

DOCUMENT RESUME

ED 108 615

IR 002 131

AUTHOR Foote, Thomas
TITLE Weekly Log Record Sort (WLSORT).
INSTITUTION Southwest Regional Laboratory for Educational
Research and Development, Los Alamitos, Calif.
REPORT NO SWRL-TN-5-72-14
PUB DATE 21 Mar 72
NOTE 20p.
EDRS PRICE MF-\$0.76 HC-\$1.58 PLUS POSTAGE
DESCRIPTORS Communication Skills; *Computer Programs; *Data
Processing; Electronic Data Processing; Flow Charts;
Information Processing; Information Storage;
*Management Systems; Recordkeeping; *Student Records;
Systems Development
IDENTIFIERS *Computer Software Documentation; FORTRAN V; Sort
Routines

ABSTRACT

Computer routines to sort the weekly log records submitted by teachers participating in the Southwest Regional Laboratory's communications skills monitoring program are described. Written in Univac FORTRAN V, Weekly Log Record Sort (WLSORT) sorts log records on magnetic tape to enable subsequent computer programs to interpret the input data by district, school, and classroom. This document is intended to serve as the software documentation for the programs. Included are a program description, data format specifications, program constraints and limitations, and operating instructions. Program flowcharts, program listings, and sample data forms are also provided. (DGC)

* Documents acquired by ERIC include many informal unpublished *
* materials not available from other sources. ERIC makes every effort *
* to obtain the best copy available. nevertheless, items of marginal *
* reproducibility are often encountered and this affects the quality *
* of the microfiche and hardcopy reproductions ERIC makes available *
* via the ERIC Document Reproduction Service (EDRS). EDRS is not *
* responsible for the quality of the original document. Reproductions *
* supplied by EDRS are the best that can be made from the original. *



SOUTHWEST REGIONAL LABORATORY
TECHNICAL NOTE

DATE: March 21, 1972

NO: TN 5-72-14

TITLE: WEEKLY LOG RECORD SORT (WLSORT)

AUTHOR: Tom Foote

ABSTRACT

WLSORT sorts CS-1 weekly log records located on magnetic tape to enable the new weekly log processor program CS-1-WLSR1, (TN 5-72-15) to interpret all data in terms of classes rather than groups.

U.S. DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY.

IR 008131

WEEKLY LOG RECORD SORT (WLSORT)

1.0 - PROGRAM IDENTIFICATION

WLSORT*

2.0 - OBJECTIVE

Weekly Log records written on magnetic tape in order of district, school, teacher, class, group, month, and day, must be re-sorted in order of district, school, teacher, class, month, day, and group (i.e., shift of the last three fields only) to enable the program CS-1-WLSR1 to interpret all data in terms of classes rather than groups.

3.0 - PROGRAM DESCRIPTION

3.1 - Program Logic

WLSORT reads one weekly log record at a time from magnetic tape, calculates its record identification value (i.e., the value that will be sorted on) according to the month, day, and group read for each record, and assigns a corresponding index value (i.e., the value that will point to that record for output) according to the order in which each record is read. Exit from this loop is triggered by either (1) a device error, (2) a transmission abort, (3) a class containing more than 240 records, (4) an end-of-file read on the input tape, or (5) a record containing a new class identification number. Cases 4 and 5 are considered to be the normal exits, indicating that all records for a class have been properly read and are waiting to be sorted. After exiting the read-a-record loop, and before calling SUBROUTINE SORT, the record containing the new class identification number is stored and labeled as the first record of the next class (whose following records are yet to be read). Ignoring this latest record (which belongs to the next class) and supplied with the last class' array of record ID's and corresponding index values, SUBROUTINE SORT rearranges the record ID's in ascending order (using a bubble sort) and concurrently moves their corresponding index values.

The first value returned in the rearranged index array then directs the output loop to write onto tape, the record with the smallest record ID; the second index value triggers writing of the record with the next smallest record ID, and so on until all records for that class have been written. If a class once sorted on an output tape is not the last class on the input tape, the program returns to continue reading the next class. The program

*WLSORT is an adaptation of CS-1-Weekly Log Sort (TN 5-71-15).

terminates once the read-a-class/sort-a-class cycle is completed for all classes.

3.2 - Variables

All variables in this program are of integer type. The value of the subscript KOUNT corresponds to a selected record within the current class and all arrays subscripted by KOUNT are dimensioned to 240.

- IALPHA (KOUNT, L), INTEGER ALPHA. This array (dimension 240 by 40) contains all the ALPHA data read from one weekly log record.
- ICLASS (KOUNT), INTEGER CLASS. This two-digit value identifies the class to which a given weekly log record belongs.
- IDAY (KOUNT), INTEGER DAY. This two-digit value identifies the day (range 1 to 31) on which the weekly log sheet was dated.
- IDST (KOUNT), INTEGER DISTRICT, SCHOOL, AND TEACHER. This six-digit value identifies the district, school, and teacher to which a particular log record belongs.
- IGROUP (KOUNT), INTEGER GROUP. This single-digit value identifies the group to which a particular weekly log record belongs.
- IMDG (KOUNT), INTEGER MONTH, DAY, AND GROUP. This array of record identification values is computed for each record by concatenating the values IMONTH, IDAY, and IGROUP, e.g., IMONTH = 10, IDAY = 30, IGROUP = 2 yields IMDG = 10302. Once computed for each record within a class, these record ID's are passed to SUBROUTINE SORT and rearranged in ascending order.
- IMONTH (KOUNT), INTEGER MONTH. This two-digit value identifies the month (July = 1, August = 2 . . . June = 12) in which data on a given sheet was recorded.
- INIT, INITIAL VALUE. As the initial value of the main read loop INIT is set to 1 for the first record of the first class read. For subsequent classes, however, INIT is set to 2 to account for the new class record which marked the end of the old class' read loop (i.e., that record which waits in array position 1 of the new class array about to be read).
- KEY (KOUNT), KEY ARRAY. This index array is stored with each counter value of the read loop's DO variable KOUNT, i.e., integers 1, 2, 3, . . . NRECS, originally in that order. SUBROUTINE SORT then rearranges this array by allowing each of the array values to follow their corresponding values of the record ID array

(IMDG) as the IMDG array is sorted. The first value in the rearranged index array KEY then directs the output loop to write the record with the smallest IMDG index value, while the second KEY value triggers writing of the record with the next smallest IMDG index value, and so on.

KOUNT, KOUNTER VALUE. This DO variable for the read loop counts the records being read for a class. In addition to serving as a subscript for each record, each KOUNT value is immediately stored into the KEY array as described in the "KEY (KOUNT)" definition above.

KOUNT2, 2nd KOUNTER VALUE. To avoid alteration of a DO variable outside its respective loop, KOUNT is renamed KOUNT2 upon exit of the read loop in order to compute LESONE. LESONE is the argument passed to SUBROUTINE SORT, which indicates the total number of records to be sorted.

LESONE, LESS ONE. This value equals the number of records read, minus one. By purposely ignoring the last record read, i.e., the record with the discrepant class number, this subtraction yields the exact count of records to be sorted for the preceding class and is thereby used as the third argument passed to SUBROUTINE SORT.

LOGREC(L), WEEKLY LOG RECORD. As the array containing all alpha and numeric data on a weekly log record, this serves as a major argument to be passed to the read/write subroutine NTRAN.

LSTAT, STATUS VALUE. This value indicates the status of NTRAN. The LSTAT status codes are as follows for read and write operations:

- 1 indicates transmission not complete
- 2 indicates end-of-tape (indicates end-of-file in write operations)
- 3 indicates device error
- 4 indicates transmission aborted

NO MORE, NO MORE. This flag is either set to 1 or 0 to indicate whether "no more" classes remain to be sorted.

NO SUB, Number of the SUBscript. This subscript value identifies the weekly log record currently being printed. Since the compiler will not permit the use of nested subscripts e.g., IMONTH(KEY(J)), the current KEY(J) value is restored as NOSUB, which in turn is used as the subscript for writing the elements of the newly sorted record.

4.0 - SUBROUTINES AND FUNCTIONS

- 4.1 - SUBROUTINE SORT (SORTON, FOLLOW, NRECS) sorts the array SORTON (using an ascending bubble sort) while rearranging the array FOLLOW. In turn FOLLOW is employed to fetch and write records in their sorted order.

Argument Description

FOLLOW(NREC), FOLLOW the SORTON elements. This integer array enters the subroutine, containing numbers 1, 2, 3, . . . NRECS, in that order. As out-of-order record ID's are shifted into order, their corresponding elements in the FOLLOW array are also shifted. FOLLOW is finally returned with its elements rearranged, and is used by the main program to fetch and write the records in their new sorted order.

SORTON(NREC), SORT ON this array. The record identification numbers within this array are examined with respect to their order (i.e., relative to their immediate neighbors) within this array so as to trigger sort operations necessary to rearrange them in ascending order.

NRECS, NUMBER OF RECORDS. As the third argument in the subroutine list, this integer value indicates the number of record ID's to be sorted.

- 4.2 - SUBROUTINE NTRAN (UNIT, READ/WRITE CODE, BLOCK LENGTH, BLOCK NAME, STATUS VALUE). This library routine acts in conjunction with subroutines ENCODE and DECODE to read and write formatted records with lengths greater than those handled by the standard FORTRAN READ/WRITE statements.

5.0 - DATA SPECIFICATIONS

5.1 - Input Formats

Each record has been preprocessed from the raw weekly log scanner data, and appears on the input tape in order of district, school, teacher, class, group, month, and day. The record format then corresponds to the optical scan sheet format (see Appendix A), although irrelevant blanks have been deleted. As the following read sequence shows, each record contains: 1) IALPHA, the array of 238 alpha character responses; 2) IDST, the 6 digit integer identifying the district, school, and teacher; 3) ICLASS, the 2 digit integer identifying the class; 4) IGROUP, the 1 digit integer identifying the group; 5) IMONTH and IDAY, the 2 digit integers identifying the month and day respectively; and 6) IUNIT, the 10 digit

integer indicating the units to which the record pertains. These values are read in according to the following format:

```

+ READ (31,200) (IALPHA(KOUNT,L),L=1,40), IDST(KOUNT),
+ ICLASS(KOUNT), (GROUP(KOUNT), IMONTH(KOUNT), IDAY(KOUNT),
+ IUNIT(KOUNT)
200 FORMAT(22A6/17A6,A4,I6,I2,I1,I2,I2,I10,3X)

```

An example of records conforming to this format appears in Appendix B.

5.2 - Output Formats

Each record will be written on the output tape in order of district, school, teacher, class, month, day, and group. The contents of each record will be rearranged as IGROUP is shifted as indicated below:

```

+ WRITE (31,500) (IALPHA(NOSUB,K),K=1,40), IDST(NOSUB),
+ ICLASS(NOSUB), IMONTH(NOSUB), IDAY(NOSUB), IGROUP(NOSUB),
+ IUNIT(NOSUB)
500 FORMAT(22A6/17A6,A4,I6,I2,I2,I2,I1,I10,3X)

```

An example of records conforming to this format appears in Appendix C.

6.0 - PROGRAM CONSTRAINTS AND LIMITATIONS

6.1 - Programming Language

Univac 1108 FORTRAN V

6.2 - Vendor

University Computing Company

6.3 - Storage Requirements

7210 octal words

6.4 - Hardware Configuration

Univac 1108 (EXEC 2), card reader, 2 magnetic tape units and printer

6.5 - Program Parameters

Number of records per class should not exceed 240 unless array dimensions and loop limits are expanded beforehand.

6.6 - Error Messages

If the number of records per class exceeds 240, the program prints the following message before terminating all processing:
 ERROR . . . NUMBER OF RECORDS FOR THIS CLASS EXCEEDS DO
 LOOP LIMIT OF 240.

If an end-of-tape marker is encountered in the write sequence, the program prints the following message before terminating all processing:
 WRITE ERROR I.E., . . .
 ERRORS IN TAPE OR TRANSMISSION ON RECORD ID NUMBER (DSTCMD) =
 01 02 01 01 1 4 1 (i.e., the identification number of the record in question)

7.0 - OPERATING INSTRUCTIONS

At UCC, the program was run with the following control card configuration:

```
@ RUN,W FOOTE,LS3512,3,150
@ MSG DELIVER JOB TO SWRL
@RAKEX ASG H=1343 RINGIN
@RAKEX ASG F=1035 NORING
@ FOR DECK1,DECK1
```

Main Program (see listing, section 9.0)

```
@ FOR DECK2,DECK2
```

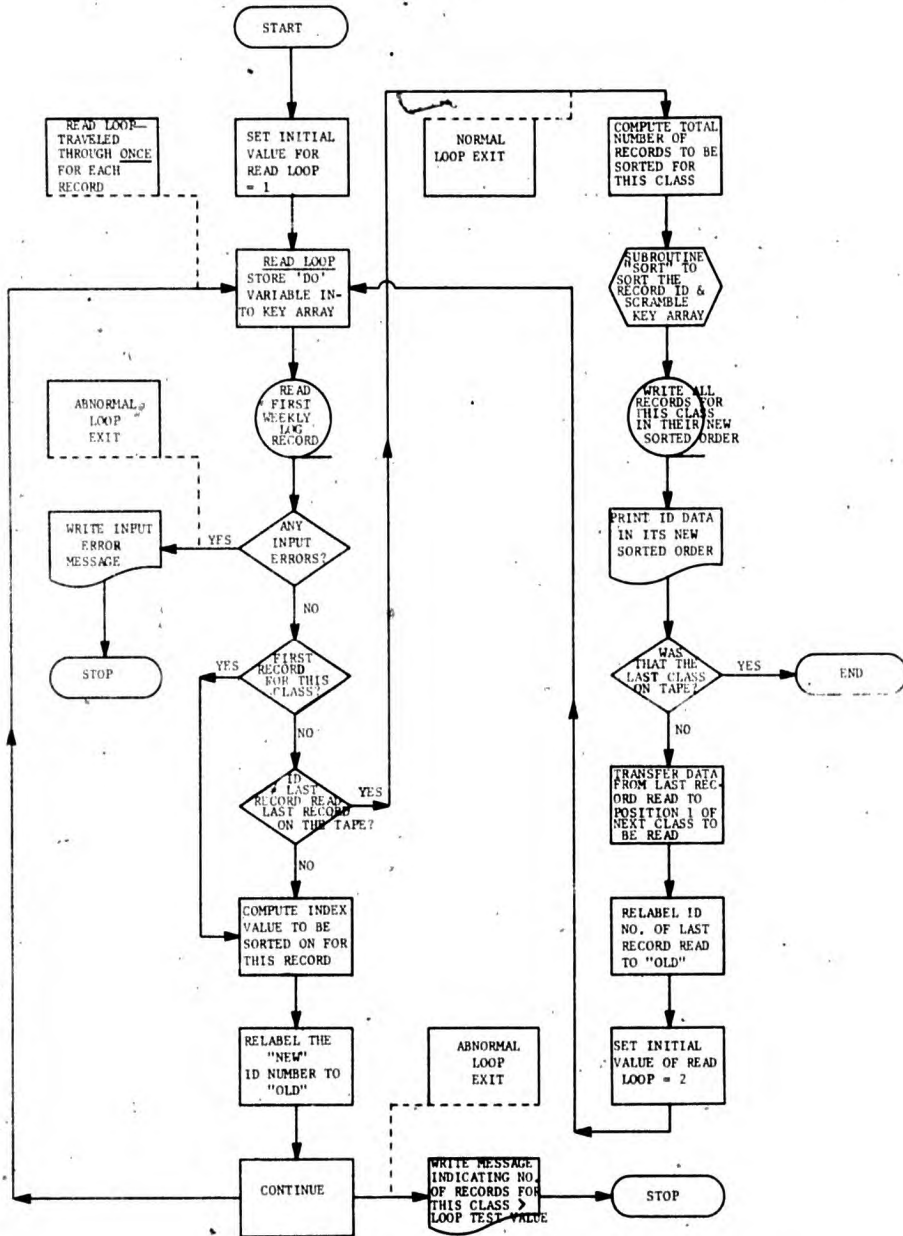
SUBROUTINE SORT

```
@ XQT DECK1
@ XQT TUTIL (TUTIL is a write-to-tape routine, local to UCC)
REWIND H
PRINT H 20 RECS
REWINT F
REWINT H
```




FLOWCHART LAYOUT FORM

PROGRAMMER: Tom Foote DATE:
 PROGRAM ID: WLSORT PAGE 1 OF 1



9.0 - PROGRAM LISTING

```

C*****WLSORT SORTS WEEKLY LOG RECORDS IN ORDER OF DISTRICT,SCHOOL,
C*****TEACHER,CLASS,MONTH, DAY, AND GROUP.*****
COMMON LOGREC(44), IDST(240),ICLAS(240),
+IGROUP(240),IMONTH(240),IDAY(240),IUNIT(240),
+IMDG(240),IALPHA(240,40),KEY(240)
NOMORE=0
INIT=1
C THE FOLLOWING LOOP READS ONE WEEKLY LOG RECORD AT A TIME.
C EXIT FROM THE READ LOOP IS CAUSED BY 1)DEVICE ERROR,
C 2) TRANSMISSION ABORT, 3) NUMBER OF RECORDS/CLASS GREATER
C THAN 240, 4)EOF READ ON INPUT TAPE, OR 5) ENCOUNTER OF LOG
C RECORD BELONGING TO A DISCREPANT ID NUMBER(USUALLY A NEW CLASS).
C CASES 4 AND 5 ARE CONSIDERED TO BE THE NORMAL EXITS.
40 DO 99 KOUNT=INIT,240
KOUNT2=KOUNT
KEY(KOUNT)=KOUNT
CALL NTRAN(4,2,44,LOGREC,LSTAT)
100 IF(LSTAT.EQ.-1) GO TO 100
C EXIT CASE 4, EOF READ
IF (LSTAT.EQ.-2) GO TO 101
C EXIT CASES 1 OR 2, DEVICE ERROR OR TRANSMISSION ABORT
IF(LSTAT.EQ.-3.OR.LSTAT.EQ.-4) GO TO 102
CALL DECODE(LOGREC,132)
READ(31,200) (IALPHA(KOUNT,L),L=1,40),IDST(KOUNT),
+ ICLASS(KOUNT),IGROUP(KOUNT),IMONTH(KOUNT),IDAY(KOUNT),
+ IUNIT(KOUNT)
200 FORMAT(22A6/17A6,A4,I6,I2,I1,I2,I2,I10,3X)
IGROUP(KOUNT)=IABS(IGROUP(KOUNT))
WRITE(6,205) IDST(KOUNT),ICLASS(KOUNT),IGROUP(KOUNT),
+ IMONTH(KOUNT),IDAY(KOUNT),IUNIT(KOUNT)
205 FORMAT(I8,I4,I2,I4,I4,4X,I10)
IF(KOUNT.EQ.1) GO TO 210
C IF THIS IS THE INITIAL PASS THROUGH THE LOOP FOR THIS CLASS,
C BYPASS THIS CHECK FOR A CHANGE IN CLASS.
IF (ICLASS(KOUNT).NE.IOLDCL.OR.IDST(KOUNT).NE.IOLDST)
+ GO TO 400
C COMBINE THE MONTH, DAY, AND GROUP VALUES READ FOR THIS CLASS,
C RECORD INTO THE MORE READILY SORTABLE INDEX VALUE IMDG
210 IMDG(KOUNT)=IMONTH(KOUNT)*1000+IDAY(KOUNT)*10+IGROUP(KOUNT)
IOLDCL=ICLASS(KOUNT)
IOLDST=IDST(KOUNT)
99 CONTINUE
C EXIT CASE 3, NUMBER OF RECORDS/CLASS GREATER THAN 240
WRITE(6,220)
220 FORMAT(' ERROR...NO.OF RECORDS FOR THIS CLASS EXCEEDS DO
+ LOOP LIMIT OF 240')
STOP
C PREPARE TO SORT ALL RECORDS WITHIN THE PRECEEDING CLASS, NEGLECTING
C THE RECORD LAST READ
400 LESONE=KOUNT2-1
CALL SORT(IMDG,KEY,LESONE)

```

```

C BY LOOKING AT THE SCRAMBLED KEY ARRAY, THE FOLLOWING LOOP
C FETCHES AND WRITES ONE WEEKLY LOG RECORD AT A TIME, IN
C ITS NEW SORTED ORDER.
      DO 300 J=1,LESONE
        NOSUB=KEY(J)
        WRITE(6,206) IDST(NOSUB),ICLASS(NOSUB),
+       IMONTH(NOSUB),IDAY(NOSUB),IGROUP(NOSUB),IUNIT(NOSUB),
+       IMDG(J),KEY(J)
206   FORMAT(I8,I4,I4,I4,I4,I10,I8,I6)
        CALL ENCODE(LOGREC,132)
        WRITE(31,500)(IALPHA(NOSUB,K),K=1,40), IDST(NOSUB),
+       ICLASS(NOSUB),IMONTH(NOSUB),IDAY(NOSUB),IGROUP(NOSUB),
+       IUNIT(NOSUB)
500   FORMAT(22A6/17A6,A4,I6,I2,I2,I2,I1,I10,3X)
        CALL NTRAN(8,1,44,LOGREC,LSTAT)
310   IF(LSTAT.EQ.-1)GOTO 310
C     GO TO WRITE THE LSTAT WRITE ERRORS
        IF(LSTAT.EQ.-2)GOTO 610
        IF(LSTAT.LE.-3)GOTO 90
300   CONTINUE
C IF THIS WAS THE LAST CLASS ON THE TAPE, EXIT THE PROGRAM.
C OTHERWISE, STORE ALL DATA FROM THE RECORD LAST READ(I.E.
C THE FIRST RECORD OF THE NEW CLASS) INTO THE NUMBER ONE POSITION
C OF THE NEXT BATCH TO BE READ AND SORTED.
        IF(NOMORE.EQ.1)GOTO 9
        DO 510 K=1,40
510   IALPHA(1,K)=IALPHA(KOUNT2,K)
        IDST(1)=IDST(KOUNT2)
        ICLASS(1)=ICLASS(KOUNT2)
        IMONTH(1)=IMONTH(KOUNT2)
        IDAY(1)=IDAY(KOUNT2)
        IGROUP(1)=IGROUP(KOUNT2)
        IUNIT(1)=IUNIT(KOUNT2)
        IMDG(1)=IMONTH(KOUNT2)*1000+IDAY(KOUNT2)*10+IGROUP(KOUNT2)
        KEY(1)=1
        IOLDST=IDST(1)
        IOLDCL=ICLASS(1)
        INIT=2
        GO TO 40
101  WRITE(6,111)
111  FORMAT('1',20X,'END OF FILE')
        NOMORE=1
        GO TO 400
610  WRITE(6,620)
620  FORMAT(' END OF TAPE ENCOUNTERED IN WRITE SEQUENCE')
        GOTO 9
90   WRITE(6,91)
91   FORMAT(' WRITE ERROR I.E.....')
102  WRITE(6,112)IDST(KOUNT2-1),IOLDCL,IGROUP(KOUNT2-1),
+       IMONTH(KOUNT2-1),IDAY(KOUNT2-1),LSTAT
112  FORMAT(' ERRORS IN TAPE OR TRANSMISSION ON RECORD ID NUMBER
+       (DSTCGMD)=' , I6,I2,I1,I2,I2,'LSTAT=' , I4)
-9   END FILE 8
      REWIND 8
      STOP
      END

```

SUBROUTINE SORT

1.0 - PROGRAM IDENTIFICATION

SUBROUTINE SORT (SORTON, FOLLOW, NRECS)

2.0 - OBJECTIVE

Sort an array of record ID numbers into ascending order and concurrently move their corresponding index values (initially in order of 1, 2, . . . NRECS).

3.0 - PROGRAM DESCRIPTION

3.1 - Program Logic

SUBROUTINE SORT is passed, 1) SORTON, an array of unsorted record ID numbers, 2) FOLLOW, an index array containing numbers (1, 2, . . . NRECS) corresponding to each record ID number, and 3) NRECS, the number of elements contained in each of those arrays. In the rare event that only one index value is passed, SUBROUTINE SORT bypasses any sorting activity and returns to the main program. Otherwise, an ascending bubble sort is applied. The outer loop is entered and its DO variable is set to indicate the first of a complete pass through record ID array SORTON and index array FOLLOW. Similarly, the inner loop is entered and its DO variable is set to indicate the first comparison made between two elements within the record ID array: If those two elements are out of order with respect to one another, each is shifted to the other's position, as are their corresponding FOLLOW values. Comparisons continue with the last element in the last comparison compared with the element succeeding it until all elements within the array have been compared with their neighbors. Should all comparisons within a pass yield no out-of-orders, both arrays are returned to the main program in their new sorted order. Otherwise, the outer loop is again entered to begin another complete pass through the arrays.

3.2 - Variables

All variables are of integer type

FOLLOW(NRECS) FOLLOW the SORTON elements. This array enters the subroutine, containing numbers 1, 2, 3, . . . NRECS, in that order. As out-of-order record ID's are shifted into order, their corresponding elements in the FOLLOW array are also shifted. FOLLOW is

- finally returned with its elements rearranged, and is used by the main program to fetch and write the records in their new, sorted order.
- FTEMP, FOLLOW TEMPORARY. This value serves as the temporary storage location for an element within the FOLLOW array during the operation in which that element is shifted in position with a neighboring element.
- KMPARE, KOMPARE. This DO variable of the comparison loop appears in the arithmetic expression for the subscript of the two array elements being compared.
- MORE, MORE PASSES. This flag contains a value of either 1 or 0 to indicate whether more passes through the record ID array are required to finish the sort.
- NPASS, NUMBER OF THE PASS. This DO variable indicates the number of times the record ID array has been completely stepped through. If only one record ID is passed to SUBROUTINE SORT, NPASS is set to zero, sort operations are bypassed, and the single record ID value and corresponding FOLLOW value, are immediately returned to the main program.
- NRECS, NUMBER OF RECORDS. As the third argument in the subroutine list, this value indicates the number of record ID's to be sorted.
- SORTON(NRECS), SORT ON this array. The record identification numbers within this array are examined with respect to their order (i.e., relative to their immediate neighbors) within this array so as to trigger sort operations necessary to rearrange them in ascending order.
- STEMP, SORTON TEMPORARY. STEMP serves as the temporary storage location for an element within the SORTON array during the operation in which that element is shifted in position with a neighboring element.

4.0 - SUBROUTINES AND FUNCTIONS

None

5.0 - DATA SPECIFICATIONS

Not Applicable

6.0 - PROGRAM CONSTRAINTS AND LIMITATIONS

6.1 - Programming Language

Univac 1108 FORTRAN V

6.2 - Vendor

University Computing Company

6.3 - Storage Requirements

176 octal words

6.4 - Hardware Configuration

Not applicable

6.5 - Program Parameters

The number of elements contained in either array should not exceed 240 unless array dimensions and loop limits are expanded beforehand.

6.6 - Error Messages

None

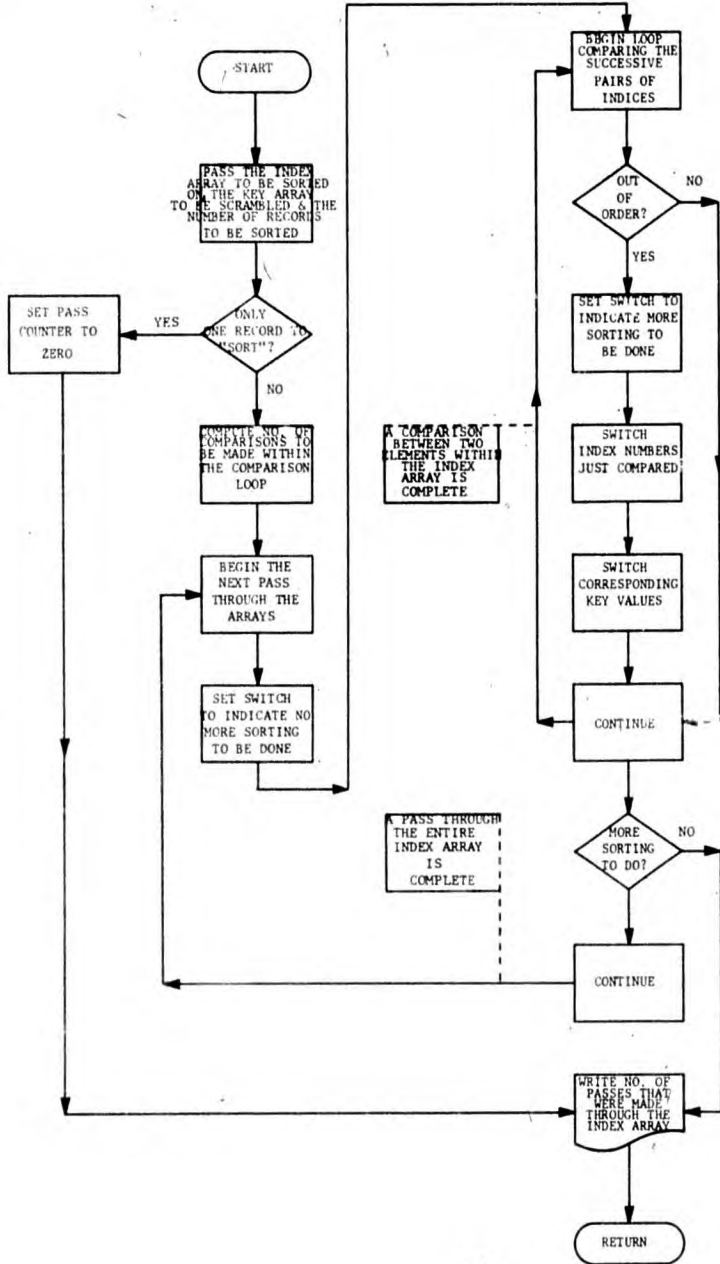
7.0 - OPERATING INSTRUCTIONS

Executed under main program control.



FLOWCHART LAYOUT FORM

PROGRAMMER : Tom Foote _____ DATE : _____
 PROGRAM I.D. : Subroutine Sort _____ PAGE 1 of 1 _____



9.0 - PROGRAM LISTING

```

C THE FOLLOWING SUBROUTINE SORTS THE ARRAY "SORTON," (SMALLEST
C VALUE FIRST, GREATEST LAST) WHILE CORRESPONDINGLY REARRANGING THE
C KEY ARRAY "FOLLOW." NRECS IS THE TOTAL NUMBER OF ELEMENTS TO BE
C SORTED.
      SUBROUTINE SORT(SORTON,FOLLOW,NRECS)
      INTEGER SORTON,FOLLOW,STEMP,FTEMP
      DIMENSION SORTON(240),FOLLOW(240)
C IF ONLY ONE RECORD IS PASSED TO THE SUBROUTINE, SET
C NPASS, EQUAL TO ZERO, WRITE "NPASS=0", AND RETURN
C TO THE MAIN PROGRAM.
      IF(NRECS.GT.1)GOTO 10
      NPASS=0
      GO TO 22
10 LESONE=NRECS-1
C NPASS IS INCREMENTED EACH TIME A PASS THROUGH THE ENTIRE
C "SORTON" ARRAY IS TO BE MADE.
      DO 21 NPASS=1,NRECS
      TURN THE "MORE" SWITCH OFF TO INDICATE THAT NO MORE PASSES
C ARE REQUIRED TO FINISH THE SORT.
      MORE=0
C KMPARE IS INCREMENTED EACH TIME A NEW ELEMENT WITHIN THE
C "SORTON" ARRAY IS TO BE COMPARED WITH THE ELEMENT IMMEDIATELY
C PRECEDING IT.
      DO 20 KMPARE=1,LESONE
      IF(SORTON(NRECS-KMPARE+1).GE.SORTON(NRECS-KMPARE)) GO TO 20
C TURN THE "MORE" SWITCH ON TO INDICATE THAT MORE PASSES
C ARE REQUIRED TO FINISH THE SORT.
      MORE=1
C SWITCH THE TWO "SORTON" VALUES THAT WERE OUT OF ORDER.
      STEMP=SORTON(NRECS-KMPARE)
      SORTON(NRECS-KMPARE)=SORTON(NRECS-KMPARE+1)
      SORTON(NRECS-KMPARE+1)=STEMP
C SWITCH THEIR CORRESPONDING "FOLLOW" VALUES.
      FTEMP=FOLLOW(NRECS-KMPARE)
      FOLLOW(NRECS-KMPARE)=FOLLOW(NRECS+1-KMPARE)
      FOLLOW(NRECS+1-KMPARE)=FTEMP
20 CONTINUE
C IF MORE PASSES ARE REQUIRED TO FINISH THE SORT, PASS THROUGH
C THE ENTIRE "SORTON" ARRAY ONCE AGAIN. OTHERWISE RETURN
C TO THE MAIN PROGRAM.
      IF(MORE.EQ.1) GO TO 21
      GO TO 22
21 CONTINUE
C WRITE THE NUMBER OF PASSES IT REQUIRED TO COMPLETE THIS SORT.
22 WRITE(6,30) NPASS
30 FORMAT(' NPASS=',I5)
      RETURN
      END

```


SCHOOL DISTRICT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	TRACHER	CO.	EDUC.	PHYS.	AV.	ARTS	LANG.	SC.	HEALTH	PE
MONTH	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31										
DAY	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31										

8th "Grid" of "Field" 5.

Field numbers correspond to subscripts of the IALPHA array. The grid marked within each field is scanner coded to an alpha character. (usually first grid coded to "A," second grid to "B," and so on), which is in turn stored by the WLSORT program into the IALPHA array.

IS THE CLASS GROUPED YES NO IF YES, MARK GROUP NUMBER FOR THIS SHEET 7 3 3 3

DAY	UNIT	INSTRUCTIONAL TIME	CLERICAL TIME
MONDAY	145	15:00-5:00	16:00-5:00
TUESDAY	20	18:00-5:00	19:00-5:00
WEDNESDAY	23	21:00-5:00	22:00-5:00
THURSDAY	26	24:00-5:00	25:00-5:00
FRIDAY	26	27:00-5:00	28:00-5:00

MONDAY		TUESDAY		WEDNESDAY		THURSDAY		FRIDAY	
Skill or Activity	Daily Assessment Passed	Skill or Activity	Daily Assessment Passed	Skill or Activity	Daily Assessment Passed	Skill or Activity	Daily Assessment Passed	Skill or Activity	Daily Assessment Passed
29	30	31	32	33	34	35	36	37	38
39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58
59	60	61	62	63	64	65	66	67	68
69	70	71	72	73	74	75	76	77	78
79	80	81	82	83	84	85	86	87	88
89	90	91	92	93	94	95	96	97	98
99	100	101	102	103	104	105	106	107	108
109	110	111	112	113	114	115	116	117	118
119	120	121	122	123	124	125	126	127	128
129	130	131	132	133	134	135	136	137	138
139	140	141	142	143	144	145	146	147	148
149	150	151	152	153	154	155	156	157	158
159	160	161	162	163	164	165	166	167	168
169	170	171	172	173	174	175	176	177	178

DAY	UNIT	INSTRUCTIONAL TIME	CLERICAL TIME
MONDAY	179	180:00-5:00	181:00-5:00
TUESDAY	182	183:00-5:00	184:00-5:00
WEDNESDAY	185	186:00-5:00	187:00-5:00
THURSDAY	188	189:00-5:00	190:00-5:00
FRIDAY	191	192:00-5:00	193:00-5:00

	OUTCOME 1		OUTCOME 2		OUTCOME 3		OUTCOME 4		Criterion Exercise Retest
	Practice Exercise A	OTHER	Practice Exercise B	OTHER	Practice Exercise C	OTHER	Practice Exercise D	OTHER	
MONDAY	194	195	196	197	198	199	200	201	202
TUESDAY	203	204	205	206	207	208	209	210	211
WEDNESDAY	212	213	214	215	216	217	218	219	220
THURSDAY	221	222	223	224	225	226	227	228	229
FRIDAY	230	231	232	233	234	235	236	237	238

INITIAL INSTRUCTION

SECOND INSTRUCTION

At the end of every week send the completed form to the District Program Supervisor.

AT THE TOP INDICATE:

1. Date of the last school day of the week.
 - a. Month: Blacken the rectangle around the appropriate number.
 - b. Day: Blacken the appropriate rectangle in both rows. If the date is 1 through 9, blacken the zero in the first row and the appropriate rectangle in the second row.

Examples

MONTH	0	1	2	3	4	5	6	7	8	9
DAY	0	1	2	3	4	5	6	7	8	9

October 4

MONTH	0	1	2	3	4	5	6	7	8	9
DAY	0	1	2	3	4	5	6	7	8	9

January 23

FOR EACH DAY INDICATE:*

1. The sequence number of the book you are using by blackening the appropriate rectangle under one of these series names:
 - RRS (Reading Readiness Series)
 - PPS (Pre-Primer Series)
 - PS (Primer Series)
 - FRS (First Reader Series)
2. Whether a test was given. If yes, blacken the "Y" rectangle. If no, blacken the "N" rectangle.
3. Time spent on program in
 - a. Initial instruction: Blacken an "I" rectangle under one of the time ranges. If no time was spent, do not make a mark.
 - b. Review: Blacken an "R" rectangle under one of the time ranges. If no time was spent, do not make a mark.

AT THE END OF THE WEEK INDICATE:

1. Last page completed in the text that week: Blacken the appropriate numbers in the three rows. If the page number is less than 10, blacken the zeroes in the top two rows and the appropriate number in the bottom row. If the page number is between 10 and 99, blacken the zero in the top row and the appropriate numbers in the bottom two rows.

* If the class is grouped, follow remainder of the directions, once for each group, under the headings Group 1, Group 2, etc.

APPENDIX B

EXAMPLES OF THE INPUT RECORDS

238 ALPHA Character Responses Per Record

District Group
 School Month
 Teacher Day
 Class Relevent Unit
 112 34 5678910

FILE NUMBER	RECORD NUMBER	LENGTH	District	Group	School	Month	Teacher	Day	Class	Relevent Unit
FILE NUMBER 1 AAAATCCALBCC HA BC BABA	RECORD NUMBER 2485 ACAABAACBACAACA	LENGTH 44								RC
FILE NUMBER 1 AAAATCCAAAJC	RECORD NUMBER 2486 BCARCARCACBACABA	LENGTH 44 BC BA					BABA		BA	
FILE NUMBER 1 AAAATCCAAABGC	RECORD NUMBER 2487 BCABCANCBCAHCA	LENGTH 44							BC BA BABABA	
FILE NUMBER 1 AAAATCCAAACGC	RECORD NUMBER 2488 BCABCARCACBACACA	LENGTH 44							BC BA BABABA	
FILE NUMBER 1 AAAATCCAAACJC	RECORD NUMBER 2489 BCABC CC CC	LENGTH 44							BC BA BABABA	
FILE NUMBER 1 AAAATCCABACC	RECORD NUMBER 2490 CCACCACCACCACBABA	LENGTH 44 RA BC					BABA		BA	
FILE NUMBER 1 AAAATCCABACC	RECORD NUMBER 2491 CCACCA	LENGTH 44							BABA	
FILE NUMBER 1 AAAATCCABACC	RECORD NUMBER 2492 CCACCACCACCA	LENGTH 44							BA BC BABA BABABA	
FILE NUMBER 1 AAAATCCABACC	RECORD NUMBER 2493 CCACCACHACBA	LENGTH 44							BA BA	
FILE NUMBER 1 AAAATCCACACC	RECORD NUMBER 2494 CCACCACCACCACBA	LENGTH 44							BA BABABA	
FILE NUMBER 1 AAAATCCACACC	RECORD NUMBER 2495 CCB F	LENGTH 44							BA BABABA	
FILE NUMBER 1 AAAATCCACACC	RECORD NUMBER 2496 ACHARRACCACACCBA	LENGTH 44 BABA BABA							CCBCCA C B B BB	
FILE NUMBER 1 AAAATCCACACC	RECORD NUMBER 2497 ACAACAACAACAACA	LENGTH 44							BABA BABA BABA	

19

APPENDIX C

EXAMPLES OF THE OUTPUT RECORDS

238 ALPHA Character Responses Per Record

FILE NUMBER	RECORD NUMBER	LENGTH	District		Month		Teacher	Class	Group	Relevant Unit										
			School	Year	1	2				3	4	5	6	7	8					
FILE NUMBER 1	RECORD NUMBER 60	LENGTH 44	BA	BC	130101	2	8130													
BAABDBBABBDC	IFCIBFICIGCIFD		BA	BC																
FILE NUMBER 1	RECORD NUMBER 61	LENGTH 44	BA	BABC	130101	2	8200													
BAABDBBABCAC	JGCJFBJEBJCCJGBBC		BA	BABC																
FILE NUMBER 1	RECORD NUMBER 62	LENGTH 44	BC	BA	130101	2	8270													
BAARDDBBABCNC	JGCJNCJEC		BC	BA																
FILE NUMBER 1	RECORD NUMBER 63	LENGTH 44	BB	BC	130102	1	3260													
BAAADBCAICCC	ACBADBADAAADADC		BB	BC																
FILE NUMBER 1	RECORD NUMBER 64	LENGTH 44	BC	BC	130102	1	4170													
BAAADBCAJAOC	AD AHCACBACCAGA		BC	BC																
FILE NUMBER 1	RECORD NUMBER 65	LENGTH 44	BC	BC	130102	1	4170													
BAAADBCAJAJC	ACBACBAC ACA		BC	BC																
FILE NUMBER 1	RECORD NUMBER 66	LENGTH 44	BC	BC	130102	1	4170													
BAAADBCAJBRC	APC		BC	BC																
FILE NUMBER 1	RECORD NUMBER 67	LENGTH 44	BC	BC	130102	1	4170													
BAAADBCAJJEC	CBGAB A		BC	BC																
FILE NUMBER 1	RECORD NUMBER 68	LENGTH 44	BA	BC	130102	1	4240													
BAAADBCAJDBC	BC BCBBCBCCBGA		BA	BC																
FILE NUMBER 1	RECORD NUMBER 69	LENGTH 44	BA	BC	130102	1	4310													
BAAADBCAKANC	BFBBFBBCBBF BCB		BA	BC																
FILE NUMBER 1	RECORD NUMBER 70	LENGTH 44	BA	BC	130102	1	5170													
BAAADBCAKBEC	BCFCBCCGACFA		BA	BC																
FILE NUMBER 1	RECORD NUMBER 71	LENGTH 44	BA	BC	130102	1	5140													
BAAADBCAKCBC	C BCACCE CFBCCB		BA	BC																
FILE NUMBER 1	RECORD NUMBER 72	LENGTH 44	BA	BC	130102	1	5210													
BAAADHCAKCCG	CEACHACH		BA	BC																