SLAC Computation Research Group
Stanford, California

CGTM No. 172
March 1976

TIDY

Roger B. Chaffee
Computation Research Group
Stanford Linear Accelerator Center

TIDY was written by Harry M. Murphy, Jr.[1] This version has been modified considerably, by Gerry Tool at LRL Berkeley, by Alice Barlow at NASA Ames, and by myself at LRL and at SLAC. This writeup is similarly descended from the original report: features have been added and some of the original material has been changed, but the basic structure remains that of Mr. Murphy.

---

(1) Murphy, Harry M., Jr., "TIDY, a Computer Code for Renumbering and Editing FORTRAN Source Programs", Technical Report No. AFWL-TR-66-93, Air Force Weapons Laboratory, Kirtland Air Force Base (October 1966)

Table of Contents

Section I

Introduction

This report describes TIDY, a FORTRAN computer code for renumbering, editing, and--in general--tidying FORTRAN source programs whose statement numbering has become unwieldy and whose readability has deteriorated as a result of many revisions, patches and corrections. Such programs frequently result when many programmers work together on the development of a very complex computer code, or when the FORTRAN source has been produced from a higher-level language by a pre-processor such as MORTRAN.(2)

Often such codes become so complex in their statement numbering that weeks of study are required before one can begin to understand the logical flow within the routines. Even after a programmer is familiar with a program, much time is wasted in trying to locate numbered statements and in generating new statement numbers that have not been used already.

TIDY provides a simple means for "cleaning up" such complex codes. TIDY reads the old FORTRAN program routine-by-routine, and prepares and punches a new version of the program, which has the following characteristics:

(1) All statement numbers increase in consecutive order. They may be left- or right-justified to any column, according to the user's style.

(2) Statement numbers are assigned only to statements that are referred to by other statements.

(3) All statement number references are updated to conform to the new statement number assignments.

(4) FORMAT statements may optionally be collected from within the body of the routine and placed at the end.

(5) Only those FORMAT and CONTINUE statements actually referred to within the routine are retained.

(6) As an optional feature, all CONTINUE statements, except those used to terminate a do-loop, may be removed, and references to them rerouted to the next executable statement. In the same option, references to an unconditional GO TO statement will be changed to refer directly to the target of the GO TO.

(7) Excess blanks are deleted from each statement,

---

(2) Cook, A.J., and Shustek, L.J., "A User's Guide to MOPTRAN2", Computation Group Technical Memo No. 165 (unpublished).

and blanks are inserted as necessary to ensure uniformity and improve readability.

(8) Consecutive blank comment cards in excess of two are deleted.

(9) All statements in each new routine are optionally labeled in columns 73 through 79 with a unique letter-number combination (e.g., "B 75", "B 76", "C 5", and so forth. The letter (or letters) indicates the routine, while the number indicates the position of the statement in the routine. For example, the third card in the second routine will normally be labeled "B 3", while the seventh card in the third routine would be "C 7".

(10) In addition to the labeling described above, column 80 of each END statement is punched with a " - " sign (11-punch). This allows automatic page ejection when listing TIDY-processed routines on machines that have the "X-skip" feature.

(11) Certain FORTRAN-II statements such as "READ INPUT TAPE 12, 34, X" and "WRITE TAPE 13" are rewritten in the FORTRAN IV format as "READ (12,34) X" and "WRITE (13)" and so forth.

TIDY recognizes a number of control cards, described in Section II, that permit deletion of comments, variations in statement numbering, suppression of FORMAT collection, card punching, and so forth.

In addition to the new "cleaned up" programs, TIDY offers a limited set of FORTRAN diagnostics that comment on such FORTRAN errors and trouble spots as missing or duplicate statement numbers, incorrect parenthesis counts, illegal DO-loop indexing, illegal statements, and inaccessible parts of the program. If the old program contains references to nonexistant statement numbers, TIDY generates pseudo-statement numbers and uses these numbers in the referencing statements so that the programmer can make the necessary corrections to his program with minimum repunching.

An optional statement number directory lists all new statement numbers, the corresponding old statement numbers, and the location in the old program of the statements.

TIDY accepts and processes all FORTRAN programs written in accordance with the ASA Standard for FORTRAN.(3 4 5 6 7.)

---

(3) Heising, W.P., "History and Summary of FORTRAN Standardization and development for the ASA," Comm. of the ACM Vol 7. No, 10, October 1964, p. 590.
(4) ASA Sectional Committee X3, "A Programming Language for Information Processing on Automatic Data Processing Systems," Comm. of the ACM, Vol. 7, No. 10, October 1964,

In general, this standard includes all statements in what are called the "FORTRAN II" and "FORTRAN IV" languages. In addition, TIDY accepts many CDC[8] and IBM[9] dialect statements. The programmer who uses TIDY to process programs containing dialect statements should check his TIDY-output very carefully for errors.

TIDY accepts and processes FORTRAN statement with up to 19 continuation cards.

Multiple instruction statements with dollar signs separating the instructions, which are allowed in the CDC dialect, are not processed correctly by TIDY. However, they are flagged in the output listing, and could be re-coded by the programmer after the TIDY job.

Machine language routines cannot be processed by TIDY. However, if the first card of such routines has as its first word "IDENT", "MACHINE" or "ASCENT", the entire routine will be copied down to and including the next END statement. The punched output will be appropriately labeled in columns 73-80.

---

pp. 591-625.
(5) American Standard FORTRAN, X3.9-1966, American Standards Association, 10 East 40th St., New York, NY
(6) American Standard Basic FORTRAN, X3.10-1966, American Standards Association, 10 East 40th St., New York, NY
(7) "FORTRAN Accepted as American Standard," Comm. of the ACM, Vol 9, No. 7, July 1966, p 537.
(8) "Chippewa Operating System FORTRAN Reference Manual", Control Data Corporation Publication No. 60132700.
(9) "IBM System/360 and System/370 FORTRAN IV Language" Order No. GC28-6515-8.

SECTION II

Using TIDY

General Instructions

Since TIDY is a large FORTRAN computer code, it would be impractical to compile it each time it is to be run. A compiled version should be available as a deck, a disk file, or a tape file. The control cards necessary to run TIDY are given in Appendix I (p. 20).

Whatever the exact arrangements, bear in mind that TIDY is a FORTRAN program itself, and differs from ordinary programs only in that it reads another FORTRAN program or programs as its data deck. The data deck may be an actual FORTRAN source card deck, or card images on an input file.

The input to TIDY should consist of the program with all the subroutines and functions, if any, in the order that you wish them processed. If you wish, you may include more than one program in a single job. On the other hand, you may process only one or two routines from a program. Each routine must terminate with an END statement.

Finally, after the last END statement of the last routine to be processed, place two TIDY control cards, "*LAST" and "*STOP".(10)

TIDY accepts certain control cards, identified by an asterisk "*" in column one, which specify the execution of certain options. These control cards consist of four- or six-letter codes punched anywhere on a single input card. As in FORTRAN, blanks are ignored. For clarity, other letters and symbols may follow the control card keyword itself. The control cards recognized by TIDY are shown in Table I.

Some control cards, for instance *BASE, have an associated value. The value may appear anywhere on the card following the keyword, and must be preceded by an equals sign and followed by a period.

Cards with "**" punched in columns one and two are ignored by TIDY and may be used to comment on the program being processed.

Cards with * in column 1 which are neither control cards nor comments are copied directly to the output file.

The control cards are sensed by TIDY during the first

---

(10) The SLAC version of TIDY tests for the end of the input file while reading, and will generate its own *LAST cards. However, this feature is not available in standard FORTRAN, and may not be implemented.

pass. Some of the corresponding options, such as the option to suppress FORMAT statement collection, or the *SKIP option, are immediate in effect. Others, such as the statement numbering options or request for a cross-reference list of statement numbers, become effective only after the first pass. If conflicting control cards of the latter type appear in a routine, (e.g. *CARDS and *NOCARDS), the control card appearing last in the routine will dominate.

With obvious exceptions, such as *SKIP, all TIDY options remain in effect until countermanded by a later control statement. Thus, a control statement that appears in one routine will remain in effect in all the following routines until cancelled.

The individual TIDY control cards are discussed below. Preceeding each discussion is an example of the control statement, with the required portion of the control word underlined. In addition, those codes that are immediate in action are labeled "(IMMEDIATE)", while those deferred until after the first pass are labeled "(DEFERRED)".

Table I

TIDY CONTROL CARDS
The Default Value is Given First.

Miscellaneous Control Cards

*LAST	Stops all processing.
*STOP	Same as *LAST.
*SKIP	Skips to an END card
*NEWRO	Resets everything to starting values.

Cards to control what is punched.

*CARD	Requests punched output.
*NOCARD	Suppresses punched output.
*COLL	Collect format statements at the end
*NOCOLL	of the routine, or leave them in place.
*COMM	Transmit comments to the output.
*NOCOMM	Delete comment statements.
*NOCONT	Delete CONTINUEs, reroute GOTO-s.
*CONT	Leave GOTO-s and labeled CONTINUEs.

(Table I, continued)

Cards to control what is printed.

*LIST	List original cards.
*NOLIST	Don't list original cards.
*LIST = 2.	List new ("pass 2") cards.
*NOLIST = 2.	Don't list new cards.
*NOREFE	Don't make cross-reference table.
*REFE	Make cross-reference table.

Cards which affect the format of the punched output.

*NOEXEM	Process all statements.
*EXEM	Exempt non-executable statements.
*LEFT=n.	Statement numbers start in column n.
*RIGH=n.	Statement numbers end in column n.
*COLU=n.	FORTRAN starts in column n.
*NOCOLU	First letter of FORTRAN is not moved.

Cards which affect the new statement numbers.

*NOBASE	Same as *BASE=0.
*BASE=n.	Sets "zero-th" statement number.
*STAT=n.	Sets statement number increment.

Cards which affect the serial numbers in columns 73-79.

*SERI	Use serial letters and numbers.
*NOSERI	Leave columns 73-79 blank.
*NOLABE	Use the alphabet for labels (col 73-75.)
*LABE	Take label from card 1 of each routine.
*ROUT	Set the routine counter (see *NOLABEL).
*IDST=n.	Sets the serial number increment.
*IDIN=n.	Same as *ID STEP=n.

Detailed Description of Control Cards

*NOBASE

*BASE = 100.

(DEFERRED)

Although TIDY normally starts the statement numbering from zero and increments the statement numbers uniformly, (e.g. 1, 2, 3,...,etc., or 5, 10, 15,...,etc.), a base number may be added to all statement numbers by the use of the *BASE control statement. In this case, the base is the number given between the "=" sign and the period. For the example given above, TIDY might

assign statement numbers 101,102,103,....
*NOBASE is equivalent to *BASE=0.

*CARDS

*NOCARDS

(DEFERRED)

Normally, TIDY punches the TIDY-processed version of the FORTRAN program. The *NOCARDS control code inhibits such punching. Similarly, *CARDS restores normal punching. If TIDY discovers a serious error in the FORTRAN source deck, card-punching will automatically be suppressed, although TIDY will go on to print a listing of the TIDY-processed faulty deck. The *CARDS control code will override the punching suppression if it appears in the source deck after the card or cards in error.

TIDY may be executed on a "compile-only" basis through the use of the *NOCARDS option. This gives a "quick look" at how TIDY would translate a program, without the cost of punching cards. The same effect may be had in the IBM OS environment by dummying out the file which is normally destined for the punch.

*COLLECT

*NOCOLLECT

(IMMEDIATE)

Normally, TIDY collects all FORMAT statements encountered within each routine and places them at the end of the routine. This collection may be suspended by the *NOCOLLECT control card. Collection is restored by the *COLLECT control card. FORMAT statements bracketed by *NOCOLLECT and *COLLECT control cards will be left in their position in the FORTRAN routine, while others will be collected normally and will appear at the end of the routine.

*COLUMN=n.

*NOCOLUMN

(DEFERRED)

Programming styles differ, especially in how the text is indented. The card *COLUMN=n., where n is a number from 7 to 72, causes TIDY to begin all processed cards in column n. The default value is, of course, 7. Due to the difficulty in processing statements in which blanks could be significant, such as FORMAT statements, this scheme is not carried over to continuation cards, which start in column 7.

The *NO COLUMNS command causes TIDY to retain leading blanks, although imbedded blanks are processed normally. This preserves any indenting that the programmer has used, and is useful when the programmer uses indented lines to highlight the logical flow of the program.

*COMMENTS

*NOCOMMENTS

(IMMEDIATE)

Although TIDY normally processes FORTRAN comment cards and puts them in the updated version, the *NOCOMMENTS control card may be used to inhibit the output of all comment cards. The option is cancelled by the *COMMENTS card. Comment cards to be removed may be bracketed by a *NOCOMMENTS card and a *COMMENTS card, in a manner analogous to the collection of FORMAT cards suggested above.

*NOCONTINUES

*CONTINUES

(IMMEDIATE)

TIDY normally treats a CONTINUE statement as superfluous, unless it ends a DO-loop. These statements are eliminated, and the label, if any, is used for the next executable statement. Similarly, a label on a GOTO statement is superfluous. The label is removed, and any statement which refers to it is rerouted to refer to its target statement.

The *CONTINUES control card disables this feature. Chains of GOTOs are not shortened and all labeled CONTINUE statements are retained, although unlabeled CONTINUEs and CONTINUEs with unused labels are deleted.

The *NOCONTINUES card restores the original treatment of superfluous material.

This processing is especially useful for code that is produced by a pre-processor such as MORTRAN. On the other hand, a particular programming style might require that the CONTINUE cards and GOTOs be left in.

*EXEMPT

*NOEXEMPT

(IMMEDIATE)

Certain programmers like to punch FORTRAN statements, such as COMMON and DIMENSION statements, in special ways so that the names line up in columns, and so forth. Normally, TIDY, in its attempt to eliminate unessential blanks in FORTRAN statements, will destroy such column arrangements. The *EXEMPT control card causes TIDY to exempt all non-executable statements from internal processing, and, hence, from elimination of blanks. The *NOEXEMPT card restores normal processing. If only certain statements in a routine need protection, they may be bracketed by a pair of *EXEMPT and *NOEXEMPT control cards. Note that statements protected from internal processing are also protected from diagnostic checking, and may contain errors that TIDY would normally discover.

*ID STEP = 2.

*ID INCREMENT = 2.

(DEFERRED)

These control statements have identical effects. They set the card sequence increment so that the identification columns (columns 76 through 79) of the output cards will be labeled in multiples of the increment given between the "=" sign and the period. In the above examples, TIDY might label the output cards 2, 4, 6, 8,...,etc. A negative or zero increment is interpreted by TIDY as the default increment of 1. Care must be taken not to assign too large an increment, since the maximum label value is 9999.

*LABEL

*NOLABEL

(DEFERRED)

TIDY inspects columns 73, 74, and 75 of the first card of each routine for possible use in labeling the revised deck. However, TIDY normally does not use these columns. Instead, TIDY uses a unique letter-number combination to identify output cards.

For example, all cards belonging to the first routine of a program will normally be labeled with the letter "A" in column 75. Within this routine, the cards will be sequentially numbered in columns 76 through 79. The next routine will normally be labeled with the letter "B" in column 75, and will be internally numbered. The following routine will be routine "C", and so on. (The *NOSERIAL control card suppresses these labels entirely.)

The *LABEL control card causes TIDY to label the cards of each routine with the contents of columns 73,74, and 75

of the first card of that routine. Note that the label does not come from the *LABEL card itself unless the *LABEL card is the first card of the routine. Ordinarily, the programmer will put the label that he wishes on the first card of his routine (the SUBROUTINE, FUNCTION, or PROGRAM card), and follow that card with the *LABEL control card. Of course, if columns 73-75 of the first card are blank, then the output cards will be blank in these columns and sequentially numbered in columns 76 through 79.

The *NOLABEL control card cancels the labeling option and restores normal alphabetic labeling. TIDY keeps count of the routines being processed, so that, if labeling is restored, the letter of the alphabet corresponding to that routine is used. TIDY has provisions for generating 676 unique one- and two- letter alphabetic labels.

*LAST

(IMMEDIATE)

This control, which is identical in action to *STOP, calls for immediate termination of TIDY, and is used to indicate that the last routine has been processed. Because TIDY uses an input buffering routine that reads one card ahead of the statement being processed, both a *LAST and a *STOP card should follow the last END statement of the input FORTRAN deck. If both cards are not used, the TIDY job may abort because of an input "end-of-file".

Note that *LAST and *STOP should not appear in a routine to be processed since they cause an immediate stop, and therefore will not permit a partially processed routine to finish processing.

In addition to the use of the *STOP and *LAST control cards, TIDY execution may be terminated by the pseudo-FORTRAN command "FINIS". This command is translated by TIDY as *STOP.

*LEFT ADJUST = 2.

(DEFERRED)

In the TIDY-processed deck, statement labels in columns 1-5 normally start in column 1, that is, they are "left-adjusted". To conform to different programming styles, TIDY will optionally start these labels in any column from 1 to 5. The above example would cause TIDY to start all statement labels in column 2, except, of course, those with 5 digits, which must start in column 1. *LEFT with no number given, or an illegal number (not in the range 1-5) is equivalent to *LEFT=1.

See also *RIGHT ADJUST.

*LIST

*NOLIST

(IMMEDIATE)

TIDY normally prints two lists of each routine, one showing the original version and one showing the TIDY-processed version. The listing of the original routine may be suppressed by the *NOLIST card. Only the final version of the TIDY-processed routines will then be listed.

If TIDY issues a diagnostic message while the *NOLIST option is in effect, the card in error and the following card are listed, together with the message.

Use of the *NOLIST option will usually result in a substantial saving in execution time, as well as a 50% saving in paper.

See also *PRINT.

*NEWROUTINE

(DEFERRED)

This control card resets the routine counter so that, if normal alphabetic labeling is done, the current routine being processed will be labeled "A".

The *NEWROUTINE also restores all TIDY flags to their default value, as if TIDY were starting anew. *NEWROUTINE implies the following options: *BASE=0., *CARDS, *COLLECT, *COMMENTS, *NOEXEMPTIONS, *ID INCREMENT=1., *NOLABELS, *LIST, *NOREFERENCES, and *STATEMENT INCREMENT=1. If any of those options are not desired following the execution of a *NEW ROUTINE card, they must be reset by the use of the appropriate control cards.

*REFERENCES

*NOREFERENCES

(DEFERRED)

This card calls for a statement number reference directory, which lists all new statement numbers, the corresponding old statement numbers, and the location of the statements in the old routine.

The *NOREFERENCES statement inhibits the preparation of the directory.

*RIGHT ADJUST = 4.

(DEFERRED)

In the TIDY-processed deck, statement labels in columns 1-5 normally start in column 1, that is, they are "left-adjusted". To conform to different programming styles, TIDY will optionally end them in any column from 1 to 5, that is right-adjust them to any cclumn. The above example would cause TIDY to end all statement labels in column 4, except, of course, those with 5 digits, which fill columns 1-5. *RIGHT with no number, or an illegal number (not in the range 1-5), is equivalent to *RIGHT = 5.

See also *LEFT ADJUST.

*ROUTINE = 26.

(DEFERRED)

This control statement sets the routine counter to the number between the "=" sign and the period. If the normal alphabetic labeling option is on, the current routine will be labeled with the letter(s) corresponding to the number given (e.g. "Z" for the above example). The use of *ROUTINE does not change any other control card options.

*SERIAL

*NOSERIAL

(DEFERRED)

The revised deck produced by TIDY normally has a label and serial number combination punched in columns 73-79, as described under *LABEL above. The *NOSERIAL card causes TIDY to suppress all identification in columns 73-79. The *SERIAL control card cancels this suppression and restores punching of columns 73-79 according to the labeling option currently in effect.

*SKIP

(IMMEDIATE)

This control card causes TIDY to skip processing of the routine in which it appears. Instead, TIDY will simply list the routine until the END card is found. No internal processing is done other than the test for the END statement. The routine being skipped is not punched. The routine counter, however, is incremented.

When the END card is found, the *SKIP option is cancelled and TIDY resumes normal processing with the following card.

*STATEMENT NUMBER INCREMENT = 2.

(DEFERRED)

This control statement causes TIDY to increment statement numbers by the value given between the "=" sign and the period. For the above example, TIDY would assign statement numbers in the sequence 2, 4, 6, 8,.... A negative or zero number in the *STAT control statement results in the default increment of 1.

*STOP

(IMMEDIATE)

This control statement, like *LAST, causes immediate termination of the TIDY job. For further comments, see the description of the *LAST control card.

Section III

An Example

Input

The following input deck was used in a TIDY job, with the control cards shown in Appendix I (p. 20). (This code is the FORTRAN output from a MORTRAN program, and bears the usual five-digit statement numbers and multiple CONTINUE cards.)

```
*LEFT ADJUST =2.
*STATEMENT STEP=10.
*NO SEPIAL
*REFERENCE DIRECTORY
      SUBROUTINE TLOOP
      LOGICAL*1 CARD(80),STRING
      INTEGER INFO(6)
      DATA INFO/6*0/
      COMMON /TOKENC/ INTERP,NSTRNG,MAXSTR,STRING(100)
      MAXSTR=100
10010 CONTINUE
10011 CONTINUE
      CALL TOKRED(INFO,CARD,80)
      IF((INFO(1).LE.0))GOTO 10012
10020 CONTINUE
10021 CONTINUE
      CALL TOKEN(INFO,CARD,  23HX,XMIN,XMAX,FROM,TO,BY;)
      WRITE(6,10030)INFO,INTERP,(STRING(I),I=1,NSTRNG)
10030 FORMAT  (1X,6I4,I4,1X,100A1)
      IF (INFO(4).EQ.2) GOTO 10010
      IF (INFO(4).NE.0) CALL TOKERR(INFO,CARD)
      INFO(4)=0
      IF((INTERP.EQ.1))GOTO 10022
      GOTO 10021
10022 CONTINUE
      GOTO 10011
10012 CONTINUE
      STOP
      STOP
      END
*STOP
*STOP
```

Listing

This page shows the TIDY listing of the original cards. It is nearly the same as the input deck, except for the line numbers and the diagnostic message. Following the listing is the cross-reference directory of statement labels, requested by the *REFERENCES card. The first input card is used for a header on this page, but on subsequent pages the first FORTRAN card will be used.

```
    * T I D Y *          ROUTINE   1    PASS 1  PAGE  1
                                    *LEFT ADJUST =2.
  1 *LEFT ADJUST =2.
  1 *STATEMENT STEP=10.
  1 *NO SERIAL
  1 *REFERENCE DIRECTORY
  1       SUBROUTINE TLOOP
  2       LOGICAL*1 CARD(80),STRING
  3       INTEGER INFO(6)
  4       DATA INFO/6*0/
  5       COMMON /TOKENC/ INTERP,NSTRNG,MAXSTR,STRING(100)
  6       MAXSTR=100
  7 10010 CONTINUE
  8 10011 CONTINUE
  9       CALL TOKRED(INFO,CARD,80)
 10       IF((INFO(1).LE.0))GOTO 10012
 11 10020 CONTINUE
 12 10021 CONTINUE
 13       CALL TOKEN(INFO,CARD,   23HX,XMIN,XMAX,FROM,TO,BY;)
 14       WRITE(6,10030)INFO,INTERP,(STRING(I),I=1,NSTRNG)
 15 10030 FORMAT  (1X,6I4,I4,1X,100A1)
 16       IF (INFO(4).EQ.2) GOTO 10010
 17       IF (INFO(4).NE.0) CALL TOKERR(INFO,CARD)
 18       INFO(4)=0
 19       IF((INTERP.EQ.1))GOTO 10022
 20       GOTO 10021
 21 10022 CONTINUE
 22       GOTO 10011
 23 10012 CONTINUE
 24       STOP
 25       STOP
****(  1) ***THE ABOVE STATEMENT CANNOT BE REACHED
 26       END

                STATEMENT NUMBER DIRECTORY
  NEW    OLD    LOC              OLD    LOC       NEW
   10 = 10010,(   7).          10010,(   7) =     10.
   10 = 10011,(   8).          10011,(   8) =     10.
   20 = 10021,(  12).          10012,(  23) =     30.
   10 = 10022,(  21).          10021,(  12) =     20.
   30 = 10012,(  23).          10022,(  21) =     10.
   40 = 10030,(  15).          10030,(  15) =     40.
 OLD STATEMENT NUMBERS NOT APPEARING IN THIS DIRECTORY
 WERE NOT REFERENCED AND HENCE ARE DELETED.
```

Output

This page shows the TIDY listing of the processed code. The punched output is the same, without the page headings.

```
* T I D Y *          ROUTINE   1    PASS 2  PAGE  1
                                  SUBROUTINE TLOOP
      SUBROUTINE TLOOP
      LOGICAL*1 CARD(80),STRING
      INTEGER INFO(6)
      DATA INFO/6*0/
      COMMON /TOKENC/ INTERP,NSTRNG,MAXSTR,STRING(100)
      MAXSTR=100
 10   CALL TOKRED (INFO,CARD,80)
      IF ((INFO(1).LE.0)) GO TO 30
 20   CALL TOKEN (INFO,CARD,23HX,XMIN,XMAX,FROM,TO,BY;)
      WRITE (6,40) INFO,INTERP,(STRING(I),I=1,NSTRNG)
      IF (INFO(4).EQ.2) GO TO 10
      IF (INFO(4).NE.0) CALL TOKERR (INFO,CARD)
      INFO(4)=0
      IF ((INTERP.EQ.1)) GO TO 10
      GO TO 20
 30   STOP
      STOP
C
 40   FORMAT (1X,6I4,I4,1X,100A1)
      END
```

Appendix I

JCL

I have compiled TIDY with the FORTRAN (H Extended) compiler, using OPT=2. The compiled version is in WYL.CG.RBC.LOADMODS(TIDY). The deck necessary to run a TIDY job is

```
//ABCTIDY  JOB  ABC$XY,999,CLASS=E
//    EXEC LOADGO,REGION=200K,PARM='EP=MAIN'
//SYSLIN DD DSN=WYL.CG.RBC.LOADMODS(TIDY),DISP=SHR
//FT01F001 DD UNIT=SYSDA,SPACE=(TRK,(5,5)),
//      DCB=(RECFM=VBS,LRECL=1030,BLKSIZE=3094)
//FT02F001 DD UNIT=SYSDA,SPACE=(TRK,(5,5)),
//      DCB=(RECFM=VBS,LRECL=1030,BLKSIZE=3094)
X     //FT08F001 DD SYSOUT=B,
X     //      DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
//SYSIN DD *
      Intermixed
         FORTRAN
            and
               TIDY
                  Control
                     Cards
                        Go
                           Here...
/*
```

If you want to set up a file with the card output, and use it later from WYLBUR rather than getting punched cards that you will have to feed back in again, you will have to define FORTRAN unit 8 as a disk file. The following three lines could replace the two above marked "X".

```
//FT08F001 DD DSN=WYL.XY.ABC.TIDYCARD,DISP=(NEW,KEEP),
//    UNIT=DISK,VOL=SER=SCR001,SPACE=(TRK,(5,5),RLSE),
//    DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
```