

**PUBLICATIONS  
UPDATE**

General

Fundamentals of COBOL  
Sorting

Programmer Reference

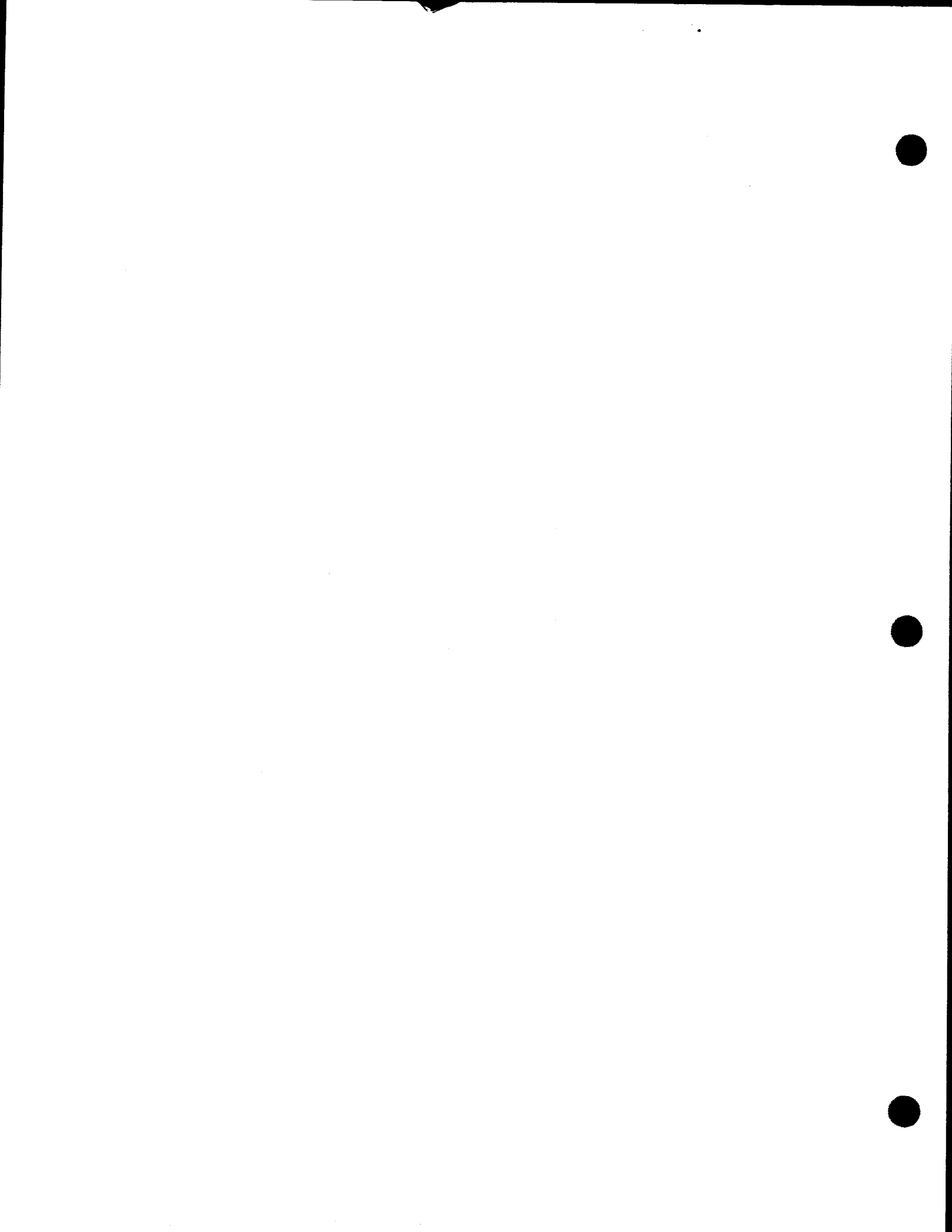
UP-7503.3 Rev. 1-A

This Library Memo announces the release and availability of Updating Package A to "SPERRY UNIVAC Fundamentals of COBOL Sorting Programmer Reference", UP-7503.3 Rev. 1.

This update includes minor corrections to the document.

Copies of Updating Package are now available for requisitioning. Either the updating package only or the complete manual with the updating package may be requisitioned by your local Sperry Univac representative. To receive only the updating package, order UP-7503.3 Rev. 1-A. To receive the complete manual, order UP-7503.3 Rev. 1.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ and MZ	Mailing Lists 1D, 3D, 4D, 5D, 8, 9, 10, 11, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 37, 38, 43, 44, 51, 52, 53, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 65, 66, 75, 75U, 76, 76U, 81, 83, 89 (Package A to UP-7503.3 Rev. 1, 5 pages plus Memo)	Library Memo  RELEASE DATE: January, 1981



**UNIVAC<sup>®</sup>**

**FUNDAMENTALS  
OF COBOL**

**SORTING**

**PROGRAMMERS REFERENCE**

This manual is published by the Univac Division of Sperry Rand Corporation in loose leaf format. This format provides a rapid and complete means of keeping recipients apprised of UNIVAC® Systems developments. The information presented herein may not reflect the current status of the programming effort. For the current status of the programming, contact your local Univac Representative.

The Univac Division will issue updating packages, utilizing primarily a page-for-page or unit replacement technique. Such issuance will provide notification of software changes and refinements. The Univac Division reserves the right to make such additions, corrections, and/or deletions as, in the judgment of the Univac Division, are required by the development of its Systems.

UNIVAC is a registered trademark of Sperry Rand Corporation.





## PREFACE

This manual explains the general organizational aspects and specific statement formats of a COBOL Sort program. It is another in the series of manuals entitled "Fundamentals of COBOL". The information in this manual is oriented toward level 2 of the USASI COBOL standard for Sorting (see discussion of COBOL levels and modules in "Fundamentals of COBOL-Language").

Sorting generally constitutes a significant portion of the work load in a commercial data processing installation. The COBOL Sort feature offers the user a means of obtaining an efficient sort function without programming the function in detail.

A COBOL program may contain any number of sorts, and each sort may have its own independent special procedures. The special processing procedures may include addition, deletion, creation, altering, editing, or other modification of the individual records in the file.





## CONTENTS

<b>PREFACE</b>	1 to 1
<b>CONTENTS</b>	1 to 1
<b>1. INTRODUCTION</b>	1-1 to 1-2
1.1. SORT PROGRAM ORGANIZATION	1-1
1.1.1. SORT Operation	1-1
1.1.2. Input and Output Procedure	1-2
1.2. TYPES OF SORT PROGRAMS	1-2
<b>2. PROGRAMMING CONSIDERATIONS</b>	2-1 to 2-7
2.1. GENERAL	2-1
2.2. ENVIRONMENT DIVISION	2-1
2.2.1. FILE-CONTROL	2-1
2.2.2. I-O-CONTROL	2-2
2.2.2.1. SAME AREA	2-2
2.3. DATA DIVISION	2-3
2.3.1. Sort File Description	2-4
2.4. PROCEDURE DIVISION	2-4
2.4.1. RELEASE	2-5
2.4.2. RETURN	2-5
2.4.3. SORT	2-6
<b>APPENDIX A. EXAMPLES OF A SORT PROGRAM</b>	A-1 to A-4



# 1. INTRODUCTION

## 1.1. SORT PROGRAM ORGANIZATION

Like any other file, the sort file is a set of records. It is described in the Data Division by means of a special type of file description called a Sort File Description. The sort file is an internally contained, intermediate representation of the file, which follows the initial input of unsorted records and precedes the final output of sorted records.

### 1.1.1. SORT Operation

In general, a SORT operation proceeds as follows:

- (1) Control passes to a SORT statement. The SORT statement specifies:
  - (a) the sort file to be created;
  - (b) the keys on which the records are to be sorted; and
  - (c) the input and output procedures or names the source of the unsorted input records and those files which are to receive the sorted output records.
- (2) The input procedure, if named in the SORT statement, is executed. If no input procedure is specified, the USING option of the SORT statement must be used to name the input file. The effect of either option is to make input records available to the SORT operation.
- (3) The records are sorted in accordance with the keys specified in the KEY clause of the SORT statement.
- (4) In this phase of the operation, the sorted records are moved from the sort file to the output file. This can be accomplished in either of two ways:
  - (a) By the output procedure, if one is named in the SORT statement. Control is passed to this procedure by the SORT statement.

or

  - (b) If no output procedure is named, the GIVING option of the SORT statement must be used to specify the output file.
- (5) The operation of the SORT statement terminates and control passes to the next statement in sequence.

### 1.1.2. Input and Output Procedures

Whenever an input or output procedure is in control, all control transfers (by ALTER, GO TO, or PERFORM statements) occurring within these procedures must refer to procedures contained within that input or output procedure. Conversely, control cannot be transferred into an input or output procedure from points outside the physical limits of the given input or output procedure. Neither the input nor the output procedure may contain a SORT statement.

### 1.2. TYPES OF SORT PROGRAMS

There are two types of sort programs:

#### ■ Basic

In a basic SORT program the Procedure Division contains one SORT statement and a STOP RUN statement in the first nondeclaratives section. Other sections consist exclusively of input and output procedures.

#### ■ Extended

An extended SORT program may contain more than one SORT statement. These SORT statements can be located anywhere in the Procedure Division except in the DECLARATIVES section or in the input and output procedures identified within a SORT statement.

When control passes to the SORT statement the SORT operation is completely automatic, affected only by the KEY parameters and the file or record parameters contained in the SORT statement. The input and output procedures may contain any necessary statements for special processing procedures that will add, delete, select, alter, or otherwise modify records.

Special procedures in the input procedure may be used to modify key parameters. Such modifications will occur prior to the actual sort, and can be made to affect the final sequence of the records in the output file, following the SORT operation. Record modifications in the output procedures will appear in the sorted output but do not affect the final sequence.

## 2. PROGRAMMING CONSIDERATIONS

### 2.1. GENERAL

The remainder of this manual discusses the specific formats in the COBOL language used in writing a source program which includes a SORT operation. Formats in the Environment, Data, and Procedure Divisions are described.

### 2.2. ENVIRONMENT DIVISION

The COBOL SORT feature may require the use of the following additional facilities in the Environment Division of a COBOL source program.

#### 2.2.1. FILE-CONTROL

**Format:**

*Option 1:*

```
FILE-CONTROL. { SELECT file-name  
ASSIGN TO [integer] implementor-name-1 [, implementor-name-2] . . . } . . .
```

*Option 2:*

```
FILE-CONTROL. { SELECT file-name ASSIGN TO implementor-name-1  
[, implementor-name-2] . . . OR implementor-name-3 [, implementor-name-6] . . .  
[ FOR MULTIPLE REEL ] . } . . .
```

**Description:**

Options 1 and 2 are similar to the FILE-CONTROL paragraph shown and discussed in "Fundamentals of COBOL-Language". This information will not be repeated here.

Option 1 must be specified for sort files described in the Data Division.

Option 2 must be used when the GIVING option of the SORT statement is specified. When sorting, the programmer cannot reserve alternate areas in memory.

In Option 1, the sort file will appear on each hardware device specified by implementor-name-1, implementor-name-2, etc. The OR clause in Option 2 enables the programmer to specify several alternate hardware devices and the sort file will appear on only one of the designated hardware devices. After the SORT operation is completed, the COBOL SORT feature provides an indication as to which hardware device contains the sort file. The hardware device containing the sort file is automatically addressed when this file is OPENed for input.

### 2.2.2. I-O-CONTROL

The basic format of the I-O-CONTROL paragraph is shown and discussed in "Fundamentals of COBOL—Language". Note that only in sorting is it permissible to specify a sort-file-name as the object of a RERUN or MULTIPLE FILE clause. The use of the SAME AREA clause in connection with a sort file is discussed below.

#### 2.2.2.1. SAME AREA

**Format:**

```
[ ; SAME { RECORD } AREA FOR file-name-1 { , file-name-2 } . . . ] . . .
```

**Description:**

The SAME AREA clause names two or more files that are to share the same main storage area.

If the RECORDS option is selected, then:

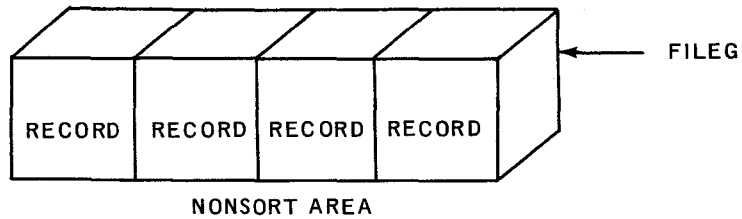
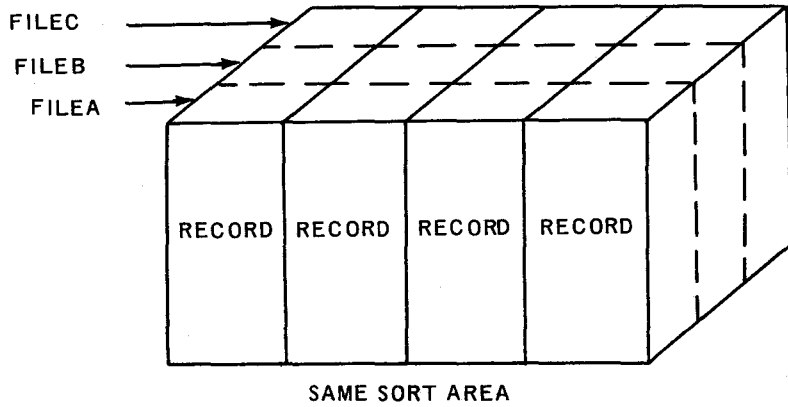
- All specified files can be open during any phase of the SORT operation.
- All specified files, including sort files, share the same *record* area.
- This record area may contain only one logical record at any one time, and this record is the record currently being processed. It is the programmer's responsibility to determine which record of which file is in the record area at any point in the program.

If the SORT option is selected, then:

- All specified sort files share the same buffer storage area.
- Only the sort files associated with the particular phase (input procedure, sort, or output procedure) of a SORT operation can be open during the execution of that particular phase. It is the programmer's responsibility to make certain that only the sort files named in this clause are open and that all other files are closed.
- Those specified files, which are not sort files, do not share the same storage area as the sort files but will be assigned their own storage areas by the compiler.

If the RECORDS option is chosen, the action is similar to that described in the I-O-CONTROL paragraph discussion in "Fundamentals of COBOL—Language".

Assume that the SORT option is selected and that the SAME SORT AREA clause specifies four files, three of which are sort files (files A, B, and C) and the fourth file (file G) is a nonsort file. The diagram below illustrates the storage configuration that the compiler will set up in memory.



The storage area shared by the sort files is the area that will be used by the open sort file for storage. The FILEG area and any other area assigned to files not named in the SAME SORT AREA clause may also be allocated as needed for sorting the sort files named in the SAME SORT AREA clause. The extent of such allocation must be specified by the implementor.

### 2.3. DATA DIVISION

The COBOL SORT feature requires the use of a special type of file description called a Sort File Description, the format and rules of which are explained below.

#### 2.3.1. Sort File Description

**Formats:**

*Option 1:*

SD *file-name* COPY *library-name*

*Option 2:*

SD *file-name*

[ ; RECORD CONTAINS [*integer-1* TO] *integer-2* CHARACTERS ]

[ ; DATA { RECORD IS  
RECORDS ARE } *data-name-1* [, *data-name-2*] . . . ] .

**Description:**

The Sort File Description furnishes information concerning the physical structure, identification, and record names of the file to be sorted.

Option 1 is used only when the COBOL library contains the full Sort File Description entry; otherwise Option 2 must be used.

The level indicator SD must precede the file-name to identify the beginning of the Sort File Description entry.

All semicolons shown are optional, although the entry must be terminated by a period.

The clauses which comprise the Sort File Description entry also comprise a Standard File Description and are discussed in "Fundamentals of COBOL—Language". Such clauses are optional except for the DATA RECORDS clause, and the order of their appearance is immaterial.

### 2.4. PROCEDURE DIVISION

The COBOL sort feature makes use of three verbs in the Procedure Division, in addition to the verbs of the basic language:

- RELEASE
- RETURN
- SORT



### 2.4.1. RELEASE

**Format:**

RELEASE *record-name* [FROM *identifier*]

**Description:**

The RELEASE statement transfers the records from the input file to the initial phase of a SORT operation. One record is transferred each time a RELEASE statement is executed. RELEASE may be used only within the range of an input procedure associated with a SORT statement. This SORT statement must refer to a sort file whose DATA RECORD clause contains the same record-name as the RELEASE statement.

The record must be read into memory before it can be released to the SORT program.

Record-name and identifier must not be the same storage area in memory. The contents of identifier are moved to record-name; then the contents of record-name are released to the sort file. After the transfer, information in the record area is no longer available, but information in the data area associated with identifier is available. Moving is effected by the same rules as govern the MOVE statement without the CORRESPONDING option.

After the RELEASE statement is executed, the contents of record-name are no longer available. When control passes from the input procedure, the sort file contains all the records placed in the sort file by the RELEASE statements.

### 2.4.2. RETURN

**Format:**

RETURN *file-name* RECORD [INTO *identifier*]

; AT END *imperative-statement*

**Description:**

The RETURN statement is a request for the next ordered record from a sort file. It may appear only within an output procedure associated with a SORT statement for file-name, (i.e., file-name must be a sort file described by an SD entry).

The execution of the RETURN statement causes the next record, in the order specified by keys listed in the SORT statement, to be made available for processing. The record is located in the record area associated with the sort file.

When a file consists of more than one type of logical record, the shared storage area is implicitly redefined for each record and only the information present in the current record is accessible.

When the INTO option is used, the data record is available in both the input record area and the data area associated with identifier. This option may be used only when the input file contains just one type of record. Identifier and the record area associated with the file must not be the same storage area in memory. Moving is effected according to the rules governing the MOVE statement without the CORRESPONDING option.

When the SORT key indicates the end of the sort file, the imperative statement in the AT END phrase is executed. After this, no RETURN statements may be executed within the current output procedure.

### 2.4.3. SORT

#### Format:

```
SORT file-name-1 ON { DESCENDING } KEY { identifier-1 } . . .  
[ ; ON { DESCENDING } KEY { identifier-2 } . . . ] . . .  
{ INPUT PROCEDURE IS section-name-1 [THRU section-name-2] }  
{ USING file-name-2. }  
{ OUTPUT PROCEDURE IS section-name-3 [THRU section-name-4] }  
{ GIVING file-name-3. }
```

#### Description:

The SORT statement creates the sort file in a three-phase operation:

Phase 1 – Executes an input procedure or transfers the unsorted records directly from another file (by means of USING option).

Phase 2 – Sorts the records in accordance with specified keys.

Phase 3 – Executes output procedure or transfers sorted records directly to an output file (by means of GIVING option).

*File-name-1* must be a sort file described by a Sort File Description in the Data Division, and each KEY identifier must be a data item described in records associated with *file-name-1*.

*Section-name-1* is the name of an input procedure, and *section-name-3* is the name of an output procedure. *File-name-2* and *file-name-3* must be described by a File Description (not a Sort File Description) in the Data Division.

Upon encountering a SORT statement specifying input and output procedures, by means of the USING/GIVING options, the compiler automatically inserts the control mechanisms needed to execute the three phases of a SORT operation, and to pass control to the statement following the SORT statement after the SORT operation is completed.

Each input and output procedure must consist of one or more consecutively written sections, no part of which can be common to both procedures. The input procedure must include at least one RELEASE statement in order to transfer records to the sort file. The output procedure must include at least one RETURN statement to transfer sorted records to the output file.

Control must not be passed to an input procedure or to an output procedure except by a SORT statement because RELEASE and RETURN statements have no meaning unless controlled by a SORT statement.

Input and output procedures may consist of any procedures needed to select, modify, create, or copy records. Three restrictions must be observed in writing the procedural statements that comprise such procedures:

- (a) No SORT statement may be written as part of an input or output procedure.
- (b) Neither an input nor an output procedure may contain any transfer of control (by means of ALTER, GO TO, and PERFORM statements) to points outside the procedure.

Sort sequence depends on the use of the ASCENDING or DESCENDING option:

- (a) When ASCENDING is used, the sort sequence is from the lowest value of the key to the highest value according to the ordered character set specified by the implementor.
- (b) When DESCENDING is used, the sort sequence is from the highest value of the key to the lowest value.

The record description for every record listed in the DATA RECORDS clause of the sort file must contain the KEY items identifier-1, identifier-2, etc. The following rules apply to KEY items:

- (a) They may not be variable length items.
- (b) When a KEY item appears in more than one record, the relevant data descriptions must be equivalent in each record.
- (c) The KEY item must be the same number of character positions from the beginning of each record.
- (d) They may neither contain an OCCURS clause nor be subordinate to entries that contain an OCCURS clause.

If the USING option is specified, all the records in file-name-2 are transferred automatically to file-name-1. File-name-2 must not be open since the SORT statement automatically performs the necessary OPEN, READ, USE, and CLOSE functions for file-name-2.

If the GIVING option is specified, all the sorted records in file-name-1 are automatically transferred to file-name-3 as the implied output procedure for this SORT statement. When the SORT statement is executed, file-name-3 must not be open. The SORT statement automatically opens file-name-3 before transferring the records and automatically closes the file after the transfer is completed.

Examples of sort programs using the input and output procedural approach and the USING and GIVING option approach are shown in Appendix A of this manual.



## APPENDIX A. EXAMPLES OF A SORT PROGRAM

The examples which follow illustrate the two basic organizational approaches that may be adopted by the programmer in writing a SORT program in COBOL. Only those functions that are necessary for illustrating the COBOL SORT feature are shown.

Example 1:

```
ENVIRONMENT DIVISION.  
. .  
INPUT-OUTPUT SECTION.  
FILE-CONTROL.  
    SELECT TO-BE-SORTED ASSIGN TO MAG-TAPE-D.  
    SELECT AFTER-THE-SORT ASSIGN TO MAG-TAPE-E.  
    SELECT WORK-ING ASSIGN TO MAG-TAPE-A, B, C.  
    SELECT FOR-THE-PRINTER ASSIGN TO PRINTER.  
. .  
DATA DIVISION.  
FILE SECTION.  
FD TO-BE-SORTED. . .DATA RECORD IS A-FILE  
    RECORD CONTAINS 80 CHARACTERS.  
01 A-FILE.  
    02 1-ACCOUNT PICTURE IS X(4).  
    02 1-TYPE PICTURE IS X.  
    02 1-AREA PICTURE IS XXX.  
. .  
FD AFTER-THE-SORT. . .DATA RECORD IS B-FILE  
    RECORD CONTAINS 80 CHARACTERS.  
01 B-FILE PICTURE IS X(80).  
. .  
SD WORK-ING FILE. . .DATA RECORD IS C-FILE.  
01 C-FILE.  
    02 3-ACCOUNT PICTURE IS X(4).  
    02 3-TYPE PICTURE IS X.  
    02 3-AREA PICTURE IS XXX.
```

FD FOR-THE-PRINTER. . .DATA RECORD IS D-FILE.  
01 D-FILE PICTURE IS X(80).  
PROCEDURE DIVISION.  
OPENING.  
    OPEN INPUT TO-BE-SORTED.  
    OPEN OUTPUT AFTER-THE-SORT FOR-THE-PRINTER.  
A-SORT SECTION.  
    SORT WORK-ING ON ASCENDING KEY 3-ACCOUNT 3-TYPE 3-AREA  
    INPUT PROCEDURE IS B-SORT THRU B1-SORT  
    OUTPUT PROCEDURE IS C-SORT THRU C1-SORT.  
    GO TO OK-PRINT.  
B-SORT SECTION.  
    READ TO-BE-SORTED AT END GO TO B1-SORT.  
    RELEASE C-FILE FROM A-FILE.  
    GO TO B-SORT.  
B1-SORT SECTION.  
    CLOSE TO-BE-SORTED.  
C-SORT SECTION.  
    RETURN WORK-ING RECORD INTO B-FILE AT END GO TO C1-SORT.  
    WRITE B-FILE.  
    GO TO C-SORT.  
C1-SORT SECTION.  
    CLOSE AFTER-THE-SORT.  
OK-PRINT SECTION.  
    OPEN INPUT AFTER-THE-SORT.  
A-LOOP.  
    READ AFTER-THE-SORT AT END GO TO A-1-MOVE.  
    MOVE SPACES TO D-FILE.  
    MOVE B-FILE TO D-FILE.  
    WRITE D-FILE.  
    GO TO A-LOOP.  
A-1-MOVE.  
    EXIT.  
ALL-DONE SECTION.  
    CLOSE AFTER-THE-SORT FOR-THE-PRINTER.  
    STOP RUN.

In this example, the files defined by FD entries are opened before the SORT statement is executed. The SORT statement sets the sorting mechanism in motion by transferring control to the INPUT PROCEDURE. The files can be opened within the INPUT PROCEDURE, but of course the programmer must take care that he does not attempt to OPEN the input file more than once or CLOSE the input file before input is completed.

Sorting is conducted on an ASCENDING KEY. Note that the KEY elements are in the sort file WORK-ING, not the input file TO-BE-SORTED.

The INPUT PROCEDURE is completed when the entire input file has been transferred to the sort file. The SORT mechanism then transfers control to the OUTPUT procedure, and records are written on the file AFTER-THE-SORT in the order specified by the KEY parameters. The sorted records are then listed after the SORT function has been accomplished.

Example 2:

```
.  
. .  
DATA DIVISION.  
FILE SECTION.  
FD TO-BE-SORTED. . .RECORD CONTAINS 80 CHARACTERS.  
  DATA RECORD IS A-FILE.  
01 A-FILE  
  02 1-ACCT PICTURE IS X(4).  
  02 1-DEPT PICTURE IS X(4).  
  02 1-BODY PICTURE IS X(72).  
FD AFTER-THE-SORT RECORD CONTAINS 80 CHARACTERS.  
  . . .DATA RECORD IS B-FILE.  
01 B-FILE.  
  02 2-ACCT PICTURE IS X(4).  
  02 2-DEPT PICTURE IS X(4).  
  02 2-BODY PICTURE IS X(72).  
SD WORK-ING FILE. . .DATA RECORD IS C-FILE.  
01 C-FILE.  
  02 3-ACCT PICTURE IS X(4).  
  02 3-DEPT PICTURE IS X(4).  
  02 3-BODY PICTURE IS X(72).  
FD FOR-THE-PRINTER. . .DATA RECORD IS D-FILE.  
01 D-FILE.  
  02 4-ACCT PICTURE IS X(4).  
  02 4-DEPT PICTURE IS X(4).  
  02 4-BODY PICTURE IS X(72).  
. .  
PROCEDURE DIVISION.  
OPENING.  
  OPEN OUTPUT FOR-THE-PRINTER.  
A-SORT SECTION.  
  SORT WORK-ING ON ASCENDING KEY.  
  3-ACCT 3-DEPT.  
  USING TO-BE-SORTED GIVING AFTER-THE-SORT.  
OK-PRINT SECTION.  
  OPEN INPUT AFTER-THE-SORT.
```

A-LOOP.

READ AFTER-THE-SORT AT END GO TO A-1-MOVE.  
MOVE SPACES TO D-FILE.  
PERFORM A-MOVE.  
WRITE D-FILE.  
GO TO A-LOOP.

A-MOVE.

MOVE B-FILE TO D-FILE.

A-1-MOVE.

EXIT.

ALL-DONE SECTION.

CLOSE AFTER-SORT FOR-THE-PRINTER.  
STOP RUN.

In this second example, the files that are directly involved in the sort must not be open at the time the SORT statement is encountered. This is because the USING and GIVING options perform the functions of the INPUT PROCEDURE and OUTPUT PROCEDURE (i.e., OPEN, READ RELEASE, RETURN, USE, WRITE, and CLOSE) automatically. Although the use of the GIVING and USING options requires a minimum amount of coding, it is not possible to add, delete, or modify individual records of the sort file in any way (i.e., no special processing procedures may be incorporated into a SORT function when this technique is adopted).



Comments concerning this manual may be made in the space provided below. Please fill in the requested information.

System: \_\_\_\_\_

Manual Title: \_\_\_\_\_

UP No: \_\_\_\_\_ Revision No: \_\_\_\_\_ Update: \_\_\_\_\_

Name of User: \_\_\_\_\_

Address of User: \_\_\_\_\_

Comments:

CUT


FOLD

FIRST CLASS  
PERMIT NO. 21  
BLUE BELL, PA.

**BUSINESS REPLY MAIL**

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

**SPERRY**  **UNIVAC**

P.O. BOX 500  
BLUE BELL, PA.  
19422

ATTN: SYSTEMS PUBLICATIONS DEPT.

CUT

FOLD

## USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

*Please note: This form is not intended to be used as an order blank.*

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update No.)*

### Comments:

Cut along line.

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)  
Thank you for your cooperation

FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

---

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

---

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY UNIVAC**

**ATTN.: SYSTEMS PUBLICATIONS**

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD