File created: 30-MAY-79 00:25:14 **<LISPUSERS>CIRCLPRINT.;3**

changes to:

Read Table: OLD-INTERLISP-FILE

Package: INTERLISP

Format: XCCS

```
(RPAQQ CIRCLPRINTCOMS
       [(FNS CIRCLMAKER CIRCLMAKER1 CIRCLPRINT CIRCLMARK RLPRIN1 RLRESTORE CSEARCH PLACEPRINT C2PRINT ROOMLEFT
             RLPRIN2 CPRINT CLPRINT CIRCLNC)
        (VARS CIRCLMARKER)
        (BLOCKS (CIRCLBLOCK CIRCLMAKER CIRCLMAKER1 CIRCLPRINT CIRCLMARK RLPRIN1 RLRESTORE CSEARCH PLACEPRINT
                       C2PRINT ROOMLEFT RLPRIN2 CPRINT CLPRINT CIRCLNC (ENTRIES CIRCLPRINT CIRCLMARK RLPRIN1
                                                                              RLPRIN2 RLRESTORE CIRCLMAKER
                                                                              CIRCLMAKER1)
                       (SPECVARS RLKNT LABELIST REFLIST)
                       (LOCALFREEVARS PLACELIST LL)
                       (NOLINKFNS . T)
                       (GLOBALVARS #UNDOSAVES RESETVARSLST])

(DEFINEQ
```

(CIRCLMAKER

```
  [LAMBDA (L)                                                    (* lmm: 19 MAY 75 232)
    (PROG (LABELIST REFLIST)
          (CIRCLMAKER1 L)
```

(* An error is generated if REFLIST is non-NIL, because this means that some nodes are referenced in L without being labeled (i.e. defined) and indicates that the user may have omitted some code.
See the comments in CIRCLMAKER1 for additional information.)

```
          (COND
             (REFLIST (ERROR (QUOTE "LABEL NOT FOUND FOR REFERENCE")))
             (T (RETURN L])
```

(CIRCLMAKER1

```
  [LAMBDA (L)
```

(* This function makes use of two free variables, REFLIST and LABELIST, which are alist-like structures that provide information about nodes that have been defined (labeled) and where they have been referenced.
LABELIST is a list of lists. Each list in labelist is of the form (n . L), where n is the number that is being used to label the node L. REFLIST is also a list of lists. Each list in REFLIST is of the form
(n (L1 L2 ...) (L1' L2' ...)), where n is the number of the node that is being referred to, and L1, L2,...
are those nodes for which car is a reference to {n}, and L1', L2',...
are those nodes for which cdr is a reference to {n}.)

```
    (PROG (CARL CDRL CARLN CDRLN NODECODE)
          [COND
             ((LISTP L)
              (SETQ CARL (CAR L))
              (SETQ CDRL (CDR L]
          (COND
             [(AND CARL (ATOM CARL))
              (COND
                 [[AND (NEQ CARL (QUOTE {}))
                       (EQ (NTHCHAR CARL 1)
                           (QUOTE {))
                       (EQ (NTHCHAR CARL -1)
                           (QUOTE }))
                       (FIXP (SETQ CARLN (MKATOM (SUBSTRING CARL 2 -2]
```

(* This part of the conditional checks to see whether CARL is a reference to a node, i.e.
an atom of the form {n}, where n is a number. If so, then we check to see whether the node has yet been labeled
(this is the assoc on labelist), whether it has been previously referred to, but not labelled
(this is the assoc on reflist), or else assume that it has neither been referred to nor labelled previously.
In the first case, we replace (/rplaca) CARL with a pointer to the node that is labelled n
(as explained above, this would be cdr of nodecode)%. In the second case, we add a pointer to L to
(cadr nodecode) which is that part of REFLIST which contains pointers to those nodes for which car is the atom {n}.
In the final case, we add a new sublist to REFLIST, to record that a reference has been made to a node numbered n.
It should be noted that, throughout this program, all such manipulations are undoable, so that the user can
(for example) control-D and undo circlmaker to regain his original expression.)

```
                  (COND
                     ((SETQ NODECODE (ASSOC CARLN LABELIST))
                      (/RPLACA L (CDR NODECODE)))
                     ((SETQ NODECODE (ASSOC CARLN REFLIST))
                      (/NCONC1 (CADR NODECODE)
                               L))
                     (T (SETQ REFLIST (/NCONC1 REFLIST (LIST CARLN (LIST L)
                                                             NIL]
                 ([AND (NEQ CARL (QUOTE *))
                       (NEQ CARL (QUOTE **))
                       (EQ (NTHCHAR CARL 1)
                           (QUOTE *))
```

```
                    (EQ (NTHCHAR CARL -1)
                        (QUOTE *))
                    (FIXP (SETQ CARLN (MKATOM (SUBSTRING CARL 2 -2]
```

(* This part of the conditional checks to see whether CARL is a label for a node, i.e.
an atom of the form *n*, where n is a number. If so, then we check to see whether a node has already been labelled by the same number, in which case an error is generated (*n* is ambiguous, multiply used label)%.
Otherwise, we check to see that cdr of l is a list, in which case we physically
(but undoably) remove the label from L , add a new sublist to LABELIST, and check to see whether there is a sublist on REFLIST that indicates that references to L have already been made.
If so, then we make the appropriate changes to car and cdr of the nodes wwich referenced L, and undoably remove the sublist (NODECODE) from REFLIST, because the only purpose of this sublist was to preserve a record of those nodes which referenced L, before the label for L was found and the actual pointers from those nodes to L could be established--now that the pointers have been established, we can decrement REFLIST--
if all nodes that are referenced are in fact labeled somewhere in the structure, then REFLIST should eventually in this way be decremented to NIL-- otherwise the monitoring function CIRCLMAKER will generate an error, if it was the top level function that called CIRCLMAKER1. Finally, we call circlmaker1 recursively on L, and return the resultant value of L.
If the original cdr of l were not a list, then we would generate the CARL "IS MISPLACED LABEL" error--
this is intended to handle expressions like (*1* . {1}), which are anomalous because the label does not really point to a node, and there is a likelihood that the user has inadvertently omitted some code.)

```
            (COND
               ((ASSOC CARLN LABELIST)
                (ERROR CARL (QUOTE "IS AMBIGUOUS, MULTIPLY USED LABEL")))
               ((LISTP (CDR L))
                (/RPLACA L (CADR L))
                (/RPLACD L (CDDR L))
                (SETQ LABELIST (/NCONC1 LABELIST (CONS CARLN L)))
                [COND
                   ((SETQ NODECODE (ASSOC CARLN REFLIST))
                    (SETQ REFLIST (/DREMOVE NODECODE REFLIST))
                    [MAPC (CADR NODECODE)
                          (FUNCTION (LAMBDA (X)
                                            (/RPLACA X L]
                    (MAPC (CADDR NODECODE)
                          (FUNCTION (LAMBDA (X)
                                            (/RPLACD X L]
                (CIRCLMAKER1 L)
                (RETURN L))
               (T (ERROR CARL (QUOTE "IS MISPLACED LABEL"]
      ((LISTP CARL)
       (CIRCLMAKER1 CARL)))
   (COND
      [(AND CDRL (ATOM CDRL))
       (COND
          [[AND (NEQ CDRL (QUOTE {}))
                (EQ (NTHCHAR CDRL 1)
                    (QUOTE {))
                (EQ (NTHCHAR CDRL -1)
                    (QUOTE }))
                (FIXP (SETQ CDRLN (MKATOM (SUBSTRING CDRL 2 -2]
```

(* This branch of the conditional checks to see whether CDRL is a reference to a node, i.e.
an atom of the form {n} where n is a number. If so, then the steps followed are analogous to those described above for the case in which CARL is a reference to a node. A similar check is made below to see whether CDRL is a label for a node.
An error is generated in this case, because such a label has no meaning
(the label does not point to a node), and there is a likelihood that the user has inadvertently omitted some code.
The error message reads "*n* IS MISPLACED LABEL" %.)

```
            (COND
               ((SETQ NODECODE (ASSOC CDRLN LABELIST))
                (/RPLACD L (CDR NODECODE)))
               ((SETQ NODECODE (ASSOC CDRLN REFLIST))
                (/NCONC1 (CADDR NODECODE)
                         L))
               (T (SETQ REFLIST (/NCONC1 REFLIST (LIST CDRLN NIL (LIST L]
          ([AND (EQ (NTHCHAR CDRL 1)
                    (QUOTE *))
                (NEQ CDRL (QUOTE *))
                (NEQ CDRL (QUOTE **))
                (EQ (NTHCHAR CDRL -1)
                    (QUOTE *))
                (FIXP (MKATOM (SUBSTRING CDRL 2 -2]
           (ERROR CDRL (QUOTE "IS MISPLACED LABEL"]
      ((LISTP CDRL)
       (CIRCLMAKER1 CDRL)))
   (RETURN L])
```

(**CIRCLPRINT**

```
  [LAMBDA (L PRINTFLG RLKNT)                                    (* lmm: 24-JAN-76 1 22)
    (RESETLST
        (RESETSAVE (RADIX 10))
        (RESETVARS (#UNDOSAVES)
                   (PROG NIL
                         (CIRCLMARK L RLKNT)
                         (COND
```

```
                           (PRINTFLG (RLPRIN1 L))
                           (T (RLPRIN2 L)))
                        (RLRESTORE L))))
    L])
```

(CIRCLMARK

```
  [LAMBDA (L RLKNT)                                          (* lmm: 19 MAY 75 233)
    (PROG NIL
          (COND
             ((NULL RLKNT)
              (SETQ RLKNT 0)))
          (RETURN (CSEARCH L])
```

(RLPRIN1

```
  [LAMBDA (L)
    (PROG (PLACELIST LL)
          (SETQ PLACELIST NIL)
          (SETQ LL (LINELENGTH))
          (CLPRINT L (QUOTE CAR))
          (TERPRI])
```

(RLRESTORE

```
  [LAMBDA (L)                                                (* lmm: 19 MAY 75 234)
    (PROG NIL
          (COND
             ((AND (LISTP L)
                   (EQ (CAAR L)
                       CIRCLMARKER))
              (RPLACA L (CADAR L))
              (RLRESTORE (CAR L))
              (RLRESTORE (CDR L])
```

(CSEARCH

```
  [LAMBDA (L)                                                (* lmm: 19 MAY 75 234)
    (PROG (NXPOINT)
          (COND
             ((LISTP L)
              (SETQ NXPOINT (CAR L))
              [COND
                 [(LISTP NXPOINT)
                  (COND
                     ((EQ (CAR NXPOINT)
                          CIRCLMARKER)
                      [COND
                         ((NULL (CDDR NXPOINT))
                          (NCONC1 NXPOINT (SETQ RLKNT (ADD1 RLKNT]
                      (RETURN NIL))
                     (T (/RPLACA L (LIST CIRCLMARKER NXPOINT))
                        (CSEARCH NXPOINT]
                 (T (/RPLACA L (LIST CIRCLMARKER NXPOINT]
              (RETURN (CSEARCH (CDR L])
```

(PLACEPRINT

```
  [LAMBDA (P)
    (COND
       (P (COND
             ((IGREATERP (CDAR P)
                         -1)
              (SPACES (IDIFFERENCE (CDAR P)
                                   (POSITION)))
              (PRIN1 (CAAR P))
              (RPLACD (CAR P)
                      -1)))
          (PLACEPRINT (CDR P])
```

(C2PRINT

```
  [LAMBDA (L CAMEFROM)                                       (* lmm: 19 MAY 75 234)
    (PROG NIL
          (TERPRI)
          (PLACEPRINT PLACELIST)
          (TERPRI)
          (TERPRI)
          (RETURN (CPRINT L CAMEFROM])
```

(ROOMLEFT

```
  [LAMBDA NIL
    (IDIFFERENCE LL (POSITION])
```

(RLPRIN2

```
  [LAMBDA (L)
    (RESETLST
        (PROG (PLACELIST LL)

          (* Rather than checking NCHARS of every atom before printing it just to make sure that lines don't go over the boundary
          (causing LISP to put in TERPRI's where we don't want them) LINELENGTH is just changed to be something huge and we
          are conservative about how much we think will fit on a line... If a structure has atoms with more than 8 characters, and
          appears on the end of a line, it might overflow, though)

              (SETQ LL (IDIFFERENCE (LINELENGTH)
                            8))
              (RESETSAVE (LINELENGTH (IPLUS LL 80)))
              (CPRINT L (QUOTE CAR))
              (TERPRI)
              (PLACEPRINT PLACELIST)
              (TERPRI)))])
```

(CPRINT

```
  [LAMBDA (L CAMEFROM)

          (* This function does most of the work involved in circlprinting in the double line format.
          IN this format, a node is labeled by the appearance of its number on the line below where the node begins.
          If the node is car of the node we came from (i.e. if CAMEFRON = 'CAR) then the node begins with a left parens, and the
          node's number should begin immediately below that left parens.
          If the node is cdr of the node we came from (i.e. if CAMEFROM = 'CDR) then the node is a tail of a list, and the number
          identifying it should appear on the line below the beginning of that tail.
          Thus, when labeeling a node we have to save the position where the node begins.
          Also, we have to alternate printing nodes and printing their labels below them.
          The function that prints labels below nodes is PLACEPRINT. CPRINT saves the information necessary to print labels in the
          correct position by adding sublists to an alist called PLACELIST.
          When CPRINT adds a sublist to PLACELIST, this sublist is of the form
          (N . P), where N is the number of the ndoe being labeled, and P is the position where printing of the label should begin.
          When PLACEPRINT prints a line of labels, it merely cdr's thru PLACELIST checking to see if there are any sublists of the
          form (N . P), where P is greater than -1.0 For each such sublist, N is printed and then the sublist is physically altered to be of
          the form (N . -1)%. Thus PLACELIST can also be used as a list of all labels that have been printed
          (or will be printed, when PLACEPRINT is called next), and CPRINT can merely do an assoc on placelist to see if a given
          node has been previously labeled.)

    (PROG (LN N CARL CARLN CDRL CDRLN LABELEDCDRL? LABELEDCARL? EXSPACES ROOM)
          (COND
             ((ILESSP (SETQ ROOM (ROOMLEFT))
                      3)
              (C2PRINT L CAMEFROM))
             ((OR (NLISTP L)
                  (NLISTP (CAR L))
                  (NEQ (CAAR L)
                       CIRCLMARKER))
              (ERROR (QUOTE "UNCIRCLMARKED LIST STRUCTURE")))
             ((AND (SETQ LN (CDDAR L))
                   (SETQ N (CAR LN))
                   (FASSOC N PLACELIST))                          (* L has already been printed; print a back reference)
              [COND
                 ((ILESSP ROOM (IPLUS 2 (CIRCLNC N)))
                  (RETURN (C2PRINT L CAMEFROM]
              (PRIN1 (QUOTE {))
              (PRIN1 N)
              (PRIN1 (QUOTE }))))
             ([AND LN (ILESSP ROOM (IPLUS 3 (SETQ EXSPACES (CIRCLNC N]
              (C2PRINT L CAMEFROM))
             (T [COND
                   (LN (SETQ PLACELIST (NCONC1 PLACELIST (CONS (CAR LN)
                                                               (POSITION]
                                                                (* If LN is not NIL, the structure needs to be labeled)
                     [COND
                        ((EQ CAMEFROM (QUOTE CAR))
                         (PRIN1 (QUOTE %(]
                     [COND
                        (LN (COND
                               ((OR (NEQ EXSPACES 1)
                                    (NEQ CAMEFROM (QUOTE CAR)))       (* Make sure there is enough space to clearly label L)
                                (SPACES EXSPACES]
                     [COND
                        ((NLISTP (SETQ CARL (CADAR L)))
                         (PRIN2 CARL))
                        ((EQ L CARL)
                         (COND
                            ((ILESSP (ROOMLEFT)
                                     (IPLUS 2 EXSPACES))
                             (TERPRI)
                             (PLACEPRINT PLACELIST)
                             (TERPRI)
                             (TERPRI)))
                         (PRIN1 (QUOTE {))
                         (PRIN1 (CAR LN))
                         (PRIN1 (QUOTE })))
                        (T (CPRINT CARL (QUOTE CAR]
```

```
               (COND
                  ((NULL (CDR L))
                   (PRIN1 (QUOTE %))))
                  ((NLISTP (CDR L))
                   (PRIN1 (QUOTE " ."))
                   (SPACES 1)
                   (PRIN2 (CDR L))
                   (PRIN1 (QUOTE %))))
                  ((AND (SETQ CDRLN (CDDADR L))
                        (FASSOC (CAR CDRLN)
                                PLACELIST))                    (* If (CDR L) has been labeled, then print a reference.)
                   (COND
                      ([ILESSP (ROOMLEFT)
                               (IPLUS 6 (CIRCLNC (CAR CDRLN]
                       (TERPRI)
                       (PLACEPRINT PLACELIST)
                       (TERPRI)
                       (TERPRI)))
                   (PRIN1 (QUOTE " ."))
                   (SPACES 1)
                   (PRIN1 (QUOTE {))
                   (PRIN1 (CAR CDRLN))
                   (PRIN1 (QUOTE }))
                   (PRIN1 (QUOTE %))))
                  (T (SPACES 1)
                     (CPRINT (CDR L)
                             (QUOTE CDR])
```

(**CLPRINT**

```
  [LAMBDA (L CAMEFROM)
```

(* This function does most of the work involved in cirlcprinting in the single line format.
The problems encountered with the double line format (see CPRINT) do not occur here.
In particular the alist PLACELIST is used by CLPRINT only to store the numbers of the reentrant nodes that have already been labeled, not the positions where they were labeled. CLPRINT prints a description of each node
(i.e. a label, a reference, or car and cdr) as it encounters it.)

```
   (PROG (LN N LABELIT CARL CDRL CDRLN LISTPCDRL? LABELEDCDRL? EXSPACES)
         (COND
            ((ILESSP (ROOMLEFT)
                     2)
             (TERPRI)))
         (COND
            ((LISTP L)
             (COND
                ((NOT (EQ (CAAR L)
                          CIRCLMARKER))
                 (ERROR (QUOTE "UNCIRCLMARKED LIST STRUCTURE")))
                ((AND (SETQ LN (CDDAR L))
                      (SETQ N (CAR LN))
                      (SETQ EXSPACES (CIRCLNC N))
                      (FMEMB N PLACELIST))
```

(* If L is a reentrant node and has already been labeled then we simply print a reference to L.
CIRCLNC computes the number of digits in N which is 2 less than the number of characters needed to label or reference L, and makes it the value of EXSPACES. If N is greater than or equal to 10000, then an error is generated.
Otherwise, the FMEMB on placelist checks to see if L has already been labeled, in which case we print a reference to L, in the code below this comment.)

```
                 (COND
                    ((ILESSP (ROOMLEFT)
                             (IPLUS 2 EXSPACES))
                     (TERPRI)))
                 (PRIN1 (QUOTE {))
                 (PRIN1 N)
                 (PRIN1 (QUOTE }))))
                (T (COND
                      (LN                                      (* Checks to see if L has to be labeled.)
                          (COND
                             ((ILESSP (ROOMLEFT)
                                      (IPLUS 3 EXSPACES))
                              (TERPRI)))
                          (SETQ LABELIT T)))
                   [COND
                      (LABELIT                                 (* If L is to be labeled, then add N to placelist.)
                               (SETQ PLACELIST (NCONC PLACELIST LN]
                   [COND
                      ((EQ CAMEFROM (QUOTE CAR))
                       (PRIN1 (QUOTE %(]
                   (COND
                      (LABELIT                                 (* If L is to be labeled, then print a label.)
                               (PRIN1 (QUOTE *))
                               (PRIN1 (CAR LN))
                               (PRIN1 (QUOTE *))
                               (SPACES 1)))
                   (COND
```

```
                    [(LISTP (SETQ CARL (CADAR L)))
```

(* If CARL is a list then if L = CARL then print a reference to CARL automatically, else CPRINT CARL.)

```
                     (COND
                        ((EQ L CARL)
                         (COND
                            ((ILESSP (ROOMLEFT)
                                     (IPLUS 2 EXSPACES))
                             (TERPRI)))
                         (PRIN1 (QUOTE {))
                         (PRIN1 (CAR LN))
                         (PRIN1 (QUOTE }))))
                        (T (CLPRINT CARL (QUOTE CAR]
                    (T (PRIN2 CARL)))
                 (COND
                    ((LISTP (SETQ CDRL (CDR L)))          (* Check whether CDRL needs to be labeled.)
                     (COND
                        ((AND (SETQ CDRLN (CDDADR L))
                              (FMEMB (CAR CDRLN)
                                     PLACELIST))
                         (SETQ LABELEDCDRL? T))
                        (T (SETQ LABELEDCDRL? NIL)))
                     (SETQ LISTPCDRL? T))
                    (T (SETQ LISTPCDRL? NIL)))
                 (COND
                    (CDRL                                 (* make sure there will be a space between carl and cdrl.)
                          (SPACES 1)))
                 (COND
                    [LISTPCDRL?
```

(* If CDRL has been labeled and is reentrant, then print a reference.
Else if CDRL is a list the CLPRINT CDRL, else just prin1 it.)

```
                           (COND
                              (LABELEDCDRL? (SETQ N (CAR CDRLN))
                                     (COND
                                        ((ILESSP (ROOMLEFT)
                                                 (IPLUS 5 (CIRCLNC N)))
                                         (TERPRI)))
                                     (PRIN1 (QUOTE %.))
                                     (SPACES 1)
                                     (PRIN1 (QUOTE {))
                                     (PRIN1 (CAR CDRLN))
                                     (PRIN1 (QUOTE }))
                                     (PRIN1 (QUOTE %))))
                              (T (CLPRINT CDRL (QUOTE CDR]
                    ((NULL CDRL)
                     (PRIN1 (QUOTE %))))
                    (T (PRIN1 (QUOTE %.))
                       (SPACES 1)
                       (PRIN2 CDRL)
                       (PRIN1 (QUOTE %)])
```

(**CIRCLNC**

```
  [LAMBDA (N)
    (COND
       ((ILESSP N 10)
        1)
       ((ILESSP N 100)
        2)
       ((ILESSP N 1000)
        3)
       ((ILESSP N 10000)
        4)
       (T (ERROR (QUOTE "REENTRANT NODE HAS BEEN NUMBERD OVER 10000"])

)

(RPAQQ CIRCLMARKER "BEENHERE")

(DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY

(BLOCK: CIRCLBLOCK CIRCLMAKER CIRCLMAKER1 CIRCLPRINT CIRCLMARK RLPRIN1 RLRESTORE CSEARCH PLACEPRINT C2PRINT
       ROOMLEFT RLPRIN2 CPRINT CLPRINT CIRCLNC (ENTRIES CIRCLPRINT CIRCLMARK RLPRIN1 RLPRIN2 RLRESTORE
                                                       CIRCLMAKER CIRCLMAKER1)
       (SPECVARS RLKNT LABELIST REFLIST)
       (LOCALFREEVARS PLACELIST LL)
       (NOLINKFNS . T)
       (GLOBALVARS #UNDOSAVES RESETVARSLST))
)
```

FUNCTION INDEX

VARIABLE INDEX