Preserving LISP History: Source Code and All

Paul McJones

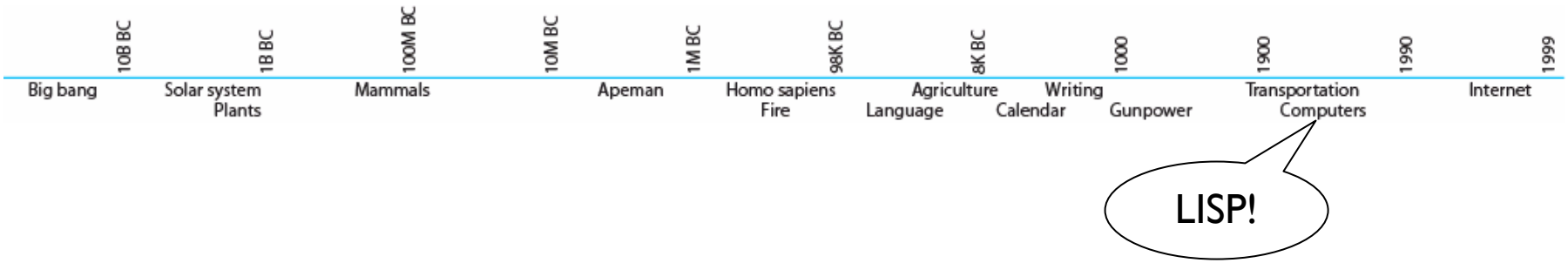
June 21, 2005

ILC 2005

Why?

- The ability to design algorithms and data structures and implement them in software was a major development of modern time

- LISP has been a test bed for language research and a production tool for 40+ years


10B BC
1B BC
100M BC
10M BC
1M BC
98K BC
8K BC
1000
1900
1990
1999
Big bang
Solar system
Plants
Mammals
Apeman
Homo sapiens
Fire
Agriculture
Language
Writing
Calendar
Gunpower
Transportation
Computers
Internet
LISP!

What?

- Our descendents will be deprived of the early history of this development unless we preserve the actual source code, manuals, and other information about historic software

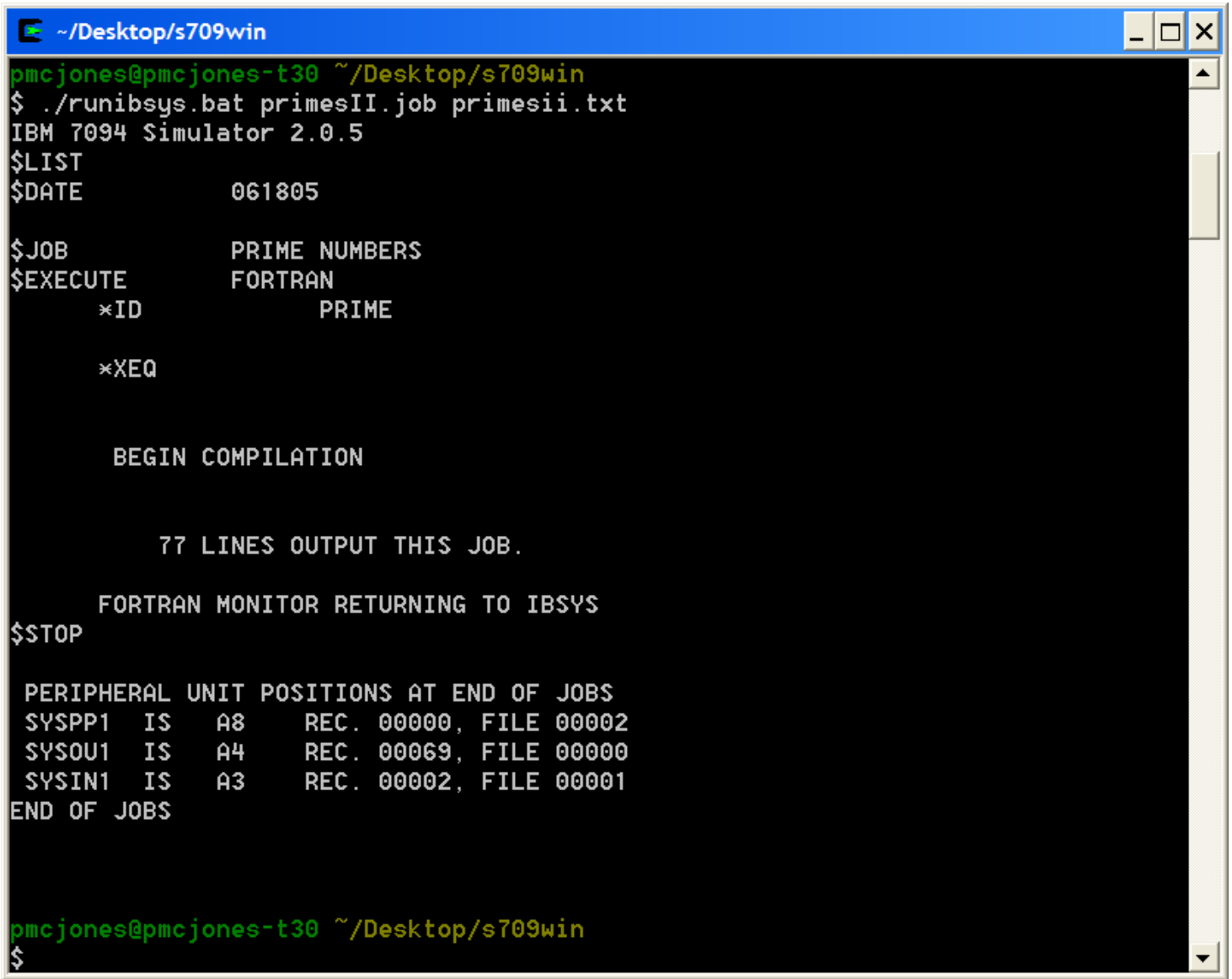
Jack Harper's work:

```
                                    2091 *                                                                                  1.5N3190
                                    2092 * CONS          BASIC LISP FUNCTION PUTS A WORD IN FREE STORAGE                    1.5N3200
                                    2093 *                                                                                  1.5N3210
03730  0634 00 4 03747              2094  CONS  SXA       CNSX,4          SAVE LINK IR                                      1.5N3220
03731 -0534 00 4 03751              2095        LXD       $FREE,4         GET FREE STORAGE LIST POINTER                     1.5N3230
03732  3 00000 4 03734              2096        TXH       **+2,4,0        SKIP IF NOT OUT OF FREE STORAGE                   1.5N3240
03733  0074 00 4 04037              2097        TSX       FROUT,4         OUT OF FREE STORAGE                               1.5N3250
03734  0771 00 0 00022              2098        ARS       18              DECREMENT TO ADDRESS                              1.5N3260
03735  0621 00 4 00000              2099        STA       0,4             PUT ADDRESS AWY                                   1.5N3270
03736  0500 00 4 00000              2100        CLA       0,4             GET POINTER TO NEXT WORD IN FREE                  1.5N3280
03737  0622 00 0 03751              2101        STD       FREE            PUT IN FREE                                       1.5N3290
03740 -0620 00 4 00000              2102        SLQ       0,4             PUT DECREMENT AWAY                                1.5N3300
03741 -0754 00 4 00000              2103        PXD       0,4             POINTER TO WORD                                   1.5N3310
03742  0774 00 4 00000              2104  CNTR1 AXT       **,4            LOW ORDER 15 BITS OF CONS COUNTER KEPT            1.5P6000
03743  2 00001 4 03746              2105        TIX       **+3,4,1        DECREMENT COUNT BY 1                              1.5P6001
03744  0074 00 4 03752              2106        TSX       ARREST,4        COUNT EXHAUSTED, RELOAD OR STOP                   1.5P6002
03745  0774 00 4 77777              2107        AXT       -1,4            RELOAD NUMBER                                     1.5P6003
03746  0634 00 4 03742              2108        SXA       CNTR1,4         PUT IN COUNTER                                    1.5P6004
03747  0774 00 4 00000              2109  CNSX  AXT       **,4            RESTORE LINK IR                                   1.5N3320
03750  0020 00 4 00001              2110        TRA       1,4             EXIT                                              1.5N3330
03751  0 00000 0 00000              2111  FREE                              POINTER TO FREE STORAGE LIST                    1.5N3340
                                    2112 *                                                                                  1.5P6005
03752 -0520 00 0 11671              2113 ARREST NZT       TCOUNT          SKIP IF COUNS COUNTER ON                          1.5P6006
03753  0020 00 4 00001              2114        TRA       1,4             OTERWISE RETURN                                   1.5P6007
03754  0601 00 0 04107              2115        STO       CNTM            SAVE AC                                           1.5P6008
03755  0500 00 0 04106              2116        CLA       CNTS            GET REST OF COUNTER                               1.5P6009
03756  0100 00 0 03763              2117        TZE       AWHOA           GO TO ERROR CALL IF EXHAUSTED                     1.5P6010
03757  0402 00 0 04110              2118        SUB       CTG             DECREMENT BY 32,768                               1.5P6011
03760  0601 00 0 04106              2119        STO       CNTS            UPDATE COUNTER                                    1.5P6012
03761  0500 00 0 04107              2120        CLA       CNTM            RESTORE AC                                        1.5P6013
03762  0020 00 4 00001              2121        TRA       1,4             EXIT TO RELOAD CNTR1                              1.5P6014
                                    2122 *                                                                                  1.5P6015
03763  0634 00 0 11671              2123  AWHOA SXA       TCOUNT,0        DEACTIVATE THE CONS COUNTER
03764  0500 00 0 04100              2124        CLA       CNTST           PICK UP  INITAL COUNT                             1.5P6017
```

Pascal Bourguignon’s work:

```
2112  *
                          2113  * CONS         BASIC LISP FUNCTION PUTS A WORD IN FREE STORAGE
                          2114  *
03730   0634 00 4 03747   2115   CONS  SXA     CNSX,4             SAVE LINK IR
03731  -0534 00 4 03751   2116         LXD     $FREE,4            GET FREE STORAGE LIST POINTER
03732   3 00000 4 03734   2117         TXH     *+2,4,0            SKIP IF NOT OUT OF FREE STORAGE
03733   0074 00 4 04037   2118         TSX     FROUT,4            OUT OF FREE STORAGE
03734   0771 00 0 00022   2119         ARS     18                 DECREMENT TO ADDRESS
03735   0621 00 4 00000   2120         STA     0,4                PUT ADDRESS AWY
03736   0500 00 4 00000   2121         CLA     0,4                GET POINTER TO NEXT WORD IN FREE
03737   0622 00 0 03751   2122         STD     FREE               PUT IN FREE
03740  -0620 00 4 00000   2123         SLQ     0,4                PUT DECREMENT AWAY
03741  -0754 00 4 00000   2124         PXD     0,4                POINTER TO WORD
03742   0774 00 4 00000   2125   CNTR1 AXT     **,4               LOW ORDER 15 BITS OF CONS COUNTER KEPT
03743   2 00001 4 03746   2126         TIX     *+3,4,1            DECREMENT COUNT BY 1
03744   0074 00 4 03752   2127         TSX     ARREST,4           COUNT EXHAUSTED, RELOAD OR STOP
03745   0774 00 4 77777   2128         AXT     -1,4               RELOAD NUMBER
03746   0634 00 4 03742   2129         SXA     CNTR1,4            PUT IN COUNTER
03747   0774 00 4 00000   2130   CNSX  AXT     **,4               RESTORE LINK IR
03750   0020 00 4 00001   2131         TRA     1,4                EXIT
03751   0 00000 0 00000   2132   FREE                             POINTER TO FREE STORAGE LIST
                          2133  *
03752  -0520 00 0 11671   2134  ARREST NZT     TCOUNT             SKIP IF COUNS COUNTER ON
03753   0020 00 4 00001   2135         TRA     1,4                OTERWISE RETURN
03754   0601 00 0 04107   2136         STO     CNTM               SAVE AC
03755   0500 00 0 04106   2137         CLA     CNTS               GET REST OF COUNTER
03756   0100 00 0 03763   2138         TZE     AWHOA              GO TO ERROR CALL IF EXHAUSTED
03757   0402 00 0 04110   2139         SUB     CTG                DECREMENT BY 32,768
03760   0601 00 0 04106   2140         STO     CNTS               UPDATE COUNTER
03761   0500 00 0 04107   2141         CLA     CNTM               RESTORE AC
03762   0020 00 4 00001   2142         TRA     1,4                E7IT TO RELOAD CETR1
                          2143  *
03763   0634 00 0 11671   2144   AWHOA SXA     TCOUNT,0           DESACTIVATE THE CONS COUNTER
03764   0500 00 0 04100   2145         CLA     CNTST              PICK UP  INITIAL COUNT
```

Dave Pitts' work



What else?

- LISP I and LISP 1.5 for IBM 704, 709, 7090; LISP 1.5 for CTSS; LISP 1.5 for Univac M-460; LISP 1.5 for AN/FSQ-32/V; LISP 1.5 at Stanford; SHARE LISP 1.5; LISP 2; LISP 1.5 for Univac 1108; Basic PDP-1 Lisp; LISP 1.5/1.6/MACLISP for PDP-6/10; BBN LISP; INTERLISP; Stanford LISP 1.6; IBM Lisp; Multics Lisp; Zetalisp; Interlisp-D; Scheme; T; New Implementation of Lisp (NIL); S-1 Lisp; Spice Lisp; Standard LISP, Portable Standard Lisp (PSL); Franz Lisp; VLISP; Le_Lisp; UtiLisp; Common Lisp; EuLisp; ISLISP; Emacs Lisp

LISP 1.5 Programmer's Manual

The Computation Center
and Research Laboratory of Electronics

Massachusetts Institute of Technology

The M. I. T. Press
Massachusetts Institute of Technology
Cambridge, Massachusetts

The Programming Language LISP: Its Operation and Applications

Information International, Inc.

The M.I.T. Press
Massachusetts Institute of Technology
Cambridge, Massachusetts, and London, England

(LISP 1.5 PRIMER
(BY
(CLARK WEISSMAN)))

INTERLISP

REFERENCE MANUAL

DANIEL WEINREB & DAVID MOON

LISP MACHINE MANUAL

What’s still needed?

- More source code
  - LISP 2, INTERLISP-10, MacLisp, Standard Lisp, LISP/360, …
- More manuals
- Oral histories
- Photographs

- And much more!

How Do I Get Involved?

- The Computer History Museum is committed to supporting efforts like this, but we (I!) need your help.

- Visit:

  community.computerhistory.org/scc/projects/LISP/

  or Google “lisp history”

- Contact me:

Paul McJones
paul@mcjones.org

navigation

- Home
- SCC Projects
  - FORTRAN and FORTRAN II
  - LISP
    - images
    - EuLisp
    - KCL
  - NLS / AUGMENT
- SCC Meetings
- Calendar
- Software Repository
- Members
- Groups
- Help and Tips
- About the SCC
- Can you help?
- Contact Us

recent items

History of LISP 2005-04-24

History of LISP

This web site is a pilot project of the Computer History Museum's Software Collection Committee to develop expertise in the collection, preservation, and presentation of historic software. The specific goal is to locate source code, design documents, and other materials concerning the original LISP I/1.5 system, and, as time permits, its many follow-ons. LISP was one of the earliest high-level programming languages and introduced many ideas such as garbage collection, recursive functions, symbolic expressions, and dynamic type-checking. The project is being carried out by Paul McJones (email: paul at mcjones dot org), who maintains a related blog Dusty Decks .

Last updated June 19, 2005.

Contents

upcoming events

SCC Meeting 2005-06-22 Hopper, CHM, 2005-06-22

No SCC meeting in July 2005-07-31

SCC Meeting 2005-08-24 Hopper, CHM, 2005-08-24

SCC Meeting 2005-09-21 Hopper, CHM, 2005-09-21

« June 2005 »

Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		