED THROUGH A NUMBER OF FUNCTIONS WHICH ARE DESCRIBED BELOW. THE PARTICULAR I/O DEVICE TO BE USED IS DETERMINED BY THE FIRST ARGUMENT TO THE FUNCTION, WHICH MUST BE A DEVICE NUMBER. (IF THE FIRST ARGUMENT IS NOT A VALID DEVICE NUMBER, AN ERROR OCCURS.) THESE DEVICE NUMBERS ARE IN GENERAL THE SAME AS 1130 FORTRAN DEVICE NUMBERS:

DEVICE NUMBER	INPUT DEVICE	OUTPUT DEVICE
1		TYPEWRITER
2	1442 CARD READ/PUNCH	1442 CARD PUNCH
3		1132 PRINTER
4	1134 PAPER TAPE READER	1055 PAPER TAPE PUNCH
5		1403 PRINTER
6	KEYBOARD	
7		1627 PLOTTER
8	2501 CARD READER	

IN GENERAL, INPUT AND OUTPUT OPERATIONS ARE COMPLETELY INDEPENDENT OF EACH OTHER, WITH TWO EXCEPTIONS:

(1) THE KEYBOARD NORMALLY USES THE TYPEWRITER FOR ECHOING INPUT.

(2) OUTPUTTING TO THE 1442 CARD READ/PUNCH CAUSES ANY BUFFERED 1442 INPUT CHARACTERS TO BE LOST, AND REQUESTS A FLUSH ON THE NEXT REQUEST FOR INPUT FROM THAT DEVICE.

AN INPUT DEVICE MAY BE REGARDED AS A CHARACTER STREAM SOURCE, AND EACH OUTPUT DEVICE AS A CHARACTER STREAM SINK. IN GENERAL, ANY CHARACTER ON THE IBM 029 KEYPUNCH MAY BE INPUT OR OUTPUT, WITH THE EXCEPTION OF THE 0-8-2 PUNCH, AND WITH THE ADDITION OF "CARRIAGE RETURN" ON OUTPUT. (CARRIAGE RETURNS ARE READ AS SPACES ON KEYBOARD INPUT; THEY CANNOT BE READ FROM CARDS AT ALL; FROM PAPER TAPE THEY ARE SIMPLY IGNORED.) ANY OTHER CHARACTERS WILL BE READ AS SPACES. ALL INPUT/OUTPUT STREAMS ARE BUFFERED TO SOME EXTENT; I.E. CHARACTERS ARE NOT TRANSFERRED TO OR FROM DEVICES SINGLY, BUT IN BATCHES OF 15 TO 120.

THE CHARACTER STREAMS MAY BE DELIMITED IN TWO WAYS:

(1) CARRIAGE RETURNS, WHICH SEPARATE LINES OF INPUT OR OUTPUT.

(2) EOF (END-OF-FILE) RECORDS. ON INPUT, THESE SEPARATE GROUPS OF ONE OR MORE LINES FOR THE BENEFIT OF THE "READ" FUNCTION (SEE BELOW). ON OUTPUT THEY CREATE SEPARATOR RECORDS, THE NATURE OF WHICH MAY VARY DEPENDING ON THE PARTICULAR DEVICE (SEE "PGSKP", BELOW).

CHARACTERISTICS PECULIAR TO EACH I/O DEVICE ARE CONSIDERED BELOW.

****

***** TYPEWRITER *****

****

THE TYPEWRITER IS THE SIMPLEST OF THE I/O DEVICES. TYPEWRITER OUTPUT IS PLACED IN A SIXTEEN-CHARACTER BUFFER WHICH IS PROCESSED BY AN INTERRUPT HANDLER IN SUCH A WAY THAT EACH CHARACTER IS PRINTED AS SOON AS POSSIBLE; I.E. ONE DOES NOT HAVE TO OUTPUT A CARRIAGE RETURN SO THAT A LINE WILL BE PRINTED. THIS BUFFERING MAY CONFUSE THE INEXPERIENCED 1130 LISP USER; WHEN KEYBOARD INPUT IS REQUESTED DIRECTLY AFTER TYPEWRITER OUTPUT, THE "KB SELECT" LIGHT WILL FLASH ON SIXTEEN CHARACTERS BEFORE THE END OF THE OUTPUT. OTHER WEIRD THINGS MAY ALSO HAPPEN. HOWEVER, THIS BUFFERING ALSO HAS THE ADVANTAGE THAT A GREAT DEAL OF COMPUTATION CAN BE DONE WHILE THOSE LAST FEW CHARACTERS ARE BEING PRINTED.

****
***** KEYBOARD *****
****

THE KEYBOARD IS THE SIMPLEST OF THE INPUT DEVICES. KEYBOARD INPUT IS NORMALLY BUFFERED 100 CHARACTERS AT A TIME (HOWEVER, SEE "DDTIN"). WHEN THE BUFFER IS EXHAUSTED AND ANOTHER CHARACTER HAS BEEN REQUESTED, THE KEYBOARD HANDLER WILL ACCEPT UP TO 100 CHARACTERS OF INPUT BEFORE HONORING THE REQUEST. CHARACTERS ARE ACCEPTED UNTIL ONE OF THREE THINGS HAPPENS:

(1) THE EOF KEY IS PRESSED. THIS DOES NOT PRODUCE AN EOF RECORD (AN UNFORTUNATE COINCIDENCE OF NAMES). IT ECHOES A CARRIAGE RETURN ON THE TYPEWRITER, ENTERS A SPACE IN THE INPUT BUFFER, AND TERMINATES INPUT. THE PROGRAM REQUEST FOR A CHARACTER IS THEN HONORED.

(2) THE 0-8-2 KEY IS PRESSED (THIS CODE IS SIGNIFICANT ONLY FROM THE KEYBOARD.) THIS ECHOES THE CHARACTERS #?#<CR> (WHERE <CR> DENOTES A CARRIAGE RETURN), ERASES ALL CHARACTERS FROM THE KEYBOARD INPUT BUFFER, AND RETURNS TO THE REQUESTING PROGRAM A SPECIAL CHARACTER CALLED "KEYBOARD KILL". ALL FUNCTIONS EXCEPT READ WILL MERELY MAKE ANOTHER REQUEST FOR A CHARACTER; BUT THE READ FUNCTION (SEE BELOW) FORGETS THE ENTIRE S-EXPRESSION IT WAS READING AND STARTS AGAIN.

(3) 100 CHARACTERS HAVE BEEN INPUT. A CARRIAGE RETURN IS ECHOED, BUT NO SPACE IS PUT INTO THE BUFFER. INPUT IS TERMINATED.

IF MISTAKES ARE MADE, THE BACKSPACE AND ERASE FIELD KEYS MAY BE USED IN THE USUAL WAY. IF BACKSPACE IS PRESSED N TIMES IN A ROW, THE LAST N CHARACTERS IN THE KEYBOARD INPUT BUFFER ARE ECHOED BACK IN REVERSE ORDER BETWEEN TWO #'S. THUS IF THE FOLLWOING IS TYPED (LET <BS> DENOTE BACKSPACE):

```
     ABCDEF<BS><BS><BS>XYZ
```

THE THE TYPEWRITER ECHOES:

```
     ABCDEF#FED#XYZ
```

AND THE KEYBOARD INPUT BUFFER THEN CONTAINS:

```
     ABCXYZ
```

THE ERASE FIELD KEY CLEARS THE ENTIRE KEYBOARD INPUT BUFFER, AND ECHOES ##<CR>. THUS IF ONE MAKES A MISTAKE IN TYPING AN S-EXPRESSION TO THE FUNCTION "READ", ONE MAY ERASE SINGLE CHARACTERS, SINGLE LINES, OR ENTIRE MULTI-LINE S-EXPRESSIONS.

IF EITHER THE BACKSPACE OR ERASE FIELD KEY IS PRESSED WHEN THE BUFFER IS EMPTY, IT IS SIMPLY IGNORED.

NOTE: BECAUSE OF THE ESSENTIALLY ASYNCHRONOUS NATURE OF THE KEYBOARD AND TYPEWRITER HANDLERS, IT IS POSSIBLE FOR SPURIOUS CARRIAGE RETURNS TO BE ECHOED. THIS GENERALLY HAPPENS ONLY WHEN MANY BACKSPACES ARE TYPED ON A SINGLE LINE, AND IN ANY CASE DOESN'T AFFECT THE INPUT.

WHILE YOU'RE TYPING ON THE KEYBOARD, TAKE A LOOK AT WHAT THE ACCUMULATOR AND EXTENSION LIGHTS ON THE 1130 CONSOLE ARE DOING (THIS IS ONE WAY TO TELL THAT KEYBOARD INPUT IS WANTED).

****
***** 1442 CARD READER *****
****

THE 1442 INPUT HANDLER READS ONLY THE FIRST 72 COLUMNS OF EACH CARD. INPUT IS BUFFERED A CARD (72 CHARACTERS) AT A TIME. INPUT BUFFERING IS SIMILAR TO THAT FOR THE KEYBOARD, BUT LESS COMPLICATED: THERE ARE NO 0-8-2, ERASE FIELD, BACKSPACE, OR EOF KEY OPERATIONS. THE SEPARATIONS BETWEEN CARDS ARE NEVER SEEN (WITH ONE EXCEPTION; SEE BELOW); THE 1442 INPUT LOOKS LIKE A SOLID STREAM OF CHARACTERS. COLUMN 72 OF ONE CARD IS FOLLOWED BY COLUMN 1 OF THE NEXT.

THE ONE EXCEPTION TO CHARACTER STREAM SMOOTHNESS IS THE EOF RECORD, WHICH IS A CARD WITH THE CHARACTERS /*/*/ IN COLUMNS 1-5. (LISP USERS SHOULD

AVOID USING THIS CHARACTER COMBINATION IN PROGRAMS.) THIS RECORD IS USED TO SEPARATE S-EXPRESSIONS FOR THE READ FUNCTION (Q.V.) IF FOR SOME REASON AN INPUT FLUSH IS REQUESTED, THE 1442 HANDLER WILL READ CARDS UNTIL AN EOF RECORD IS FOUND, AND THEN READ THE FOLLOWING CARD INTO THE BUFFER, WHEN THE NEXT INPUT REQUEST IS MADE. NOTE THAT EXCEPT FOR THE OPERATION OF THE READ FUNCTION, EOF RECORDS ARE SIMPLY IGNORED; AND IN ANY CASE COLUMNS 6-80 OF AN EOF RECORD ARE IGNORED. (THESE COLUMNS MAY BE USED FOR IDENTIFYING INFORMATION, ETC.)

****

***** 1442 CARD PUNCH *****

****

THE 1442 OUTPUT HANDLER ALSO HAS A 72 CHARACTER BUFFER. CHARACTERS ARE ACCUMULATED UNTIL A CARRIAGE RETURN IS OUTPUT, AT WHICH TIME AN ENTIRE CARD IS PUNCHED AT ONCE.

IF THE 1442 IS A READ/PUNCH AND THE 1442 INPUT HANDLER IS PART OF THE 1130 LISP SYSTEM, THE FOLLOWING EXTRA OPERATIONS ARE PERFORMED:

(1) 1130 LISP WILL NEVER PUNCH ON A BLANK CARD. THE CARD FROM THE READ STATION IS READ, AND THEN PUNCHED ON IF IT IS BLANK IN COLUMNS 1-72. IF IT ISN'T, THE 1442 OUTPUT HANDLER WILL WAIT AT $PRET WITH /100B IN THE ACCUMULATOR (THE "B" IS FOR "BLANK"). THE OPERATOR SHOULD THEN PUT BLANK CARDS IN THE 1442 AND PRESS "PROGRAM START".

(2) OUTPUT TO THE 1442 CAUSES THE 1442 INPUT BUFFER TO BE CLEARED, AND AN INPUT FLUSH TO BE REQUESTED; THE NEXT INPUT OPERATION FROM THE 1442 WILL CAUSE THE 1442 INPUT HANDLER TO SEARCH FOR AN EOF RECORD.

(3) ALL PUNCHED OUTPUT WILL BE SELECTED TO STACKER 2. (INPUT CARDS ARE SELECTED TO STACKER 1.)

****

***** 1132 PRINTER *****

****

THE 1132 PRINTER HANDLER HAS A 120 CHARACTER BUFFER. CHARACTERS ARE ACCUMULATED IN THE BUFFER UNTIL A CARRIAGE RETURN CHARACTER IS OUTPUT, AT WHICH TIME AN ENTIRE LINE IS PRINTED AT ONCE. THE USER SHOULD BE CAREFUL OF THIS; SOMETIMES A LINE WILL SEEM TO BE "LOST", WHEN IN REALITY IT IS STILL IN THE BUFFER, WAITING FOR A CARRIAGE RETURN TO BE OUTPUT.

CHANNEL 9 AND 12 PUNCHES ARE HANDLED NORMALLY: THEY CAUSE AN AUTOMATIC SKIP TO CHANNEL 1 (I.E. AT THE END OF EACH PAGE AN AUTOMATIC SKIP TO A NEW PAGE IS PERFORMED).

CHARACTERS NOT IN THE 1132 CHARACTER SET (A TO Z, 0 TO 9, AND THE CHARACTERS ()+-/*,.'=&$) PRINT AS SPACES.

****

***** 1134 PAPER TAPE READER *****

****

(NOT YET IMPLEMENTED.)

****

***** 1055 PAPER TAPE PUNCH *****

****

(NOT YET IMPLEMENTED.)

****
***** 1403 PRINTER *****
****

(NOT YET IMPLEMENTED.)

****
***** 2501 CARD READER *****
****

(NOT YET IMPLEMENTED.)

****
***** 1627 PLOTTER *****
****

(NOT YET IMPLEMENTED.)

THERE ARE CERTAIN OPERATIONS COMMON TO ALL OUTPUT DEVICES: A COMMON HANDLER KEEPS TRACK OF THE NUMBER OF CHARACTERS LEFT ON EACH LINE FOR EACH DEVICE (SEE "CHRCT" AND "LINEL" BELOW). IF NO CARRIAGE RETURN HAS BEEN OUTPUT BY THE TIME THE END OF A LINE HAS BEEN REACHED, A CARRIAGE RETURN IS AUTOMATICALLY INSERTED INTO THE OUTPUT. (THIS PREVENTS OVERFLOW OF OUTPUT BUFFERS.)

THERE ARE CERTAIN OPERATIONS COMMON TO ALL INPUT DEVICES ALSO. A COMMON HANDLER MAINTAINS A ONE-CHARACTER BUFFER FOR EACH INPUT DEVICE, KNOWN AS THE "PEEK CHARACTER". THIS IS DONE SO THAT FUNCTIONS MAY PEEK AT A CHARACTER WITHOUT EFFECTIVELY REMOVING IT FROM THE INPUT STREAM; I.E. THE NEXT FUNCTION WHICH GETS A CHARACTER FROM THE SAME DEVICE WILL GET THE SAME CHARACTER AND NOT ITS SUCCESSOR. (SEE "PEEKC" AND "PEEKCH" BELOW.)

THE USER SHOULD KEEP IN MIND THAT, WITH THE TWO EXCEPTIONS NOTED ABOVE, ALL INPUT/OUTPUT STREAMS ARE COMPLETELY ASYNCHRONOUS AND INDEPENDENT. THUS IF ONE READS HALF A CARD AND THEN ASKS FOR KEYBOARD INPUT, THE OTHER HALF OF THE CARD IS STILL IN A BUFFER, WAITING TO BE READ.

****
***** INPUT FUNCTIONS *****
****

READ (LAMBDA (N) ---

READ OBTAINS JUST ENOUGH CHARACTERS FROM DEVICE N TO FORM A COMPLETE S-EXPRESSION; THIS S-EXPRESSION IS BUILT INTERNALLY AND RETURNED. READ USES THE FUNCTION INTERN ON IDENTIFIERS SO THAT THEY WILL BE PLACED ON THE OBLIST. THIS FUNCTION IS THE ONLY INPUT FUNCTION THE USER WILL USUALLY NEED; THE OTHER INPUT FUNCTIONS ARE SPECIALIZED AND SELDOM USED.
IF N=6 (KEYBOARD) AND THE 0-8-2 KEY IS PRESSED, READ FORGETS THE ENTIRE S-EXPRESSION IT WAS READING AND STARTS OVER AGAIN FROM SCRATCH.
IF AN EOF RECORD IS READ WHILE READ IS READING AN S-EXPRESSION, AN ERROR OCCURS. EOF RECORDS CAN THUS BE USED TO SEPARATE S-EXPRESSIONS IN THE

INPUT STREAM; UNBALANCED PARENTHESES WILL THEN CAUSE AN ERROR, WITHOUT SWALLOWING OTHER S-EXPRESSIONS IN THE PROCESS. READ WILL IGNORE ANY EOF RECORDS OR RIGHT PARENTHESES READ BEFORE THE FIRST CHARACTER OF THE S-EXPRESSION IS FOUND.

```
PEEKC      (LAMBDA (N) -----
```

PEEKS AT THE NEXT CHARACTER AVAILABLE FROM DEVICE N AND RETURNS IT AS A STRING.

```
PEEKCH     (LAMBDA (N) (INTERN (PEEKC N)))
```

LIKE PEEKC, BUT RETURNS AN IDENTIFIER.

```
READC      (LAMBDA (N) ---
```

LIKE PEEKC, BUT ACTUALLY REMOVES THE NEXT CHARACTER FROM THE INPUT STREAM RATHER THAN JUST PEEKING AT IT.

```
READCH     (LAMBDA (N) (INTERN (READC N)))
```

LIKE READC, BUT RETURNS AN IDENTIFIER.

```
TYI        (LAMBDA (N) ---
```

READS A CHARACTER FROM DEVICE N AND RETURNS AS A NUMBER THE EBCDIC CODE VALUE OF THAT CHATACTER.

****
***** OUTPUT FUNCTIONS *****

```
PRIN1      (LAMBDA (N . X) ---
```

OUTPUTS TO DEVICE N THE S-EXPRESSIONS X. STRINGS ARE OUTPUT WITH ENCLOSING COMMAS, AND IDENTIFIERS CONTAINING DELIMITERS OR LOOKING LIKE NUMBERS WILL BE PRINTED WITH &'S AS APPROPRIATE. NO CHARACTERS ARE INSERTED BETWEEN THOSE OF CONSECUTIVE S-EXPRESSIONS. PRIN1 RETURNS THE VALUE OF THE LAST ARGUMENT. NOTE THAT (PRIN1 N) DOES NOTHING.

```
     PRINT:    ,OUTPUT:,23
     RESULT:   23
```

```
PRINC      (LAMBDA (N . X) ---
```

LIKE PRIN1, BUT DOES NOT INSERT ENCLOSING COMMAS AROUND STRINGS OR USE &'S FOR APPROPRIATE IDENTIFIERS.

```
EXAMPLE:  (PRINC 3 ,OUTPUT:, 23)
     PRINT:    OUTPUT:23
     RESULT:   23
```

PRINT (LAMBDA (N . X) ---

LIKE PRIN1, BUT INSERTS CARRIAGE RETURNS BEFORE THE FIRST S-EXPRESSION AND AFTER EACH S-EXPRESSION. THUS SUCCESSIVE INVOCATIONS OF PRINT WILL LEAVE BLANK LINES BETWEEN GROUPS OF OUTPUT. NOTE THAT (PRINT N) OUTPUTS JUST A CARRIAGE RETURN TO DEVICE N, AND RETURNS NIL. NOTE ALSO THAT 1130 LISP HAS NO FUNCTION "TERPRI", SINCE "PRINT" CAN SERVE THE SAME PURPOSE.

```
EXAMPLE:  (PRINT 3 ,OUTPUT:, 23)
     PRINT:    ,OUTPUT:,
     PRINT:    23
     RESULT:   23
```

TYO (LAMBDA N K) ---

OUTPUTS THE CHARACTER WHOSE EBCDIC CODE IS K ONTO DEVICE N. IF THE CODE IS NOT ONE OF THE PERMISSIBLE CHARACTERS A SPACE IS USED. RETURNS K.

PGSKP (LAMBDA (N) ---

PGSKP RETURNS NIL AFTER PERFORMING THE APPROPRIATE FUNCTION ON DEVICE N:

N	FUNCTION
1	OUTPUT FIVE CARRIAGE RETURNS
2	OUTPUT CARRIAGE RETURN, /*/*/, CARRIAGE RETURN (EOF RECORD)
3	OUTPUT CARRIAGE RETURN, SKIP TO CHANNEL 1 (NEW PAGE)
4	OUTPUT CARRIAGE RETURN, /*/*/, CARRIAGE RETURN (EOF RECORD)
5	OUTPUT CARRIAGE RETURN, SKIP TO CHANNEL 1 (NEW PAGE)
7	(AS YET UNDEFINED)

CHRCT (LAMBDA (N) ---

CHRCT RETURNS THE NUMBER OF CHARACTERS LEFT ON THE CURRENT LINE OF OUTPUT FOR DEVICE N. THUS, IF TYPEWRITER LINES ARE 100 CHARACTERS LONG AND 20 CHARACTERS HAVE BEEN PRINTED ON THE CURRENT LINE, (CHRCT 1) = 80.

LINEL (LAMBDA (N . K) ---

LINEL TAKES ONE OR TWO ARGUMENTS. IF ONE IS GIVEN, LINEL RETURNS THE LENGTH OF LINES FOR DEVICE N. IF TWO ARE GIVEN LINEL SETS THE LINE LENGTH FOR DEVICE N TO K, AND THEN RETURNS THE OLD LINE LENGTH. IF K<1, 72 IS ASSUMED. IF K IS GREATER THAN A CERTAIN MAXIMUM FOR DEVICE N, THAT MAXIMUM IS ASSUMED, AS FOLLOWS:

DEVICE NUMBER:	1	2	3	4	5	7
MAXIMUM LINE LENGTH:	100	72	120	32767	120	32767

****
***** INPUT/OUTPUT STRING FUNCTIONS *****
****

READSTR (LAMBDA (X) ---

READSTR USES THE CHARACTER STRING X AS INPUT FOR A PSEUDO-DEVICE 0 AND CALLS THE READ FUNCTION TO FORM AN S-EXPRESSION. EXCESS CHARACTERS ARE

IGNORED.

```
EXAMPLE:   (READSTR ,NIL,) = NIL
           (READSTR ,  ' /8000 QUUX  ,) = (QUOTE /8000)
```

PRIN1STR   (LAMBDA (X) ---

PRIN1STR CALLS PRIN1 TO PRINT THE S-EXPRESSION X ON PSEUDO-DEVICE 0 AND ACCUMULATES THE OUTPUT CHARACTERS INTO A STRING, WHICH IS RETURNED.

```
EXAMPLE:   (PRIN1STR '(A B ,C, . D)) = ,(A B ,,C,, . D),
```

PRINCSTR   (L&MBDA (X) ---

LIKE PRIN1STR, BUT USES PRINC INSTEAD OF PRIN1.

```
EXAMPLE:   (PRINCSTR '(A B ,C, . D)) = ,(A B C . D),
```

FLATSIZE   (LAMBDA (X) (STRLENGTH (PRIN1STR X)))

FLATSIZE RETURNS THE NUMBER OF CHARACTERS PRIN1 WOULD OUTPUT IF IT WERE TO PRINT THE S-EXPRESSION X.

FLATC   (LAMBDA (X) (STRLENGTH (PRINCSTR X)))

LIKE FLATSIZE, BUT USES PRINC INSTEAD OF PRIN1.

****

***** CHAPTER 13 ***** SYSTEM FUNCTIONS AND THE LISP TOPLEVEL *****

****

THE 1130 LISP SYSTEM AS A WHOLE LOOKS LIKE A LISP FUNCTION WHICH OPERATES ON INPUT AND PRODUCES OUTPUT. THIS SO-CALLED TOPLEVEL LOOKS SOMETHING LIKE THIS:

```
(PROG (TOPLEVEL ERRLIST SYSI SYSO)
      (GC)
      (COND ((ZEROP (EXAM /7C)) (TEND)) ((TYP)))
      (RANDOM 0)
      (PRINT SYSO HEADER-MESSAGE)
Q     (COND      ((NULL TOPLEVEL) (PRINT SYSO (EVAL (READ SYSI))))
                 ((EQ TOPLEVEL T)(PRINT SYSO(EVAL(PRINT SYSO(READ SYSI)))))
                 (T (EVAL TOPLEVEL)))
      (GO Q)
ERR   (PRINT SYSO ERROR-MESSAGE)
      (MAPCAR 'EVAL ERRLIST)
      (GO Q))
```

IT IS HERE ASSUMED THAT (GO ERR) IS PERFORMED WHEN AN ERROR OCCURS. THE USER SHOULD NOT TAKE THE ABOVE EXPRESSION TOO LITERALLY; BUT IT SHOULD SERVE TO INDICATE WHAT IS HAPPENING.

****

***** SYSTEM FUNCTIONS *****

****

TOPL   (LAMBDA (X) ---

TOPL EFFECTIVELY DOES A (SETQ TOPLEVEL X) FOR THE ABOVE DEFINITION; IT ALTERS THE WAY IN WHICH THE SYSTEM DOES THINGS. FOR EXAMPLE, TO GET AN "EVALQUOTE" SYSTEM (1130 LISP IS AN "EVAL" SYSTEM), ONE COULD EVALUATE (TOPL '(PRINT SYSO (APPLY (READ SYSI) (READ SYSI)))). ANOTHER USEFUL INVOCATION IS (TOPL '(EVAL (READ SYSRD))). THIS CAUSES EXPRESSIONS TO BE READ AND EVALUATED FROM THE SYSTEM INPUT DEVICE, WITHOUT PRINTING OUT ANY RESULTS; THIS IS A FAST WAY TO READ IN MANY FUNCTION DEFINITIONS. NOTE THAT TOPL RETURNS NIL AS ITS VALUE.

TYP   (NLAMBDA NIL ---

TYP IS SIMILAR TO (NLAMBDA NIL (SETQ SYSO 1) (SETQ SYSI 6) (TOPL NIL)), BUT HAS ONE MORE EFFECT: IT CHANGES CERTAIN INTERNAL VARIABLES CALLED THE DEFAULT DEVICE NUMBERS TO 6 AND 1. THESE DEFAULT NUMBERS ARE USED IF SYSI OR SYSO (Q.V.) EVER TAKE ON INVALID VALUES. TYP RETURNS NIL.

TEND   (NLAMBDA NIL ---

TEND IS SIMILAR IN EFFECT TO (NLAMBDA NIL (SETQ SYSO SYSPR) (SETQ SYSI SYSRD) (TOPL T)), WHERE SYSPR AND SYSRD HAVE THE VALUES THEY WERE INITIALIZED TO; BUT ALSO CHANGES THE DEFAULT DEVICE NUMBERS AS TYP DOES. THE VALUE RETURNED BY TEND IS NIL.

PAUSE (NLAMBDA NIL ---

PAUSE WAITS FOR ALL PENDING INPUT/OUTPUT TO FINISH, THEN WAITS WITH /AAAA /5555 IN THE ACCUMULATOR AND EXTENSION UNTIL "PROGRAM START" IS PRESSED. PAUSE THEN RETURNS NIL.

QUIT (NLAMBDA NIL ---

QUIT PRINTS A SPECIAL SIGN-OFF MESSAGE, THEN "CALL EXITS", I.E. RETURNS TO THE 1130 DISK MONITOR SYSTEM. NOTE THAT ANY DATA AND/OR FUNCTIONS NOT SAVED ON SOME EXTERNAL MEDIUM (CARDS, PAPER TAPE, ETC.) WILL BE LOST.

****

***** CHAPTER 14 ***** MISCELLANEOUS FUNCTIONS *****

****

GC (NLAMBDA NIL ---

CAUSES A "GARBAGE COLLECTION" TO BE PERFORMED. THE GARBAGE COLLECTOR IS A SPECIAL ROUTINE INVOKED AUTOMATICALLY BY CERTAIN FUNCTIONS WHEN FREE STORAGE IS EXHAUSTED. IT HAS THE EFFECT OF EXAMINING ALL LIST STRUCTURE IN CORE TO SEE WHAT IS NOT NEEDED ANY MORE; THE CORE OCCUPIED BY SUCH STRUCTURES IS COLLECTED ONTO A "FREELIST" FOR FURTHER USE. UNLESS GAGGED (SEE "GCGAG"), THE GARBAGE COLLECTOR PRINTS OUT A REPORT ON THE AMOUNT OF STORAGE RECLAIMED. THE GARBAGE COLLECTOR FOR 1130 LISP AUTOMATICALLY USES THE "TWA" ("TRULY WORTHLESS ATOM") COLLECTION FEATURE; ANY IDENTIFIER WHICH HAS NO VALUE AND WHICH IS NOT PART OF ANY LIST STRUCTURE EXCEPT THE OBLIST IS REMOVED FROM THE OBLIST AND THE STORAGE OCCUPIED BY IT IS RECLAIMED. WHEN FREE STORAGE IS VERY FULL OF PROGRAMS AND OTHER DATA, THE GARBAGE COLLECTOR MAY BE INVOKED VERY FREQUENTLY, PERHAPS ONCE EVERY THREE SECONDS; SINCE IN SUCH A SITUATION THE GARBAGE COLLECTION PROCESS TAKES ABOUT ONE SECOND ON 16K OF CORE, PROGRAM EXECUTION WILL BE SLOWED BY A FACTOR OF 50% TO 200%. GARBAGE COLLECTION CAN SOMETIMES BE RECOGNIZED WHEN IT OCCURS BY A PECULIAR PATTERN OF LIGHTS IN THE EXTENSION: BIT 0 ON, THE NEXT SEVEN BITS ZERO, AND THE LOW ORDER EIGHT BITS FLASHING.

EXAM (LAMBDA (N) ---

RETURNS AS A NUMBER THE CONTENTS OF CORE LOCATION N. N MUST BE A NUMBER.

DEP (LAMBDA (N K) ---

PUTS THE NUMBER K IN CORE LOCATION N AND RETURNS K. K AND N MUST BE NUMBERS.

ERRSET (NLAMBDA (E . X) ---

ERRSET EVALUATES E AND IF NO ERROR OCCURS RETURNS (LIST (EVAL E)). IF AN ERROR OCCURS, THEN IF X=NIL OR IS NOT PRESENT, AN ERROR MESSAGE IS PRINTED; OTHERWISE THE ERROR MESSAGE IS SUPPRESSED. NOTE, HOWEVER, THAT MESSAGES FOR FATAL AND SYSTEM ERRORS ARE NEVER SUPPRESSED. THEN (ASSUMING THE ERROR WAS NOT A FATAL OR SYSTEM ERROR) ERRSET RETURNS THE NUMBER OF THE ERROR. IF THE FUNCTION "ERR" IS CALLED DURING EVALUATION OF E, NO MESSAGE IS PRINTED AND THE ARGUMENT OF ERR IS RETURNED AS THE VALUE OF THE ERRSET. NOTE THAT X, IF PRESENT, IS NOT EVALUATED.

ERR (LAMBDA (X) ---

RETURNS THE VALUE OF X TO THE MOST RECENT ERRSET, OR TO THE TOP LEVEL OF THE LISP SYSTEM IF THERE IS NO ENCLOSING ERRSET; IN THE LATTER CASE A SPECIAL MESSAGE IS PRINTED TO THE EFFECT THAT THE RESULT WAS OBTAINED FROM THE ERR FUNCTION.

```
EXAMPLES:       (ERRSET (MINUS 5)) = (-5)
                (ERRSET (MINUS 5) T) = (-5)
                (ERRSET (MINUS NIL)) = 38, AND ERROR MESSAGE 38 IS PRINTED
```

```
(ERRSET (MINUS NIL) T) = 38, AND NO MESSAGE IS PRINTED
(ERRSET (MINUS (ERR 43))) = 43, AND NO MESSAGE IS PRINTED
(MINUS (ERR 43)) = 43, AND PRINTS A MESSAGE SAYING THAT
        THE 43 WAS THE ARGUMENT OF THE ERR FUNCTION
```

```
*****************************************************
***** CHAPTER 15 ***** SYSTEM-SUPPLIED IDENTIFIERS *****
*****************************************************
```

A NUMBER OF IDENTIFIERS ARE ON THE OBLIST WHEN THE LISP SYSTEM IS FIRST EXECUTED WHICH DO NOT HAVE FUNCTIONS AS VALUES. THEY ARE INCLUDED FOR SPECIAL PURPOSES OUTLINED BELOW.

C-R SPECIAL FLAG USED BY INTERN, EVAL, AND APPLY (Q.V.) FOR RECOGNITION AND EVALUATION OF COMPOSITE CAR/CDR FUNCTIONS. VALUE IS UNDEFINED.

CR AN IDENTIFIER WHOSE VALUE IS ANOTHER IDENTIFIER, WHOSE VALUE IS ITSELF AND WHOSE PRINT NAME IS A CARRIAGE RETURN. THIS IS USEFUL FOR SPECIAL FORMATTING WITH PRIN1 AND PRINC. NOTE THAT PRIN1 DOES NOT PRINT A & BEFORE A CARRIAGE RETURN IN A PRINT NAME. THUS

```
(PRINC 1 CR ,OUTPUT 1: , 23 CR ,OUTPUT 2: , 42 CR)
```

WOULD CAUSE THE FOLLOWING TO BE PRINTED:

```
OUTPUT 1: 23
OUTPUT 2: 42
```

DDTIN A SWITCH WHICH, IF NON-NIL, SUPPRESSES BUFFERING OF KEYBOARD INPUT. THE BACKSPACE AND ERASE FIELD KEYS BECOME INEFFECTIVE, AND INPUT ROUTINES CAN SEE CHARACTERS AS THEY ARE TYPED. INITIALLY HAS THE VALUE NIL.

ERRLIST A LIST OF FORMS TO BE EVALUATED WHEN AN ERROR IS SEEN AT THE TOP LEVEL OF THE SYSTEM (I.E. NOT BY AN ERRSET). THUS

```
(SETQQ ERRLIST (PRINT 1 'FOOL!)))
```

WOULD CAUSE "FOOL!" TO BE PRINTED ON THE TYPEWRITER WHENEVER AN ERROR OCCURRED OUTSIDE AN ERRSET. ERRLIST INITIALLY HAS THE VALUE NIL.

GCGAG A SWITCH, INITIALLY HAVING THE VALUE T, WHICH IF NON-NIL GAGS THE GARBAGE COLLECTOR'S STATUS REPORT. IF NIL, THE GARBAGE COLLECTOR PRINTS A REPORT EVERY TIME IT IS INVOKED.

HEX A SWITCH, INITIALLY NIL, WHICH IF NON-NIL CAUSES PRIN1, PRINC, AND PRINT (Q.V.) TO OUTPUT NUMBERS IN HEXADECIMAL FORM RATHER THAN THE USUAL DECIMAL.

KBECHO A SWITCH, INITIALLY T, WHICH IF NIL SUPPRESSES THE ECHOING OF KEYBOARD INPUT ON THE TYPEWRITER.

LAMBDA FLAG USED BY EVAL AND APPLY TO DENOTE A FUNCTION WHOSE ARGUMENTS ARE TO BE EVALUATED. VALUE IS UNDEFINED.

MLAMBDA FLAG USED BY EVAL AND APPLY TO DENOTE A FUNCTION WHOSE ARGUMENTS ARE NOT TO BE EVALUATED AND WHOSE RESULT IS TO BE RE-EVALUATED. VALUE IS UNDEFINED.

NIL A SPECIAL IDENTIFIER DENOTING THE EMPTY LIST (). ITS VALUE IS ITSELF. IT IS ALSO USED IN MANY SITUATIONS TO DENOTE THE IDEA OF "FALSE". NOTE THAT UNLIKE OTHER IDENTIFIERS, NIL IN GENERAL IS NOT PERMITTED TO HAVE ITS VALUE ALTERED, E.G. BY SETQ OR FUNCTION ARGUMENT BINDING.

NLAMBDA FLAG USED BY EVAL AND APPLY TO DENOTE A FUNCTION WHOSE ARGUMENTS ARE NOT TO BE EVALUATED. VALUE IS UNDEFINED.

OBLIST A LIST OF ALL IDENTIFIERS SEEN BY THE INTERN FUNCTION. IDENTIFIERS SHOULD BE REMOVED ONLY BY THE REMOB FUNCTION OR THE GARBAGE COLLECTOR. IDENTIFIERS ARE ARRANGED IN ALPHABETICAL ORDER. WARNING: TAMPERING WITH THE OBLIST CAN EASILY CONFUSE, HANG UP, OR DESTROY THE SYSTEM.

SUBR FLAG USED BY EVAL AND APPLY TO DENOTE A FUNCTION CODED INTO THE LISP SYSTEM IN MACHINE LANGUAGE. VALUE IS UNDEFINED.

SYSI A VARIABLE WHOSE VALUE IS USED BY THE SYSTEM AS AN INPUT DEVICE NUMBER. IF ITS VALUE IS NOT A VALID DEVICE NUMBER, A SPECIAL DEFAULT VALUE IS USED. THIS VARIABLE IS INITIALIZED WHEN THE LISP SYSTEM IS FIRST EXECUTED.

SYSO LIKE SYSI, BUT FOR OUTPUT.

SYSPCH THE NUMBER OF THE FASTEST PUNCHING DEVICE FOR WHICH AN OUTPUT HANDLER IS AVAILABLE: THE 1442 IF AVAILABLE, ELSE THE 1Ø55. IF NO PUNCH DEVICE IS AVAILABLE, THE VALUE OF SYSPCH WILL BE Ø.

SYSPR THE NUMBER OF THE FASTEST PRINTER FOR WHICH AN OUTPUT HANDLER IS AVAILABLE: THE 1403 IF AVAILABLE, ELSE THE 1132, ELSE THE TYPEWRITER.

SYSRD THE NUMBER OF THE FASTEST READER FOR WHICH AN INPUT HANDLER IS AVAILABLE: THE 25Ø1 IF AVAILABLE, ELSE THE 1442, ELSE THE 1134, ELSE THE KEYBOARD.

T AN IDENTIFI5R WHOSE VALUE IS ITSELF. USED TO DENOTE THE IDEA OF "TRUE", AS OPPOSED TO NIL; ESPECIALLY AS THE VALUE OF A PREDICATE.

****

***** APPENDIX A ***** 1130 LISP ERROR MESSAGES *****

****

LISP 1130 KEEPS A SET OF ERROR MESSAGES IN A DATA FILE CALLED "LERRS". WHENEVER AN ERROR CONDITION OCCURS, THE APPROPRIATE MESSAGE IS READ IN AND PRINTED (UNLESS SUPPRESSED BY AN ERRSET). MOST OF THESE MESSAGES ARE SELF-EXPLANATORY; HOWEVER, THEY ARE ALL LISTED BELOW, WITH EXPLANATIONS AND POSSIBLE CAUSES.

ERRORS COME IN FIVE FLAVORS OF VARYING SERIOUSNESS:

(0) INFORMATIONAL. NOT ACTUALLY AN ERROR, JUST A MESSAGE OF SOME KIND.

(1) MINOR. AN ERROR HAS OCCURRED, BUT SOME DEFAULT ACTION IS TAKEN.

(2) MAJOR. SERIOUS ENOUGH TO HALT FUNCTION EXECUTION, BUT RECOVERABLE. CAUSES A RETURN TO THE SYSTEM TOPLEVEL, OR THE IMMEDIATELY CONTAINING ERRSET.

(3) FATAL. AN ERROR CONDITION HAS OCCURRED FROM WHICH RECOVERY IS NOT POSSIBLE. LISP RETURNS TO THE 1130 MONITOR SYSTEM.

(4) SYSTEM. AN ERROR HAS OCCURRED WHICH PROBABLY INDICATES A MALFUNCTION ON THE PART OF THE LISP SYSTEM, OR ELSE SERIOUS TAMPERING WITH THE SYSTEM BY THE USER. SYSTEM ERRORS CAUSE A DUMP OF CORE TO BE MADE, TO AID IN LOCATING POSSIBLE BUGS IN THE LISP SYSTEM. HOPEFULLY THE USER WILL NEVER SEE A SYSTEM ERROR OCCUR.

ERRORS ARE PRINTED AS A CARRIAGE RETURN, FIVE *'S, A SPACE, THE MESSAGE, AND A CARRIAGE RETURN. THE MESSAGES ARE PRINTED IN THE FORM GIVEN BELOW, POSSIBLY WITH VARIABLE INFORMATION INSERTED. IN THE MESSAGE PROTOTYPES BELOW, CERTAIN SPECIAL CHARACTERS INDICATE INSERTED INFORMATION:

@	AN S-EXPRESSION
#	A HEXADECIMAL NUMBER
%	A DECIMAL NUMBER
\	A CARRIAGE RETURN

A SEMICOLON (;) INDICATES THE END OF THE MESSAGE, AND DOES NOT PRINT.

EACH MESSAGE DESCRIPTION BELOW IS GIVEN IN THE FOLLOWING FORM:

MESSAGE PROTOTYPE WITH INDICATIONS OF VARIABLE INFORMATION INSERTION;

XX QQQQQ. DESCRIPTION OF ERROR AND POSSIBLE CAUSES.

WHERE XX IS THE NUMBER OF THE MESSAGE (THIS IS THE NUMBER RETURNED BY ERRSET AS ITS VALUE WHEN AN ERROR OCCURS), AND WHERE QQQQQ IS THE SERIOUSNESS OF THE ERROR, AS INDICATED ABOVE (INFORMATIONAL, MINOR, MAJOR, FATAL, SYSTEM).

NOTE THAT IN MANY MESSAGES AN INDICATION OF THE ROUTINE WHICH DISCOVERED THE ERROR IS GIVEN IN THE MESSAGE, SET APART BY AN ASTERISK. THUS, ERROR 05 IS DISCOVERED BY THE GARBAGE COLLECTOR.

1130 LISP 1.6 ***** THTART OF RUN ***** LITHP ITH LITHTENING;

00 INFORMATIONAL. THIS MESSAGE IS PRINTED OUT WHEN THE LISP SYSTEM IS FIRST ENTERED FROM THE 1130 MONITOR.

ERROR 01 OVERFLOW ON REGULAR PUSHDOWN LIST;

01 MAJOR. THE PUSHDOWN LIST USED FOR RECURSIVE PROCESSES HAS OVERFLOWED. USUALLY CAUSED BY EXCESSIVE FUNCTION RECURSION. INFINITE RECURSION WILL EVENTUALLY CAUSE THIS ERROR.

ERROR 02 UNDERFLOW REG PDL. SYSTEM ERROR. A CORE DUMP FOLLOWS.;

02 SYSTEM. THE SYSTEM HAS TRIED TO RETRIEVE MORE ITEMS FROM THE PUSHDOWN LIST THAN WERE PLACED ON IT. THIS GENERALLY INDICATES AN ERROR IN THE LISP SYSTEM ITSELF.

ERROR 03 OVERFLOW ON SPECIAL PUSHDOWN LIST;

03 MAJOR. THE SPECIAL PUSHDOWN LIST USED FOR SAVING AND RESTORING VARIABLE BINDINGS HAS OVERFLOWED. SIMILAR TO ERROR 01.

ERROR 04 UNDERFLOW SPEC PDL. SYSTEM ERROR. A CORE DUMP FOLLOWS.;

04 SYSTEM. THE SYSTEM HAS TRIED TO RETRIEVE MORE ITEMS FROM THE SPECIAL PUSHDOWN LIST THAN WERE PLACED ON IT. SIMILAR TO ERROR 02.

ERROR 05 FREE STORAGE SPACE FULL - FATAL * GARBAGE COLLECTOR;

05 FATAL. FREE STORAGE (THE PART OF THE COMPUTER'S MEMORY USED FOR STORAGE OF S-EXPRESSIONS) HAS BECOME COMPLETELY FILLED, AND THE GARBAGE COLLECTOR COULD NOT RECLAIM EVEN TWO WORDS (THE AMOUNT OF SPACE OCCUPIED BY A SINGLE DOTTED PAIR).

ERROR 06 FIXED-POINT NUMBER SPACE FULL - FATAL * GARBAGE COLLECTOR;

06 FATAL. THE FIXED-POINT NUMBER SPACE (THE PART OF THE COMPUTER'S MEMORY USED FOR STORAGE OF NUMBERS) HAS BECOME COMPLETELY FILLED, AND THE GARBAGE COLLECTOR COULD NOT RECLAIM EVEN ONE WORD (THE AMOUNT OF SPACE OCCUPIED BY ONE NUMBER).

WORDS AVAILABLE * FREE STORAGE % * FIXED-POINT %;

07 INFORMATIONAL. STATUS REPORT FROM THE GARBAGE COLLECTOR. AFTER A GARBAGE COLLECTION THIS MESSAGE IS PRINTED UNLESS THE IDENTIFIER GCGAG HAS A NON-NIL VALUE. EACH NUMBER IS THE NUMBER OF WORDS RECLAIMED FOR EACH PART OF STORAGE. NOTE THAT NUMBERS OCCUPY ONE WORD, BUT DOTTED PAIRS IN FREE STORAGE OCCUPY TWO WORDS.

FREE STORAGE SPACE EXHAUSTED * GARBAGE COLLECTOR;

08 INFORMATIONAL. THIS MESSAGE IS PRINTED IF FREE STORAGE BECOMES FULL, MAKING NECESSARY A GARBAGE COLLECTION. PRINTED ONLY IF GCGAG IS NIL. SHOULD BE IMMEDIATELY FOLLOWED BY MESSAGE 07, UNLESS ERROR 05 OCCURS.

FIXED-POINT NUMBER SPACE EXHAUSTED * GARBAGE COLLECTOR;

09 INFORMATIONAL. PRINTED IF THE FIXED-POINT NUMBER SPACE BECOMES FULL, MAKING NECESSARY A GARBAGE COLLECTION. PRINTED ONLY IF GCGAG IS NIL. SHOULD BE IMMEDIATELY FOLLOWED BY MESSAGE 07, UNLESS ERROR 06 OCCURS.

ERROR 10 OUTPUT DEVICE NUMBER % IS NOT AVAILABLE. THIS DEVICE WILL BE USED...;

10 MINOR. AN ATTEMPT WAS MADE TO USE AN OUTPUT DEVICE FOR WHICH THE LISP SYSTEM HAS NO DEVICE HANDLER. THE OPERATION WILL INSTEAD USE THE DEVICE ON WHICH THE ERROR MESSAGE WAS PRINTED.

ERROR 11 PUSHDOWN LIST OVERFLOW - FATAL * GARBAGE COLLECTOR;

11 FATAL. PUSHDOWN LIST OVERFLOW OCCURRED DURING GARBAGE COLLECTION. THERE IS NO EASY WAY TO RECOVER FROM THIS. YOU LOSE.

ERROR 12 ATOM ON OBLIST WITH NULL PRINT NAME * INTERN * A CORE DUMP FOLLOWS;

12 SYSTEM. THE FUNCTION INTERN FOUND AN IDENTIFIER ON THE OBLIST WITH A NULL PRINT NAME. THIS SHOULD NEVER HAPPEN DURING NORMAL LISP OPERATIONS.

ERROR 13 INPUT DEVICE NUMBER % IS NOT AVAILABLE;

13 MAJOR. AN ATTEMPT WAS MADE TO USE AN INPUT DEVICE FOR WHICH THE LISP SYSTEM HAS NO DEVICE HANDLER.

ERROR 14 IMPROPERLY STRUCTURED OBLIST - FATAL * INTERN * OBLIST IS \@;

14 FATAL. THE FUNCTION INTERN HAS FOUND THAT THE OBLIST IS NOT IN THE FORM OF A LIST OF IDENTIFIERS. THIS IS FATAL SINCE THE OBLIST IS VITAL TO THE OPERATION OF THE LISP SYSTEM. THE OBLIST IS PRINTED OUT FOR INSPECTION.

ERROR 15 ATOM WITH NULL PRINT NAME * GARBAGE COLLECTOR * A CORE DUMP FOLLOWS.;

15 SYSTEM. THE GARBAGE COLLECTOR HAS FOUND AN IDENTIFIER WITH A NULL PRINT NAME. THIS SHOULD NEVER HAPPEN DURING NORMAL LISP OPERATIONS.

ERROR 16 IMPROPERLY STRUCTURED OBLIST - FATAL * GARBAGE COLLECTOR * OBLIST IS\@;

16 FATAL. THE GARBAGE COLLECTOR HAS FOUND THAT THE OBLIST IS NOT IN THE FORM OF A LIST OF IDENTIFIERS. SIMILAR TO ERROR 14.

ERROR 17 @ IS AN INVALID ARGUMENT * INTERN;

17 MAJOR. THE FUNCTION INTERN RECEIVED AN ARGUMENT WHICH WAS NOT AN IDENTIFIER OR A STRING. THE ARGUMENT IS PRINTED IN THE MESSAGE.

ERROR 18 ARGUMENT IS ATOM WITH NULL PRINT NAME * INTERN;

18 MAJOR. INTERN OBJECTS TO RECEIVING AN ARGUMENT WITH A NULL PRINT NAME, SINCE SUCH AN ATOM CANNOT BE PUT ON THE OBLIST.

ERROR 19 NO ITEM PRECEDES DOT IN S-EXPRESSION ON DEVICE % * READ;

19 MAJOR. THE READ FUNCTION RECEIVED A DOT (.) IMMEDIATELY AFTER A ((I.E. A LEFT PARENTHESIS); THIS CAN NEVER BE PART OF A VALID S-EXPRESSION. READ REQUESTS AN INPUT FLUSH FOR THE SPECIFIED DEVICE.

ERROR 20 . OR) FIRST CHARACTER OF S-EXPRESSION ON DEVICE % * READ;

20 MAJOR. THIS ERROR CAN OCCUR FOR ONE OF TWO REASONS: EITHER THE VERY FIRST CHARACTER NOT A SPACE OR RIGHT PARENTHESIS SEEN AFTER READ WAS CALLED WAS A DOT, OR A DOT OR RIGHT PARENTHESIS IMMEDIATELY FOLLOWED A DOT. THUS .(QUUX) OR (A .) OR (B C . . Q) WOULD CAUSE THIS ERROR. NOTE THAT IF THE VERY FIRST CHARACTERS SEEN AFTER READ IS CALLED ARE RIGHT PARENTHESES AND SPACES, THEY ARE IGNORED. THUS ONE MAY PUT A LARGE NUMBER OF RIGHT PARENTHESES AFTER AN S-EXPRESSION TO MAKE SURE THE LEFT PARENTHESES ARE BALANCED WITHOUT CAUSING AN ERROR WHEN THE NEXT S-EXPRESSION IS READ FROM THE SAME DEVICE.

ERROR 21 SECOND ITEM AFTER DOT IN S-EXPRESSION @ ON DEVICE % * READ;

21 MAJOR. WHILE READING A PORTION OF AN S-EXPRESSION READ FOUND A DOTTED LIST WITH MORE THAN ONE S-EXPRESSION AFTER THE DOT. THUS (A B . C D) WOULD CAUSE THIS ERROR. THE DOTTED LIST FORMED BY IGNORING THE EXTRA ITEM(S) AFTER THE DOT IS PRINTED TO AID IN LOCATING THE ERROR; THUS THE ABOVE EXAMPLE WOULD PRINT "...IN S-EXPRESSION (A B . C) ON DEVICE...".

ERROR 22 THE DECIMAL NUMBER % READ ON DEVICE % IS TOO LARGE * READ;

LESS THAN -32768 OR GREATER THAN 32767. THE NUMBER IS ASSUMED TO HAVE A VALUE OF -32768 OR 32767 IN THOSE RESPECTIVE CASES.

ERROR 23 @ IS AN UNBOUND VARIABLE * EVAL;

23 MAJOR. EVAL TRIED TO EVALUATE THE IDENTIFIER SPECIFIED, BUT FOUND THAT ITS VALUE WAS UNDEFINED.

ERROR 24 @ IS AN INVALID FUNCTION * EVAL;

24 MAJOR. EVAL FOUND WHILE EVALUATING AN S-EXPRESSION THAT A NUMBER OR A STRING WAS BEING USED AS A FUNCTION.

ERROR 25 @ IS AN UNDEFINED FUNCTION * EVAL;

25 MAJOR. THE IDENTIFIER PRINTED WAS USED AS A FUNCTION IN AN S-EXPRESSION, BUT WHEN EVALUATING THE S-EXPRESSION, EVAL FOUND THAT THE IDENTIFIER HAD AN UNDEFINED VALUE.

ERROR 26 FIRST ARG WAS @ * LABEL * SHOULD BE BINDABLE ATOM;

26 MAJOR. IN A LABEL EXPRESSION THE FIRST ARGUMENT SHOULD HAVE BEEN A NON-NIL IDENTIFIER, WHICH COULD BE BOUND TO THE SECOND ARGUMENT (THE FUNCTION). INSTEAD THE FIRST ARGUMENT WAS THE S-EXPRESSION PRINTED.

ERROR 27 WRONG NUMBER OF ARGUMENTS * LABEL * LABEL EXPRESSION WAS\@;

27 MAJOR. A LABEL EXPRESSION CONTAINED MORE OR FEWER THAN TWO ARGUMENTS. THE FIRST SHOULD BE A NON-NIL IDENTIFIER, THE SECOND A FUNCTION. THE INVALID LABEL EXPRESSION IS PRINTED OUT FOR INSPECTION.

ERROR 28 TOO MANY ARGS FOR THE PARAMETER LIST @ * APPLY;

28 MAJOR. WHEN BINDING PARAMETERS TO ARGUMENTS, APPLY FOUND THAT TOO MANY ARGUMENTS WERE SUPPLIED TO THE FUNCTION. THE PARAMETER LIST OF THE FUNCTION IS PRINTED TO AID IN LOCATING THE ERROR.

ERROR 29 TOO FEW ARGS FOR THE PARAMETER LIST @ * APPLY;

29 MAJOR. SIMILAR TO ERROR 28, BUT TOO FEW ARGUMENTS WERE SUPPLIED.

ERROR 30 THE PARAMETER LIST @ IS BADLY STRUCTURED * APPLY;

30 MAJOR. WHILE BINDING PARAMETERS TO ARGUMENTS FOR A FUNCTION, APPLY FOUND THAT THE FUNCTION'S PARAMETER LIST WAS NOT IN THE FORM OF AN IDENTIFIER, A LIST OF IDENTIFIERS, OR A DOTTED LIST OF IDENTIFIERS, WHERE THE IDENTIFIER(S) MAY NOT BE NIL. THE INVALID PARAMETER LIST IS PRINTED FOR INSPECTION.

ERROR 31 TOO MANY ARGS FOR THE SUBR @ AT ADDRESS # * APPLY;

31 MAJOR. WHILE SPREADING ARGUMENTS FOR A SUBR, APPLY FOUND THAT TOO MANY ARGUMENTS WERE SUPPLIED. THE SUBR OBJECT IS PRINTED; IT WILL BE IN THE FORM (SUBR . /XXXX) WHERE XXXX IS THE SUBR'S HEADER WORD. THE ADDRESS OF THE HEADER WORD IS ALSO PRINTED. THIS INFORMATION CAN BE USED TO IDENTIFY THE FUNCTION.

ERROR 32 TOO FEW ARGS FOR THE SUBR @ AT ADDRESS # * APPLY;

32 MAJOR. SIMILAR TO ERROR 31, BUT TOO FEW ARGUMENTS WERE SUPPLIED.

ERROR 33 THE SUBR @ AT ADDRESS # WANTS MORE THAN 8 SEPARATE ARGS * APPLY;

33 MAJOR. SUBR'S ARE NOT ALLOWED TO RECEIVE MORE THAN EIGHT SEPARATE ARGUMENTS, OR SEVEN AND A LIST OF THE REST. UNLESS THE USER HAS DONE STRANGE THINGS TO THE SYSTEM, THIS PROBABLY INDICATES A BUG IN THE LISP SYSTEM. THIS ERROR IS RECOVERABLE, BUT SHOULD BE INVESTIGATED IMMEDIATELY. THE SUBR OBJECT AND THE ADDRESS OF THE SUBR HEADER WORD ARE PRINTED AS FOR ERROR 31.

ERROR 34 # IS AN ABSURD ADDRESS FOR A SUBR IN THE SUBR EXPRESSION @ * APPLY;

33 MAJOR. THE SUBR EXPRESSION PRINTED INDICATES THAT THE SUBR IS SOMEWHERE IN FREE STORAGE, THE FIXED-POINT NUMBER SPACE, THE PUSHDOWN LISTS, OR SOME OTHER PECULIAR LOCATION. IN ANY CASE, IT CAN'T BE A REAL SUBR. NOTE THAT ALTHOUGH SUBR OBJECTS CAN BE PRINTED, THEY CAN'T BE READ BACK IN, SINCE THE HEADER WORD WILL READ AS A NUMBER; THE RESULT WILL POINT INTO THE FIXED-POINT NUMBER SPACE AND NOT TO A SUBR. THIS IS NOT A COMMON ERROR.

ERROR 35 A C-R FUNCTION TAKES ONE ARGUMENT, BUT @ GOT @ AS ARGUMENTS * APPLY;

35 MAJOR. APPLY DETECTED A COMPOSITE CAR/CDR FUNCTION WHICH WAS GIVEN ZERO OR MORE THAN ONE ARGUEMTNS. THE FUNCTION NAME AND A LIST OF THE SUPPLIED ARGUMENTS ARE PRINTED FOR INSPECTION.

ERROR 36 FIRST ARG WAS @ * SET/SETQ/SETQQ * SHOULD BE BINDABLE ATOM;

36 MAJOR. THE SET, SETQ, AND SETQQ FUNCTIONS REQUIRE THEIR FIRST ARGUMENT TO BE A NON-NIL IDENTIFIER. THE INVALID FIRST ARGUMENT IS PRINTED.

ERROR 37 /*/*/ RECORD ENCOUNTERED INSIDE S-EXPRESSION ON DEVICE % * READ;

37 MAJOR. THE FUNCTION READ ENCOUNTERED AN EOF RECORD WHILE IN THE MIDDLE OF READING AN S-EXPRESSION. THIS ERROR DOES NOT OCCUR IF THE ONLY CHARACTERS READ BEFORE THE EOF RECORD WERE SPACES AND/OR RIGHT PARENTHESES. THUS, READ WILL SIMPLY PASS OVER AN EOF RECORD WHEN LOOKING FOR THE BEGINNING OF THE NEXT S-EXPRESSION. A USEFUL TECHNIQUE WHEN S-EXPRESSIONS ARE TO BE READ FROM CARDS IS TO SEPARATE THE S-EXPRESSIONS WITH EOF RECORDS; IF ANY OF THE EXPRESSIONS HAS TOO FEW RIGHT PARENTHESES, THE EOF RECORD WILL FORCE AN ERROR. IF NO EOF RECORD WERE PRESENT, ALL SUCCEEDING S-EXPRESSIONS WOULD BE READ AS IF THEY WERE PART OF THE ERRONEOUS ONE. SINCE WHEN KEYPUNCHING FUNCTIONS IT IS EASY TO REPUNCH AN ERRONEOUS CARD AND LEAVE THE OLD ONE (WHICH MAY HAVE TOO FEW RIGHT PARENTHESES) IN THE DECK, USING THE EOF RECORDS IS ADVISABLE SO THAT SUCH ERRORS WILL BE CAUGHT.

ERROR 38 FUNCTION @ REQUIRES A NUMERIC ARGUMENT, BUT GOT @ INSTEAD;

38 MAJOR. THE FUNCTION WHOSE NAME WAS PRINTED REQUIRES AN ARGUMENT TO BE A NUMBER, BUT THE ARGUMENT SUPPLIED WAS NOT. THE INVALID ARGUMENT IS PRINTED.

ERROR 39 REG PDL UNDERFLOW * ERRSET * SYSTEM ERROR. A CORE DUMP FOLLOWS.;

39 SYSTEM. MORE ITEMS WERE RETRIVED FROM THE PUSHDOWN LIST THAN WERE PLACED ON IT, WITHIN THE SCOPE OF AN ERRSET. THIS IS USUALLY A BUG IN THE SYSTEM.

ERROR 40 SPEC PDL UNDERFLOW * ERRSET * SYSTEM ERROR. A CORE DUMP FOLLOWS.;

40 SYSTEM. SIMILAR TO ERROR 39, BUT FOR THE SPECIAL PUSHDOWN LIST.

USER CALLED ERR FUNCTION OUTSIDE AN ERRSET * VALUE WAS @;

41 INFORMATIONAL. THE FUNCTION ERR WAS CALLED, BUT NOT WITHIN AN ERRSET; THEREFORE, THE ERR RETURNED TO THE SYSTEM TOPLEVEL. THIS MESSAGE CALLS ATTENTION TO THAT FACT. THE ARGUMENT GIVEN TO ERR IS PRINTED.

ERROR 42 THE HEXADECIMAL NUMBER # READ ON DEVICE % IS TOO LARGE * READ;

42 MINOR. THE READ FUNCTION READ A HEXADECIMAL NUMBER WHICH HAD MORE THAN FOUR DIGITS. ONLY THE LAST FOUR DIGITS ARE USED; THE REST ARE IGNORED. THUS READING /56F5E WOULD CAUSE THIS MESSAGE TO BE PRINTED AND WOULD BE ASSUMED TO BE /6F5E.

ERROR 43 THE VARIABLE LIST @ IS BADLY STRUCTURED * PROG;

43 MAJOR. THE FUNCTION PROG RECEIVED A VARIABLE LIST WHICH WAS NOT IN THE FORM OF A LIST OF NON-NIL IDENTIFIERS. THE INVALID VARIABLE LIST IS PRINTED OUT FOR INSPECTION.

ERROR 44 THE TAG @ DOES NOT OCCUR IN THE CONTAINING PROG * GO;

44 MAJOR. AN ATTEMPT WAS MADE TO GO TO A NON-EXISTENT PROG TAG. THIS MAY BE CAUSED BY MISSPELLING THE TAG. NOTE THAT ONE MAY ONLY GO TO TAGS WITHIN THE IMMEDIATELY CONTAINING PROG.

ERROR 45 FUNCTION RETURN CALLED, BUT NOT WITHIN A PROG * ARGUMENT WAS @;

45 MAJOR. THE FUNCTION RETRUN WAS CALLED OUTSIDE OF A PROG. A RETURN TO THE SYSTEM TOPLEVEL IS MADE AND THE ARGUMENT OF THE RETURN IS PRINTED.

ERROR 46 FUNCTION GO CALLED, BUT NOT WITHIN A PROG * ARGUMENT WAS @;

46 MAJOR. SIMILAR TO ERROR 45, BUT FOR THE FUNCTION GO.

ERROR 47 FUNCTION @ REQUIRES A STRING ARGUMENT, BUT GOT @ INSTEAD;

47 MAJOR. THE FUNCTION WHOSE NAME WAS PRINTED REQUIRES AN ARGUMENT TO BE A STRING, BUT THE ARGUMENT SUPPLIED WAS NOT. THE INVALID ARGUMENT IS PRINTED.

1130 LISP 1.6 ***** END OF RUN ***** THO LONG, COME AGAIN THOON;

48 INFORMATIONAL. PRINTED WHEN THE FUNCTION QUIT IS CALLED.

ERROR 49 READ WANTS MORE CHARACTERS THAN ARE IN STRING * READSTR * STRING WAS;

49 MAJOR. THE CHARACTERS OF THE STRING GIVEN TO READSTR DID NOT FORM A COMPLETE S-EXPRESSION. THUS (READSTR , (THIS IS AN (INCOMPLETE LIST) ,) WOULD CAUSE THIS ERROR.

ERROR 50 READ ERROR WAS DUE TO READSTR FUNCTION * READSTR'S ARGUMENT WAS @;

50 MAJOR. IF DURING A READSTR OPERATION THE FUNCTION READ DETECTS AN ERROR, THIS MESSAGE IS PRINTED IMMEDIATELY AFTER THE ONE FOR THE READ ERROR. THE STRING ARGUMENT WHICH CAUSED THE ERROR IS PRINTED. NOTE THAT THE ERROR MESSAGE FOR THE READ ERROR WILL REFER TO "DEVICE 0".

DUMMY MESSAGE %;

FOR SOME REASON THE ERROR ROUTINE HAS BEEN REQUESTED TO PRINT AN ERROR MESSAGE WHICH THE ERROR FILE IS LARGE ENOUGH TO CONTAIN BUT DOESN'T. THIS MAY BE CAUSED BY A BUG IN THE SYSTEM, OR BY USING AN OUTDATED ERROR FILE.

```
********************************************************
***** APPENDIX B ***** POSSIBLE DIFFERENCES BETWEEN *****
*********************** VARIOUS 1130 LISP SYSTEMS   *****
********************************************************
```

THE SOURCE CODE OF THE 1130 LISP SYSTEM IS IN IBM 1130 MACRO ASSEMBLER LANGUAGE, AND CONTAINS A LARGE NUMBER OF "CONDITIONAL ASSEMBLY" OPTIONS. THUS, BY ASSEMBLING THE SOURCE CODE WITH PARAMETERS ADJUSTED IN VARIOUS WAYS, DIFFERENT LISP SYSTEMS CAN BE GENERATED. SOME OF THE OPTIONS ARE DESCRIBED BELOW:

1130 LISP CAN BE ASSEMBLED FOR A CORE SIZE OF 16K OR 32K. USING 32K APPROXIMATELY QUINTUPLES THE AMOUNT OF FREE STORAGE AND FIXED-POINT NUMBER SPACE AVAILABLE.

ANY OR ALL HANDLERS CAN BE INCLUDED FOR THE FOLLOWING INPUT/OUTPUT DEVICES:

DEVICE NUMBER	INPUT DEVICE	OUTPUT DEVICE
1		TYPEWRITER
2	1442 CARD READER	1442 CARD PUNCH
3		1332 PRINTER
4	1134 PAPER TAPE READER	1055 PAPER TAPE PUNCH
5		1403 PRINTER
6	KEYBOARD	
7		1627 PLOTTER
8	2501 CARD READER	

NOTE THAT IT IS POSSIBLE TO INCLUDE THE 1442 INPUT HANDLER WITHOUT THE 1442 OUTPUT HANDLER, AND VICE VERSA; AND THE 1134 WITHOUT THE 1055, AND VICE VERSA. IT IS REQUIRED THAT AT LEAST ONE PRINTER AND ONE INPUT DEVICE BE INCLUDED; IF NONE ARE SPECIFIED THE TYPEWRITER AND KEYBOARD ARE USED AS DEFAULTS. NOTE THAT INCLUDING THE KEYBOARD NECESSITATES INCLUDING THE TYPEWRITER.

VARIOUS FUNCTIONS MAY BE OMITTED IF THEY ARE NOT WANTED IN A SYSTEM. FOR EXAMPLE, THE STRING FUNCTIONS (SUBSTR, STRINDEX, ETC.) MAY BE OMITTED. THUS DIFFERENT LISP SYSTEMS MAY HAVE DIFFERENT FUNCTION SUPPLIED. THE BEST WAY TO FIND OUT WHAT FUNCTIONS ARE PRESENT IN A GIVEN SYSTEM IS TO INSPECT THE VALUE OF THE IDENTIFIER "OBLIST".

NOTE THAT THE SPACE WHICH WOULD OTHERWISE BE OCCUPIED BY ANY OMITTED FUNCTIONS OR HANDLERS IS MADE PART OF FREE STORAGE; THUS IF ONE WANTS THE GREATEST POSSIBLE AMOUNT OF FREE STORAGE FOR A GIVEN APPLICATION, ONE CAN ASSEMBLE A NEW LISP SYSTEM, OMITTING ALL UNNECESSARY DEVICE HANDLERS AND FUNCTIONS.

****

***** APPENDIX C ***** ASSEMBLING AND EXECUTING 1130 LISP *****

****

****

***** ASSEMBLING 1130 LISP *****

****

ASSEMBLY OF 1130 LISP REQUIRES THE IBM 1130 VERSION 2 DISK MONITOR SYSTEM WITH THE IBM 1130 MACRO ASSEMBLER. THE SYSTEM MUST HAVE A CARD READER (SINCE THE SOURCE CODING IS ON CARDS), A DISK DRIVE, AND AT LEAST 8K OF CORE.

ASSEMBLY USING A 1442 CARD READER, MODEL 6, WITH 16K OF CORE AND ONE DISK, WITH NO LISTING REQUESTED, TAKES APPROXIMATELY 50 TO 80 MINUTES (DEPENDING ON WHAT CONDITIONAL ASSEMBLY OPTIONS ARE TAKEN).

NOTE THAT DURING ASSEMBLY A LARGE NUMBER OF Q WARNINGS WILL BE PRINTED FOR ABOUT FIFTY LONG MDX INSTRUCTIONS WITH A SECOND OPERAND OF 0. SUCH INSTRUCTIONS INSTRUCTIONS ARE INTENDED TO BE SO. Q WARNINGS MAY ALSO BE GIVEN FOR AGO CONDITIONAL ASSEMBLY INSTRUCTIONS WITH COLUMNS 35-39 BLANK. SUCH INSTRUCTIONS ARE USED TO PRINT VARIOUS MESSAGES DURING THE ASSEMBLY, FOR EXAMPLE TO WARN THAT NO PRINTER HANDLER WAS SPECIFIED AND THAT THE TYPEWRITER HANDLER WILL BE INCLUDED.

****

***** EXECUTING 1130 LISP *****

****

EXECUTION OF 1130 LISP REQUIRES THE IBM 1130 VERSION 2 DISK MONITOR SYSTEM. THE SYSTEM MUST HAVE AT LEAST 16K OF CORE (32K IF THE LISP SYSTEM WAS ASSEMBLED FOR 32K), AT LEAST ONE DISK, AND SHOULD HAVE THE DEVICES FOR WHICH HANDLERS WERE ASSEMBLED INTO THE LISP SYSTEM. (AN ATTEMPT TO USE A DEVICE FOR WHICH A HANDLER IS AVAILABLE BUT WHICH IS NOT PHYSICALLY PRESENT CAUSES A STANDARD NOT-READY WAIT.) A FILE OF ERROR MESSAGES ("LERRS") IS KEPT ON DISK, PREFERABLY IN THE FIXED AREA OF THE MASTER CARTRIDGE.

ALL INPUT/OUTPUT DEVICE ERROR-HANDLING PROCEDURES ARE STANDARD. TWO NON-STANDARD WAITS CAN OCCUR IN $PRET (THE PRE-OPERATIVE ERROR TRAP):

(1) THE FUNCTION PAUSE WAITS WITH /AAAA /5555 IN THE ACCUMULATOR AND EXTENSION.

(2) THE 1442 OUTPUT HANDLER, IF THE 1442 INPUT HANDLER IS PRESENT, WILL READ CARDS BEFORE PUNCHING THEM TO MAKE SURE THEY ARE BLANK IN COLUMNS 1-72. IF A NON-BLANK CARD IS ENCOUNTERED WHILE PUNCHING, THE 1442 OUTPUT HANDLER WILL WAIT WITH /100B IN THE ACCUMULATOR.

NOTE THAT WHILE THE SYSTEM IS WAITING FOR KEYBOARD INPUT ONE OF THE VALUES /1111, /2222, /4444, /8888 WILL BE IN THE ACCUMULATOR AND EXTENSION.