SHARE PROGRAM LIBRARY AGENCY

PROGRAM NUMBER

067019

University of Miami

1365 MEMORIAL DRIVE - CORAL GABLES, FLORIDA

(305) - 284-6257

CONTRIBUTED PROGRAM LIBRARY SUBMITTAL FORM
(for IBM S/360, 1130 and 1800)

SHARE Program Library Agency
Triangle Universities Computation C...
P. O. Box 12076
Research Triangle Park, N. C. 27709

This form should be completed and submitted with the program package to PID at the address shown above. Standards and instructions for submitting programs are in your *User Group Reference Manual* or the *Contributed Program Submittal Standards Manual* available from PID.

(1) Program Order Number *(to be filled in by PID)* 360D-06.7.019

(2) System Type *(machine)* S/360

(3) Search Key . . . . . FORTRAN SUBROUTINES GENERATE KWOC AND LEFT JUSTIFIED KWIC OUTPUT

(4) Programming Language . . . . . 360 OS FL4G

(5) Author's Name and Address . . . . Harold P. Sieglaff
3610 W Northview
Phoenix
Arizona 85021

(6) Direct Inquiries to Name and Address *(if different than Author)*

(7) Title of Program . . . . KWADE keyword as a dictionary entry

(8) Submitter's User Group Affiliation Code and Installation Code . . . . . N

(9) Submitter's Own Program Identification and Suffix *(optional)* . . . . . LJKW

(10) Primary Subject Code 06.7

(11) Secondary Subject Codes 06.6

(12) Operating or Monitor System Required . . . . . SEE ABSTRACT

(13) New or Revision Code *(if revision, show prior Program Order Number in item 1)* . . . . . N

(14) Year Completed 68

(15) Date of Submittal 12 26 68

(16) Documentation *(number of original pages submitted)* 17

(17) Abstract *(should contain sufficient information for a reader to determine the value of the program). Listed on the reverse side of this form are subjects which may serve as a guide for a descriptive abstract.*

CONTRIBUTED PROGRAM LIBRARY SUBMITTAL FORM

Subject Guide

a. Purpose
b. Programming Language used
c. Version and modification level or release number of IBM Programming System used, or program order number for non-IBM authored program used
d. Field of application
e. Type of routine (main program, subroutine, etc.)
f. Specific description of machine requirements
g. Engineering Changes (EC) level of equipment (if pertinent)

ABSTRACT These Fortran subroutines generate KWOC and left justified KWIC output from a character string supplied by the user.

The subroutines were tested using OS Fortran Language 4 GLevel, OS Fortran Language 4 H Level, and OS Version 13 on a S/360 model 50.

The output record can vary in size as follows:

OUTPUT record = INPUT record + maximum size keyword + 2

OUTPUT record must be less than or equal to 256 characters.

The subroutines can be used to process titles and/or keywords of articles of a journal.

4K bytes of core are required in addition to means of getting information into and from core (e.g. card reader, CRT, disk, drum, printer, or tape).

The subroutines should work on any S/360 machine which has OS and Fortran G or H.

(Please attach additional pages if necessary) . . . . . Total pages attached _____

Permission to Publish

"I hereby give anyone permission to reprint, reproduce, and distribute this program to anyone else."

(18) Signature of Submitter and Date Harold P. Zieglaff 26 DEC 68

(19) Signature of Installation Addressee _____

T4SF

TABLE OF CONTENTS

OPTIONAL CHANGES:

DECK KEY

Deck #1 Source deck of LJKWIC, sequence 10 thru 770 in cc 73-80; 91 cards.

Deck #2 Source deck of KWADE, sequence 10 thru 770 in cc 73-80; 86 cards.

Deck #3 Source deck of BINSER, sequence 00 thru 560 in cc 73-80; 59 cards.

Deck #4 Source deck of INSERT, sequence 00 thru 170 in cc 73-80; 18 cards.

Deck #5 Source deck of SETSKP, sequence 00 thru 1080 in cc 73-80; 112 cards.

NAME: LJKWIC (use subroutine KWADE to generate unrotated output)

ENTRY POINTS:

LJKWIC initializes processing

KWIC processes an input character string

DESCRIPTION:

LJKWIC produces left justified KWIC (keyword in context) output from a character string supplied by the calling program and writes the output on an I/O unit specified by the user.

In the preparation of a left justified KWIC output data set (e.g. tape) four steps are required:

Step 1 Input of the character string (In this write-up the input consists of card images from the card reader, disk, or tape.)

Step 2 Creation of unsorted left justified KWIC output from the input supplied in Step 1

Step 3 Sorting of the output produced in Step 2

Step 4 Printing of the output produced in Step 3

Steps 1,3,4 can be done in any way desired by the user. This write-up is of a subroutine which performs Step 2, although an example is given of Steps 1,3,4.

A word is defined as a string of A-Z 0-9 characters ending with (but not including) a non A-Z 0-9 character. A keyword is any word not in the exclusion dictionary.

The subroutine does the following:

1. compares a given word in a user supplied string against a user supplied alphanumerically ordered exclusion dictionary using a binary search. On the average a decision is made as to whether the word is a keyword in $\log_2 N$ comparisons. (N = no. of entries in the exclusion dictionary)

2. allows the user considerable freedom as to the size of the output records:

OUTPUT record = INPUT record + maximum keyword size + 2

OUTPUT record $\leq$ 256 characters

Pictorially:

INPUT string (record) = HEAD KEYWORD TAIL

OUTPUT string (record) = KEYWORD[1] KEYWORD TAIL ,* HEAD

3. allows the user to specify what part of the input string he wants scanned

Assume that the user has chosen 15 as the maximum number of characters to be used as a keyword. (b = blank)

INPUT record = THE EFFECTS OF THALIDIMIDE ON PREGNANT WOMEN

OUTPUT record 1 = EFFECTSbbbbbbbb EFFECTS OF THALIDIMIDE ON PREGNANT WOMEN,*THE

OUTPUT record 2 = THALIDIMIDEbbbb THALIDIMIDE ON PREGNANT WOMEN,*THE EFFECTS OF

OUTPUT record 3 = PREGNANTbbbbbbb PREGNANT WOMEN,*THE EFFECTS OF THALIDIMIDE ON

OUTPUT record 4 = WOMENbbbbbbbbbb WOMEN,*THE EFFECTS OF THALIDIMIDE ON PREGNANT

1. KEYWORD followed by (maximum keyword size - no. of characters in this keyword) blanks

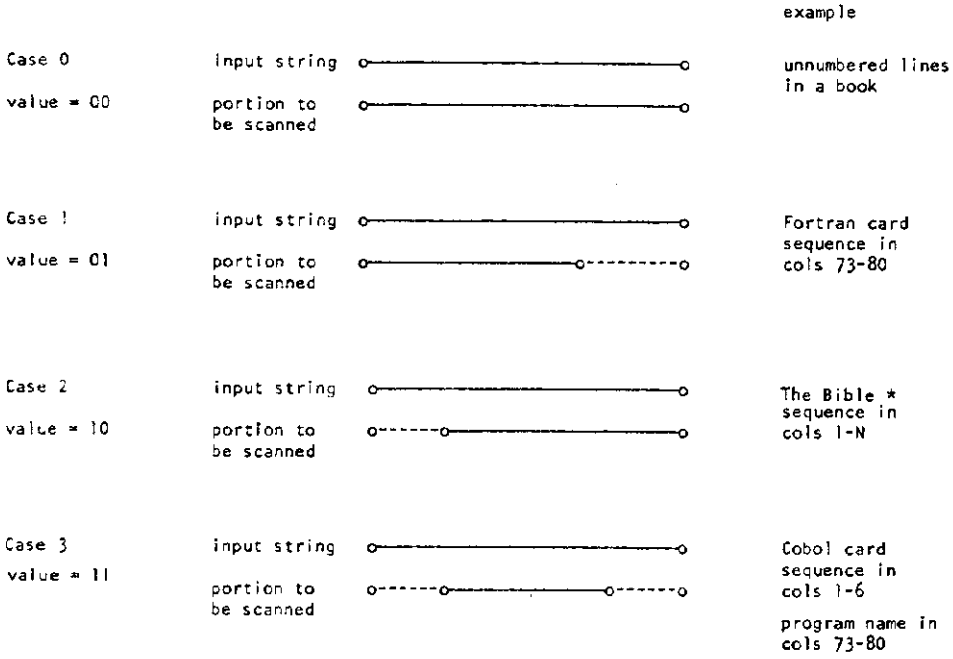
In processing a contiguous portion of a string of characters there are mathematically four possible portions to select from:

Let 0 = end point is used

1 = end point is not used

_____ = part of string to be scanned for keywords

-------- = part of string not to be scanned for keywords



Although the entire input record is output only the specified part is scanned for keywords.

* Verses in the Bible (e.g. Genesis 1:1 In the beginning ...)

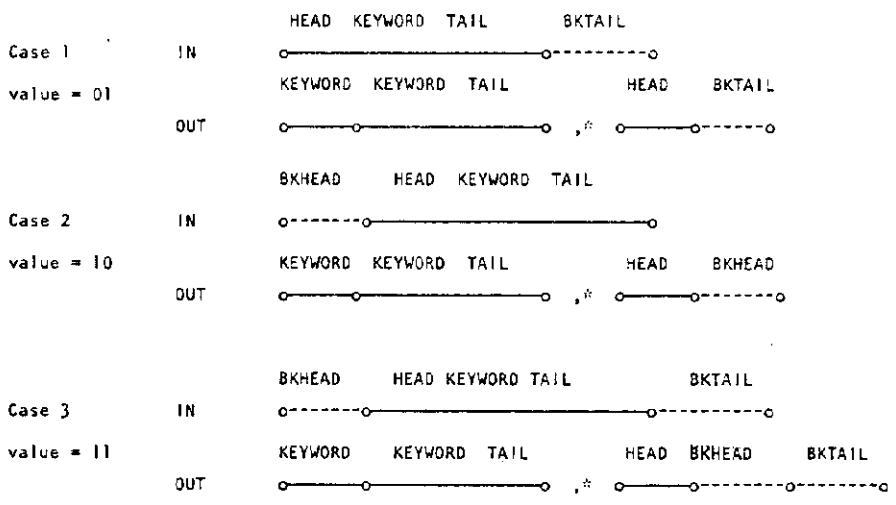
For Case 0 input records the following discussion does not apply.

Let BKHEAD = bookkeeping preceding the portion of the input string to be scanned for keywords

BKTAIL = bookkeeping following the portion of the input string to be scanned for keywords

IN = input string

OUT = output produced by the subroutine LJKWIC



AUTHOR: Harold P. Sieglaff

LANGUAGE: FORTRAN IV

MEMORY REQUIREMENTS: 2292 bytes

AVAILABILITY: LJKWIC is available as a source deck from the author & from the CONTRIBUTED PROGRAM LIBRARY of IBM.

CALLING SEQUENCES:

```
CALL LJKWIC (INBYTE, NSTART, NEND, MAXKEY, NAUT, LENGTH)
CALL KWIC   (NUMKEY, IN(1), OUT(1), TABLE(1,1), NENTRY, LSIZE)
```

PARAMETER DESCRIPTIONS:

INBYTE = no. of bytes (characters) in the input records

NSTART = no. of the position in the input record where the scan for keywords is to start

NEND = no. of the position in the input record where the scan for keywords is to end

MAXKEY = maximum no. of characters to be used as a keyword

NAUT = no. of the I/O unit which is to receive the output

LENGTH = a location into which LJKWIC will place the size of the output records
= INBYTE + MAXKEY + 2 characters

NUMKEY = a location into which KWIC will place the no. of keywords found in the input record (string)

IN(1) = the 1st location of a LOGICAL * 1 array which contains the input record (string)

OUT(1) = the 1st location of a LOGICAL * 1 array from which output records are to be written by KWIC

TABLE(1,1) = the 1st location of a LOGICAL * 1 array which contains the alphanumerically ordered exclusion dictionary

NENTRY = no. of entries in the TABLE (exclusion dictionary)

LSIZE = no. of bytes (characters) used for each entry in TABLE

RESTRICTIONS:

1) OUTPUT record = INPUT record + maximum keyword size + 2

OUTPUT record $\leq$ 256 characters

If OUTPUT record $>$ 256 characters no output is written.

2) MAXKEY $\leq$ LSIZE

3) The entries in the exclusion dictionary must be alphanumerically ordered. Alphanumeric ordering is as follows (.LT. = less than) :

blank .LT. A .LT. B .LT. C ... .LT. Z .LT. 0 .LT. 1 ... .LT. 9

There are 256 possible characters; they are ordered by the collating sequence.

4) All entries in the exclusion dictionary must use the same no. of characters. If necessary use blanks to fill out positions of an entry in the exclusion dictionary. (e.g. if the size of an entry in the exclusion dictionary = 10 use:

```
OFbbbbbbbb      for OF
ONbbbbbbbb      for ON
OTHERbbbbb      for OTHER
```

ERROR INDICATIONS: none

OTHER SUBROUTINES REQUIRED: BINSER, INSERT, SETSKP which are available as source decks from the author & from the CONTRIBUTED PROGRAM LIBRARY of IBM.

The subroutine could be modified to do the following:

Entries the user wishes to add to the exclusion dictionary at or during execution could be placed in a 2nd table & processed sequentially. (The binary search already used by the subroutine could be used to process one entry at a time.)

```
CALL KWIC (NMKEY, IN(1), OUT(1), TABLE(1,1), NENTRY, LSIZE, TABLE2(1,1), KENTRY, KSIZE)
```

If KENTRY .GE. 1 process the KENTRY entries of TABLE2 sequentially.

If KENTRY .LE. 0 do not process TABLE2

Example of its use:

Step 1 Input of character strings
Step 2 LJKWIC or KWADE processing of the input strings

```
      LOGICAL * 1   TABLE(30,500) , CARD(80) , OUT(256) , IN(256)
      EQUIVALENCE  (ITEST , CARD(1))
C***       read end of data indicator, input & output unit nos.,
C          maximum keyword size, no. of cards per input,
C          starting & ending cols for scan, no. of chars in input record
      READ (5,1) IENDAT, INUNIT, NAUT, MAXKEY, NUMCRD,
     X           NSTART, NEND, INBYTE
    1 FORMAT (A4, 7 I 5)
      IF (MAXKEY .LE. 0)  MAXKEY = 30
      IF (NSTART .LE. 0)  NSTART = 1
C***       read exclusion dictionary entries
      DO 10 NCARD = 1,500
      READ (5,2) CARD
    2 FORMAT (80 A1)
      IF (IENDAT .EQ. ITEST) GO TO 100
C***       move exclusion word to TABLE
      CALL INSERT (CARD(1), CARD(MAXKEY), TABLE(1,NCARD))
      NENTRY = NCARD
   10 CONTINUE
C***       initialize  LJKWIC  (rotation of input string)
  100 IF (NUMCRD .GT. 1) GO TO 200
      NUMCRD = 1
      IF (INBYTE .LE. 0) INBYTE = 80
      IF (NEND   .LE. 0) NEND   = 72
      GO TO 3
  200 IF (NUMCRD .NE. 2) GO TO 300
      INBYTE = 152
      NEND   = 144
      GO TO 3
  300 NUMCRD = 3
      INBYTE = 224
      NEND   = 216
```



```
    3 CALL LJKWIC (INBYTE, NSTART, NEND, MAXKEY, NAUT, LENGTH)
      NUMREC = 0
C***       read 1, 2, 3 cards
   40 DO 123  N = 1,NUMCRD
      READ (INUNIT, 2, END=30, ERR=124) CARD
      IF (ITEST .EQ. IENDAT) GO TO 30
C***       move card to  IN  buffer
  124 LSTART = 1 + 72 * (N - 1)
      CALL INSERT (CARD(1), CARD(80), IN(LSTART))
  123 CONTINUE
      CALL KWIC (NUMKEY, IN(1), OUT(1), TABLE(1,1), NENTRY, 30)
      NUMREC = NUMREC + NUMKEY
      GO TO 40
   30 WRITE (6,31) LENGTH, NENTRY, NUMREC
   31 FORMAT (1H1, 'LENGTH =', I5, '  NENTRY =', I5, '  NUMREC =', I5)
      STOP
      END
```

To use KWADE (no rotation of the input string) use this statement 3:

```
    3 CALL KWADE (INBYTE, NSTART, NEND, MAXKEY, NAUT, LENGTH)
```

The following JCL was used:

```
//GO.FT08F001  DD  DCB=(BLKSIZE=10200,LRECL=102,RECFM=FB),DISP=NEW,        X
//                 LABEL=(,NL),UNIT=2400,VOLUME=SER=0438
//GO.SYSIN  DD  *
*/+-     5    8   20    1    7   72   80   IN/OUT,MAXKEY,CARDS,START,END,INBYTE
     ***exclusion dictionary entries***
*/+-            END OF EXCLUSION DICTIONARY ENTRIES
     ***data cards to be processed***
*/+-            END OF DATA CARDS TO BE LJKWIC ED  OR  KWADE ED
/*
```

Step 3 Sorting of the output produced in Steps 1 & 2

```
//SORTKWIC  JOB  620000,'SORT KWIC SIEG',MSGLEVEL=1                      10
//STEP1     EXEC SCRATCH1                                                20
//STEP2     EXEC PROC=SORT,PARM='CORE=100000'                            30
//SORTLIB   DD   DSNAME=SYS1.SORTLIB,DISP=SHR                            40
//SORTIN    DD   DCB=(BLKSIZE=10200,LRECL=102,RECFM=FB),DISP=OLD,      X  50
//               LABEL=(,BLP),UNIT=2400,VOLUME=SER=0438                  60
//SORTOUT   DD   DCB=(BLKSIZE=10200,LRECL=102,RECFM=FB),DISP=NEW,      X  70
//               LABEL=(,BLP),UNIT=2400,VOLUME=SER=0424                  80
//SORTWK01  DD   VOLUME=REF=SCRATCH1,SPACE=(CYL,50,,CONTIG)              90
//SORTWK02  DD   VOLUME=REF=SCRATCH1,SPACE=(CYL,50,,CONTIG)             100
//SORTWK03  DD   VOLUME=REF=SCRATCH1,SPACE=(CYL,50,,CONTIG)             110
//SYSIN     DD   *                                                      120
       SORT  FIELDS=(1,20,A),FORMAT=CH                                  130
       END                                                              140
/*
```


Step 4 Printing of the output produced in Step 3

```
      LOGICAL * 1   BUFFER(256) , TITLE(70)
C***        Read input unit no.  &  no. of bytes per record  & title
      READ (5,1)  INUNIT, NBYTES, TITLE
    1 FORMAT (2 I 5, 70 A1)
C***        Output TITLE at top of page
      WRITE (6,2) TITLE
    2 FORMAT (1H1, 30X, 70A1)
C***        Read a record
    7 READ (INUNIT, 3, END=4, ERR=5) (BUFFER(N) , N = 1,NBYTES)
    3 FORMAT (255 A1, A1)
C***        Print the record
    5 WRITE (6,6) (BUFFER(N) , N = 1,NBYTES)
    6 FORMAT (1HO, 132A1/, 1HO, 8X, 124A1)
      GO TO 7
    4 STOP
      END
```

The following JCL was used:

```
//GO.FT08F001  DD  DCB=(BLKSIZE=10200,LRECL=102,RECFM=FB),DISP=OLD,      X
//                 LABEL=(,NL),UNIT=2400,VOLUME=SER=0424
//GO.SYSIN  DD  *
    5  102 KWADE OUT: 6 DEC 68 ALPHABETICALLY ORDERED LIST OF SRL MANUALS
/*
```

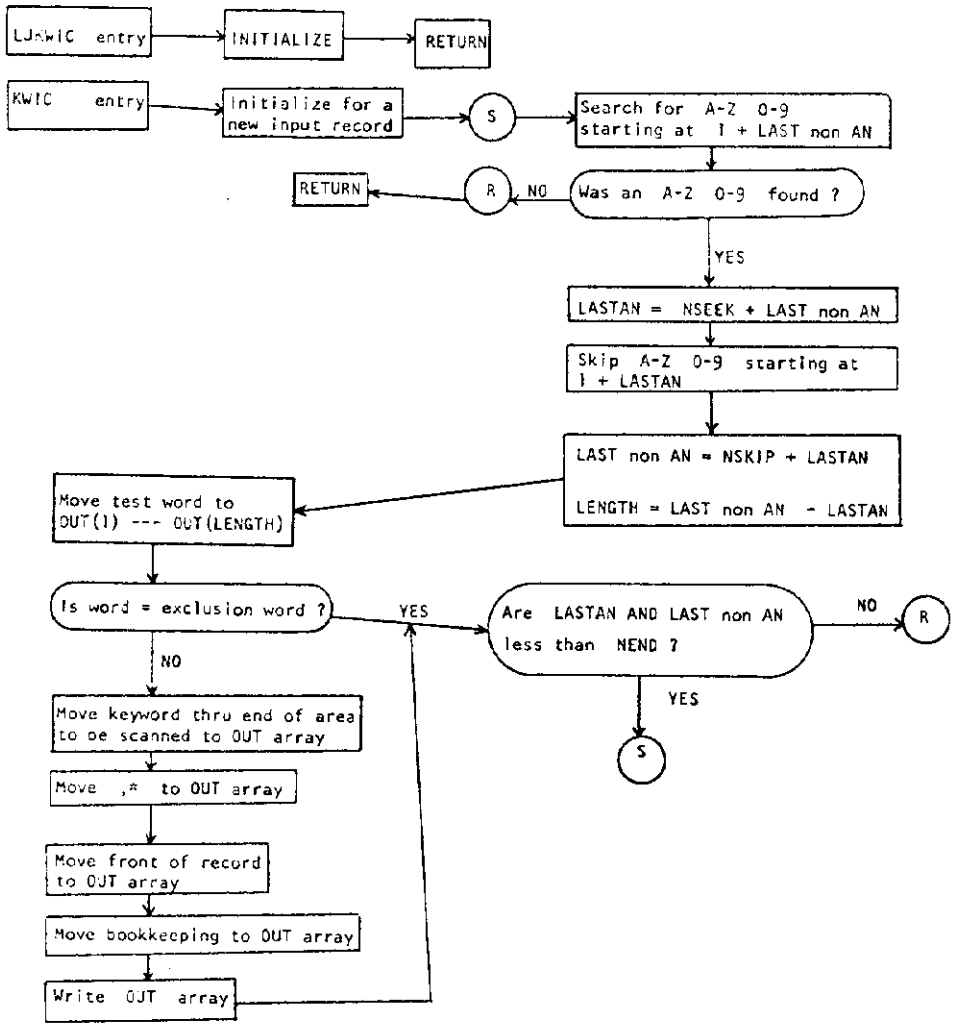
If the programs are in a library called XXXXXLIB
the following JCL could be used for STEPs 1,2,3,4

```
//KWADE   JOB  620000,'LJKWIC / KWADE SIEG',MSGLEVEL=1                    10
//JOBLIB  DD   DSNAME=XXXXXLIB,DISP=SHR                                  20
//STEP1   EXEC PGM=IKWADE                                                30
//SYSPRINT  DD SYSOUT=A                                                  40
//FT06F001  DD SYSOUT=A                                                  50
//FT08F001  DD DCB=(BLKSIZE=10200,LRECL=102,RECFM=FB),DISP=(,PASS),      60
//             LABEL=(,BLP),UNIT=2400,VOLUME=SER=1234                    70
//FT05F001  DD *                                                         80
*/+-     5     8    20     1     1   72   80      IN/OUT,MAXKEY,CARDS,    90
                                                  START, END, INBYTE
    *** exclusion dictionary words (in cols 1-20 one word per card)
        alphanumerically ordered ***
*/+-                    END OF EXCLUSION DICTIONARY ENTRIES
    *** cards to be KWADEd  cols 1-72 = data
                            cols 73-80= bookkeeping    ***
*/+-                    END OF DATA CARDS
/*                                                                      100
```



```
//STEP2  EXEC  SCRATCH1                    clear the disk pack          110
//STEP3  EXEC  PROC=SORT,PARM='CORE=100000'       sort STEP1 output     120
//SORTLIB  DD  DSNAME=SYS1.SORTLIB,DISP=SHR                              130
//SORTIN   DD  DCB=(BLKSIZE=10200,LRECL=102,RECFM=FB),DISP=OLD,       X  140
//             LABEL=(,BLP),UNIT=2400,VOLUME=SER=1234                    150
//SORTOUT  DD  DCB=(BLKSIZE=10200,LRECL=102,RECFM=FB),DISP=(,PASS),   X  160
//             LABEL=(,BLP),UNIT=2400,VOLUME=SER=1235                    170
//SORTWK01 DD  VOLUME=REF=SCRATCH1,SPACE=(CYL,50,,CONTIG)                180
//SORTWK02 DD  VOLUME=REF=SCRATCH1,SPACE=(CYL,50,,CONTIG)                190
//SORTWK03 DD  VOLUME=REF=SCRATCH1,SPACE=(CYL,50,,CONTIG)                200
//SYSIN    DD  *                                                         210
      SORT  FIELDS=(1,20,A),FORMAT=CH                                    220
      END                                                                230
/*                                                                       240
//STEP4  EXEC  PGM=PKWADE                        print sorted output     250
//SYSPRINT DD  SYSOUT=A                                                  260
//FT06F001 DD  SYSOUT=A                                                  270
//FT08F001 DD  DCB=(BLKSIZE=10200,LRECL=102,RECFM=FB),DISP=OLD,       X  280
//             LABEL=(,BLP),UNIT=2400,VOLUME=SER=1235                    290
//FT05F001 DD  *                                                         300
   8  102   KWADE OUT DATA  SRL MANUALS KEYWORDS OF                      310
/*                                                                       320
```

FLOWCHART OF LJKWIC



Testing & Timing

These subroutines have been used to generate:

(1) left justified KWIC output of the keywords of the basic SRL manuals.

(2) left justified KWIC output of the table of contents of the HASP write-up to produce an index.

(3) cross referenced KWADE (unrotated) output from Cobol decks.

(4) cross referenced KWADE (unrotated) output from Fortran decks.

The output is generated at a rate of from 600-1000 lines of unsorted output per minute.

Timing is for a S/360 model 50.

Expression of Appreciation

I wish to thank the following persons who supplied the indicated ideas before or during the writing & testing of LJKWIC and KWADE:

person	idea supplied
Rick Bousley	basic idea of the binary search
Jim Ford *	use of the TRT instruction for searches
Fay Gray	placing of bookkeeping information at the end of the output record
Bill Orth	use of instructions MVI and MVC to fill a 256 position table
John Rourke	request for a character filter for use while scanning a character string
John Winegar	efficient JCL for sorting
Earl Woods	a basic understanding of the JCL needed for sorting

* Providing 2 or more entry points to a subroutine is a problem on the S/360. I have used his solution to this problem.

(1) To allow the user to specify the characters which are to be considered alphanumeric the following changes can be made:

```
      SUBROUTINE  LJKWIC  (... ,  LETTER, NUMUSE)                    10
      LOGICAL * 1              ... ,  LETTER(1)                       20

      IF (NUMUSE .LE. 0)  CALL SETSKP (ALPHNU(1), 36)                160
      IF (NUMUSE .LE. 0)  CALL SETSEK (ALPHNU(1), 36)          2ADD  170
      IF (NUMUSE .GE. 1)  CALL SETSKP (LETTER(1), NUMUSE)            171
      IF (NUMUSE .GE. 1)  CALL SETSEK (LETTER(1), NUMUSE)            172
```

Corresponding changes can be made to subroutine KWADE

(2) To remove the restriction which limits the output record to 256 or fewer characters make these changes:

```
C***      IF THIS IS THE IST CALL PUT SPACES IN THE OUTPUT BUFFER  4ADD 270

   61  FORMAT (5 (255A1))                                               720
```

Corresponding changes can be made to subroutine KWADE

The following restrictions would still apply:

BKHEAD, BKTAIL, and MAXKEY would each have to be less than or equal to 256 characters in length.

NEND - NSTART .LE. 255 (The portion to be scanned for keywords must have 256 or fewer characters.)

The output record would have to be 1026 or fewer characters.

(3) To bypass unnecessary core to core moves in subroutine KWADE:

```
C***    OUT(1)  ---  OUT(LENGTH)  =  A KEYWORD                         550
C***    HAS STRING  IN  BEEN MOVED TO STRING  OUT  ?                   560
   40  IF (NUMKEY .GE. 1)  GO TO 62                            2SUB  570
C***    IN(START) --- IN(NEND)  TO  OUT  ARRAY                         600
       CALL INSERT (IN(NSTART), IN(NEND), OUT(KEYSIZ+1))               610

   62  WRITE (KAUT,61) (OUT(N) , N = 1,NAUCHR)                         710
```


LJKWIC OR KWADE input where RECFM=F or RECFM=FB with LRECL .LE. 999 and INBYTE .LE. LRECL

```
      LOGICAL*1  TABLE(30,500), CARD(80), IN(999), OUT(999), ANCHAR(80)  10
      EQUIVALENCE (ITEST, CARD(1))                                       20
C***      read end of data indicator, input & output unit nos.,          30
C         maximum keyword size, starting & ending cols for keyword scan, 40
C         no. of chars in the input records,  A-Z  0-9  indicator        50
      READ (5,1)  IENDAT, INUNIT, NAUT, MAXKEY, NSTART, NEND, INBYTE,  1ADD60
     X            NUMUSE                                                61
    1 FORMAT (A4, 1X, 7 I 5)                                          1ADD70
      IF (NUMUSE .GE. 1)  READ (5,2)  ANCHAR                           71
C***      read exclusion words                                           80
      DO 10  NCARD = 1,500                                             90
      READ (5,2)  CARD                                                100
    2 FORMAT (80 A1)                                                  110
      IF (IENDAT .EQ. ITEST) GO TO 3                                  120
C***      move exclusion word to TABLE                                  130
      CALL INSERT (CARD(1), CARD(MAXKEY), TABLE(1,NCARD))             140
      NENTRY = NCARD                                                  150
   10 CONTINUE                                                        160
C***      initialize  LJKWIC  or  KWADE                                170
    3 CALL LJKWIC (INBYTE, NSTART, NEND, MAXKEY, NAUT, LENGTH,    1ADD 180
     X             ANCHAR(1), NUMUSE)                                 181
      NUMREC = 0                                                      190
```

```
C***      read an input record                                              200
     40   READ (INUNIT, 4, END=30, ERR=5) (IN(N) ,N = 1,INBYTE)            210
      4   FORMAT (4 (255A1))                                               220
      5   CALL KWIC (NUMKEY, IN(1), OUT(1), TABLE(1,1), NENTRY, 30)        230
          NUMREC = NUMREC + NUMKEY                                         240
          GO TO 40                                                         250
     30   WRITE (6,31) LENGTH, NENTRY, NUMREC                              260
     31   FORMAT (1H1, 'LENGTH =', I5, '  NENTRY =', I5, '  NUMREC=', I5)  270
          STOP                                                             280
          END                                                              290
```

To execute KWADE (no rotation of the input string) use this statement 3:

```
      3   CALL KWADE (INBYTE, NSTART, NEND, MAXKEY, NAUT, LENGTH,      IADD 180
     X            ANCHAR(1), NUMUSE)                                        181
```

If NUMUSE .LE. 0 use A-Z 0-9 for alphanumeric characters.

If NUMUSE .GE. 1 use user supplied characters for alphanumeric characters. Read a card into ANCHAR(1) --- ANCHAR(80) and use the 1st NUMUSE characters as alphanumeric.

(note: If NUMUSE .GE. 1 there is an EXTRA card in the input stream. It precedes the exclusion dictionary cards and the characters on it may be in any order.)

This program is called VARPRT

Program to allow the user to determine the format of the printer output:

```
      LOGICAL * 1  BUFFER(999) , TITLE(70) , FORMAT(80)                    10
C***      read input unit no., no. of bytes per record, title              20
      READ (5,1)  INUNIT,  NBYTES,  TITLE                                 30
    1 FORMAT (2 I 5, 70A1)                                                 40
C***      read FORMAT statement to be used for writing the output          50
      READ (5,10)  FORMAT                                                  60
   10 FORMAT (80A1)                                                        70
C***      output  TITLE  at top of the page                                80
      WRITE (6,2)  TITLE                                                   90
    2 FORMAT (1H1, 30X, 70A1)                                             100
C***      read a record                                                   110
    7 READ (INUNIT, 3, END=4, ERR=5) (BUFFER(N), N = 1,NBYTES)            120
    3 FORMAT (4 (255A1))                                                  130
C***      print the record                                                140
    5 WRITE (6, FORMAT) (BUFFER(N) , N = 1,NBYTES)                        150
      GO TO 7                                                             160
    4 STOP                                                                170
      END                                                                 180
```

The following JCL was used:

```
//FT08F001  DD  DCB=(BLKSIZE=7150,LRECL=286,RECFM=FB),DISP=OLD,          X
//            LABEL=(,NL),UNIT=2400,VOLUME=SER=0424
//FT05F001  DD  *
   8  286   4 CARD OUTPUT  23 APRIL 1969
(1HO, 84A1/,  2(24X, 60A1/),  24X, 82A1)  4 LINES 84, 84, 84, 106
/*
```

For printing purposes a special symbol (e.g. #) could be used to indicate the beginning & end of each line in the bookkeeping portion(s) of the input strings. The print program could then easily break up the bookkeeping portion of the output record into the user specified print lines.

Consider the # symbols to be numbered 1 through N. The odd numbered # symbols indicate the beginning of a line & the even numbered # symbols indicate the end of a line. Characters appearing between even and odd numbered # symbols are ignored.
(note: The 2 symbols ## are a complete line and could be used to print a blank line between items of a bibliography.)

Subroutine SETSKP could be used to scan the bookkeeping portion of the output record for # For a write-up of SETSKP see:

FORTRAN CHARACTER STRING PACKAGE 3600-06.6.003
Catalog of Programs
File No. 5360-20
Form C20-1619

The following discussion applies to LJKWIC only. If the portion of the input string to be scanned for keywords sometimes ends before position NEND one could use a symbol (e.g. #) as indicated:

```
  IN  BKHEAD  HEAD   KEYWORD  TAIL  #blanks   BKTAIL

 OUT  KEYWORD   KEYWORD  TAIL #blanks  ,* HEAD  BKHEAD  BKTAIL

PRINT  KEYWORD   KEYWORD TAIL ,* HEAD  blanks     BKHEAD  BKTAIL
```

One's print program takes the OUT record (produced by LJKWIC from IN record) converts it to the PRINT record by reversing the positions of ,* HEAD and #blanks and then prints it.

is replaced with a blank

Print program for LJKWIC:

```
C***     Change  KEYWORD  KEYWORD  TAIL  #BLANKS  ,* HEAD   BOOKKEEPING      10
C        to      KEYWORD  KEYWORD  TAIL   ,* HEAD BLANKS    BOOKKEEPING      20
C        # stands for any delimiter                                           30
      LOGICAL * 1   BUFFER(999), TITLE(80) , FORMAT(80) , LIMTER , BLANK      40
      DATA BLANK /' '/                                                        50
C***     read and print the TITLE                                             60
      READ (5,1)  TITLE                                                       70
    1 FORMAT (80A1)                                                           80
      WRITE (6,2)  TITLE                                                      90
    2 FORMAT ( 1H1, 30X, 80A1)                                               100
C***     read the FORMAT statement to be used in printing the output         110
      READ (5,1) FORMAT                                                      120
C***     read input unit no., size of input records, keyword size,           130
C        no. of chars which were scanned for keywords, delimiter,            140
C        no. of the first and last characters to be printed                  150
C        refer to LJKWIC writeup:  LENGTH = 1 + NEND - NSTART               160
      READ (5,3) INUNIT, INBYTE, KEYSIZ, LENGTH, LIMTER, KFIRST, KLAST       170
    3 FORMAT (4I5, 4X, A1, 2I5)                                              180
C***     compute the no. of the last position preceding                      190
C        the bookkeeping portion                                             200
      LAST   = KEYSIZ + LENGTH + 2                                           210
      NBOOK  = LAST + 1                                                      220
C***     compute the no. of chars to search for the delimiter                230
      N2SEEK = LENGTH + 2                                                    240
C***     prepare to search for the delimiter                                 250
      CALL SETSEK (LIMTER, 1)                                                260
C***     prepare to skip blanks                                              270
      CALL SETSKP (BLANK,  1)                                                280
```

```
C***      read a record                                                    290
    8 READ (INUNIT, 4, END=5, ERR=6) (BUFFER(N) , N = 1,INBYTE)           300
    4 FORMAT (4(255A1))                                                   310
C***      search for the delimiter, write out record if not found         320
    6 CALL SEEK (BUFFER(KEYSIZ+1), N2SEEK, NUMPOS)                        330
      IF (NUMPOS .EQ. 0)  GO TO 7                                         340
C***      move a blank to the delimiter position                          350
      LIMPOS = KEYSIZ + NUMPOS                                            360
      BUFFER(LIMPOS) = BLANK                                              370
C***      search for  ,* HEAD  by skipping blanks                         380
C         find the first non-blank character                              390
      N2SKIP = LAST - LIMPOS                                              400
      IF (N2SKIP .LE. 0) GO TO 7                                          410
      CALL SKIP (BUFFER(LIMPOS+1), N2SKIP, NWHERE)                        420
      IF (NWHERE .EQ. 0) GO TO 7                                          430
      NCOMMA = LIMPOS + NWHERE                                            440
C***      compute the no. of blanks between the delimiter and  ,* HEAD    450
      NBLANK = NCOMMA - LIMPOS - 1                                        460
      IF (NBLANK .LE. 0) GO TO 7                                          470
C***      shift  ,* HEAD  to  1 + delimiter position                      480
      CALL INSERT (BUFFER(NCOMMA), BUFFER(LAST), BUFFER(LIMPOS+1))        490
C***      compute no. of position following  ,* HEAD  & move blanks there 500
      NSTART = NBOOK - NBLANK                                             510
      BUFFER(NSTART) = BLANK                                              520
      IF (NBLANK .LE. 1) GO TO 7                                          530
      CALL INSERT (BUFFER(NSTART), BUFFER(LAST-1), BUFFER(NSTART+1))      540
C***      writeout BUFFER using FORMAT                                    550
    7 WRITE (6,FORMAT) (BUFFER(N) , N = KFIRST, KLAST)                    560
      GO TO 8                                                             570
    5 STOP                                                                580
      END                                                                 590
```


Definition of an exclusion dictionary:

an exclusion dictionary is a table of alphabetically ordered words none of which is to be used as a keyword (e.g. AN, BY, FOR, ... ON, OTHER, ...)

Special symbols one could choose to be used as alphanumeric:

symbol	use
.	to tell the difference between .5 and 5
	to tie together the chemical formula $CUSO_4 \cdot nH_2O$
,	to tie together the chemical formula 1,4-benzenedicarboxaldehyde
-	to tie together the above formula
	to tie together the words: X-RAY, A-V ANEURSYM, A-V NODE
/	to indicate an author /NORBERT/WIENER , /BERTRAND /RUSSELL, /MARK /TWAIN

The 40 characters to be considered alphanumeric would be:

ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789.,-/

To allow for imbedded comments in the input string:
if the AN character starting a word is a special one (e.g. @) skip all characters between occurrences of it by seeking it.
HEAD @ skip all characters between occurrences of the special character @ TAIL

This character could be the last of the AN string supplied to KWADE and used or not used depending on the value of NUMUSE, which determines the no. of characters which are to be considered alphanumeric.

A new parameter could be added to the argument list of KWADE to indicate whether or not the "imbedded comments" feature is to be used.

When the special character is found the following sequence of operations could be done:

1) SETSEK would be reset to seek one character (the special one).
```
CALL SETSEK (LETTER(NUMUSE), 1)
```
2) Seek the character (skipping all other characters).

3) Reset SETSEK to seek any of the NUMUSE characters.
```
CALL SETSEK (LETTER(1), NUMUSE)
```
4) Replace the special character with a blank wherever it occurs.

note: Odd numbered occurrences of the special character start an imbedded comment
even numbered occurrences end an imbedded comment.

Sketch of print program for KWIN


```
      LINPER =  ? maximum no. of lines per record  (read from a card)
      LINPAG =  ? maximum no. of lines to be printed per page (read from a card)

(1)   read in and print the  TITLE,  NUMLIN = 1
(2)   read the no. of characters in  lines 1-N  and the FORMAT statements to be used
      READ (5,1)  LINSIZ                                  in the printing
    1 FORMAT (16 I 5)
(3)   read a record output by  KWIN
      READ (INUNIT, 10, END= ?,  ERR= 40)  (BUFFER(N) , N = 1, INBYTE)
   10 FORMAT (5 (255A1))
(4)   if this record would overflow the page, advance to a new page
   40 NUMLIN = NUMLIN + LINPER
      IF (NUMLIN .LT. LINPAG)  GO TO 20
      WRITE (6, 30)
   30 FORMAT (1H1)
      NUMLIN = LINPER + 1
(5)   process 16 or fewer lines
   20 LEND   = 0
      NUMLIN = NUMLIN - LINPER
      DO 2  N = 1,16
      IF (LINSIZ(N) .LE. 0)  GO TO 3
      LSTART = LEND + 1
      LEND   = LSTART + LINSIZ(N) - 1
      IF (N .NE. 1) GO TO 4
      WRITE (6, FORM1) (BUFFER(K) , K = LSTART , LEND)
      NUMLIN = NUMLIN + 2
      GO TO 2
If BUFFER(LSTART) thru BUFFER(LEND)  =  BLANK(1) thru BLANK(LINSIZ(N))  go to 2
use subroutine  LEG  or a  DO  loop
    4 WRITE (6, FORM2) (BUFFER(K) , K = LSTART , LEND)
      NUMLIN = NUMLIN + 1
    2 CONTINUE
    3 next statement
(6)   FORM1 = (1HO, 132A1)      FORM2 = (1H ,  nnX, 132 A1)
                                   nn = no. of characters used for keywords = MAXKEY
```

Program to construct an input record for KWADE : This program is called KWIN

```
      LOGICAL * 1  TABLE(30,500) , CARD(80) , IN(999) , OUT(999) , ANCHAR(80),
     X             IN2(999) , SPACES(256) , LIMTER(4) , BLANK
      DATA BLANK /' '/ , IMLIMT /0/
      EQUIVALENCE (ITEST , CARD(1)) , (IMLIMT , LIMTER(1))
C***      read end of data indicator, input & output unit nos.,
C         maximum keyword size, starting and ending cols for scan,
C         no. of chars in input record, A-Z  0-9  indicator,
C         maximum no. of input records per logical record, delimiter
      READ (5,1) IENDAT, INUNIT, NAUT, MAXKEY, NSTART, NEND, INBYTE,
     X           NUMUSE, MAXREC, LIMTER(4)
    1 FORMAT (A4, 1X, 8 I 5, 4X, A1)
      IF (NUMUSE .GE. 1) READ (5,2) ANCHAR
      N2SCAN = NEND - NSTART + 1
      LTAIL  = NSTART + N2SCAN * MAXREC
C***      compute end position for the scan by KWADE
      KEND   = LTAIL - 1
C***      compute no. of chars in the records to be input to KWADE
      IMSIZE = NSTART - 1  +  MAXREC * N2SCAN  +  INBYTE  -  NEND
C***      0 = no. of ending position of last record
      LASTEN = 0
C***      put  BLANK  in  SPACES  and  IN2
      DO 11  N = 1,256
   11 SPACES(N) = BLANK
      LAST   = INBYTE * MAXREC
      DO 12  N = 1,LAST
   12 IN2(N) = BLANK
```



```
C***      read exclusion words
      DO 10  NCARD = 1,500
      READ (5,2) CARD
    2 FORMAT (80 A1)
      IF (IENDAT .EQ. ITEST) GO TO 3
C***      move exclusion word to TABLE
      CALL INSERT (CARD(1), CARD(MAXKEY), TABLE(1,NCARD))
      NENTRY = NCARD
   10 CONTINUE
C***      initialize  KWADE
    3 CALL KWADE ( IMSIZE, NSTART, KEND, MAXKEY, NAUT, LENGTH,
     X             ANCHAR(1), NUMUSE)
      NUMREC = 0
C***      read the input record(s) of a logical record
   40 DO 50  NREC = 1,MAXREC
      READ (INUNIT, 4, END=30, ERR=5) (IN(N) , N = 1,INBYTE)
    4 FORMAT (4 (255A1))
C***      compute start & end positions for storing record
    5 LBEGIN = NSTART  +  (NREC - 1) * N2SCAN
      LEND   = LBEGIN  +  N2SCAN  -  1
C***      indicate whether the delimiter has been found
      IFEND  = -1
      IF (NREC .EQ. MAXREC) IFEND = 0
```

```
C***      search for the delimiter                                          500
      ITEST  = 0                                                           510
      DO  52  LCHAR = NSTART , NEND                                        520
      CARD(4) = IN(LCHAR)                                                  530
C***      ITEST  =  0 0 0  IN(LCHAR)     IMLIMT  =  0 0 0  DELIMITER       540
      IF (ITEST .NE. IMLIMT) GO TO 52                                      550
      NUMPOS = LCHAR                                                       560
      GO TO 53                                                             570
 52   CONTINUE                                                             580
C***      delimiter was not found                                          590
      GO TO 51                                                             600
C***      delimiter was found replace it with a BLANK                      610
 53   IFEND  = 0                                                           620
      IN(NUMPOS)  =  BLANK                                                 630
C***      if this record is smaller than the last one fill the             640
C         difference with BLANKs                                           650
      LDIF   = LASTEN - LEND                                               660
      IF (LDIF .GT. 0) CALL INSERT (SPACES(1),SPACES(LDIF), IN2(LEND+1))   670
C***      move bookkeeping and/or area to scan to  IN2                     680
 51   IF (NSTART .GT. 1  .AND.  NREC .EQ. 1)  CALL INSERT (IN(1),          690
     X                                        IN(NSTART-1), IN2(1))       700
      IF (NEND .LT. INBYTE  .AND.  IFEND .EQ. 0)  CALL INSERT              710
     X   (IN(NEND+1), IN(INBYTE), IN2(LTAIL))                              720
      CALL INSERT ( IN(NSTART), IN(NEND), IN2(LBEGIN))                     730
```



```
C***      was the delimiter found or is  NREC = MAXREC ?                   740
      IF (IFEND .EQ. -1)  GO TO 50                                         750
      LASTEN = LEND                                                        760
      CALL KWIC ( NUMKEY, IN2(1), OUT(1), TABLE(1,1), NENTRY, 30)          770
      NUMREC = NUMREC + NUMKEY                                             780
      GO TO 40                                                             790
 50   CONTINUE                                                             800
 30   WRITE (6,31) LENGTH, NENTRY, NUMREC                                  810
 31   FORMAT (1H1, 'LENGTH =', I5, '  NENTRY =', I5, '  NUMREC =', I5)     820
      STOP                                                                 830
      END                                                                  840
```

SUMMARY OF LJKWIC / KWADE PROGRAMS

main pgm	description
VARPRT	input : F or FB records where LRECL .LE. 999 and a FORMAT statement to be used in printing the records.
	output: The records printed according to the FORMAT statement read in at execution time.
PTLJKWIC	input : LJKWIC output of the form: keyword keyword tail ≠blanks ,* head bookkeeping and a FORMAT statement to be used in printing all or part of the records.
	output: The above input record is changed to the following form: keyword keyword tail ,* head blanks bookkeeping and printed according to the FORMAT statement read in at execution time.

SUMMARY OF LJKWIC / KWADE PROGRAMS

input pgm +	subroutine =	main pgm	description
KWADE	KWADE	KWADE	input : F or FB records where LRECL .LE. 999 and INBYTE .LE. LRECL
			output: Keyword as a dictionary entry (KWOC)
KWIN	KWADE	KWIN	input : One or more records of the form: BKHEAD NSTART --- NEND BKTAIL (if any) (if any) from which is produced: BKHEAD NSTART --- NEND ... NSTART --- NEND BKTAIL record 1 record 1 record N record N
			output: Keyword as a dictionary entry (KWOC)
LJWKIC	LJKWIC	LJKWIC	input : F or FB records where LRECL .LE. 999 and INBYTE .LE. LRECL
			output: Left justified keyword in context (KWIC)

cards needed to run KWIN which allows for 1 or more records (e.g. cards)
to be used as a set to generate a logical record for input to subroutine KWADE:

```
//xxxx      JOB  acct no., 'KWIN SIEG',MSGLEVEL=1                                  10
//JOBLIB    DD    DISP=SHR,DSNAME=TESTLIB                                         20
//STEP1     EXEC PGM=KWIN                                                        30
//FT08F001  DD DCB=(BLKSIZE=3458,LRECL=182,RECFM=FB),DISP=(,PASS),    X          40
//             DSNAME=JW02201,SPACE=(CYL,(20,5)),VOLUME=REF=SCRATCH1              50
//FT06F001  DD SYSOUT=A                                                         60
//FT05F001  DD *                                                                70
*/+-     5     8     20     1     80     80     37     2     #                   80 *
ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789.                                           90 **
    ***exclusion dictionary words in alphanumeric (alphabetic) order
      one word per card in cols 1-20                                      ***
*/+-          end of exclusion dictionary entries                              100
    *** cards to be KWIN ed which may be in any order   ***
/*            end of cards to be KWIN ed                                       110
//STEP2    EXEC SORT0,PARM='CORE=100000'                                       120
//SORTIN   DD    DISP=(OLD,DELETE),DSNAME=JW02201                              130
//SORTOUT DD     DCB=(BLKSIZE=3458,LRECL=182,RECFM=FB),DISP=(,KEEP),  X        140
//             DSNAME=JW02202,SPACE=(CYL,(20,5)),VOLUME=REF=SCRATCH1            150
//SORTWK01 DD    SPACE=(CYL,30 ,CONTIG),VOLUME=REF=SCRATCH1                     160
//SORTWK02 DD    SPACE=(CYL,30,,CONTIG),VOLUME=REF=SCRATCH1                     170
//SORTWK03 DD    SPACE=(CYL,30,,CONTIG),VOLUME=REF=SCRATCH1                     180
//SYSIN     DD   *                                                            190
          SORT  FIELDS=(1,20,A),FORMAT=CH                                     200
          END                                                                 210
/*                                                                            220
```



```
//STEP3     EXEC PGM=VARPRT                                                   230
//FT08F001  DD DISP=(OLD,DELETE),DSNAME=JW02202,VOLUME=REF=SCRATCH1           240
//FT06F001  DD SYSOUT=A                                                       250
//FT05F001  DD *                                                              260
    8  182     cols 11 thru 80 = title of the output                          270 ***
(1H0, 100A1/, 1H0, 20X, 82A1)  write out 182 chars 20A1 80A1 / 20X 82A1       280
/*                                                                            290
```

* cols 6-10 = no. of the input unit
11-15 = no. of the output unit
16-20 = maximum no. of characters to be used as a keyword = MAXKEY
21-25 = no. of the 1st position to be scanned for keywords = NSTART
26-30 = no. of the last position to be scanned for keywords = NEND
31-35 = no. of characters in each record on the input unit
36-40 = no. of characters to be used as alphanumeric characters = NUMUSE
41-45 = maximum no. of records (e.g. cards) to be used to generate each input record to subroutine KWADE
= MAXREC
50 = the delimiter (this character as one of the characters in positions NSTART thru NEND indicates end of a set.)
If it occurs on a record preceding the MAXREC th record of a set, the record input to KWADE is filled out with blanks.

The delimiter can but need not occur in the MAXREC th record of a set.

** cols 1 - NUMUSE of this card are used as alphanumeric characters

*** cols 1 - 5 = no. of the unit containing the sorted output
6 - 10 = no. of characters in the records on the unit
= MAXKEY + MAXREC * (NEND - NSTART + 1) + BKHEAD + BKTAIL + 2

KWIN does the following:

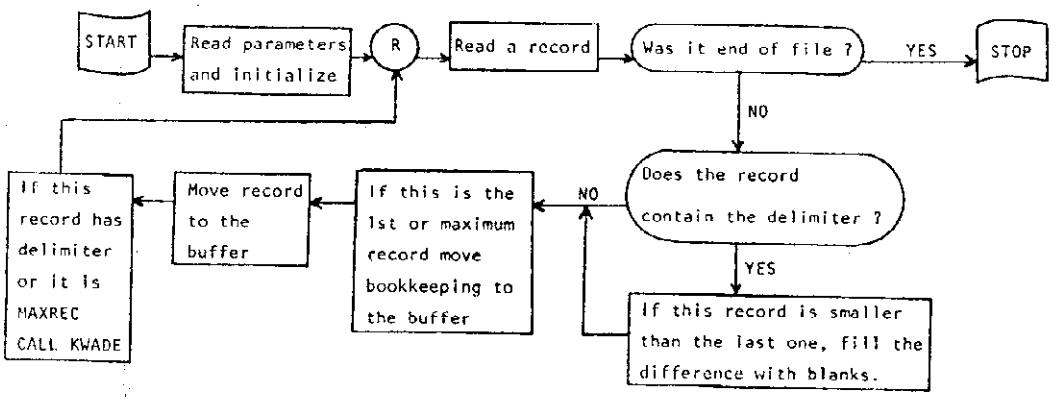
where N .GE. 1 from N input records of the form:

BKHEAD NSTART --- NEND BKTAIL

produces 1 record of the form:

BKHEAD NSTART --- NEND ... NSTART --- NEND BKTAIL
record 1 record 1 record N record N

and inputs it to KWADE.




```
(NEND - NSTART + 1) * MAXREC  .LE. 256     The portion to be scanned for keywords  .LE. 256
                      NSTART  .LE. 257     BKHEAD (if any)  .LE. 256
          INBYTE  -   NEND    .LE. 256     BKTAIL (if any)  .LE. 256
                      MAXKEY  .LE. 256     Keyword must be  .LE. 256
```