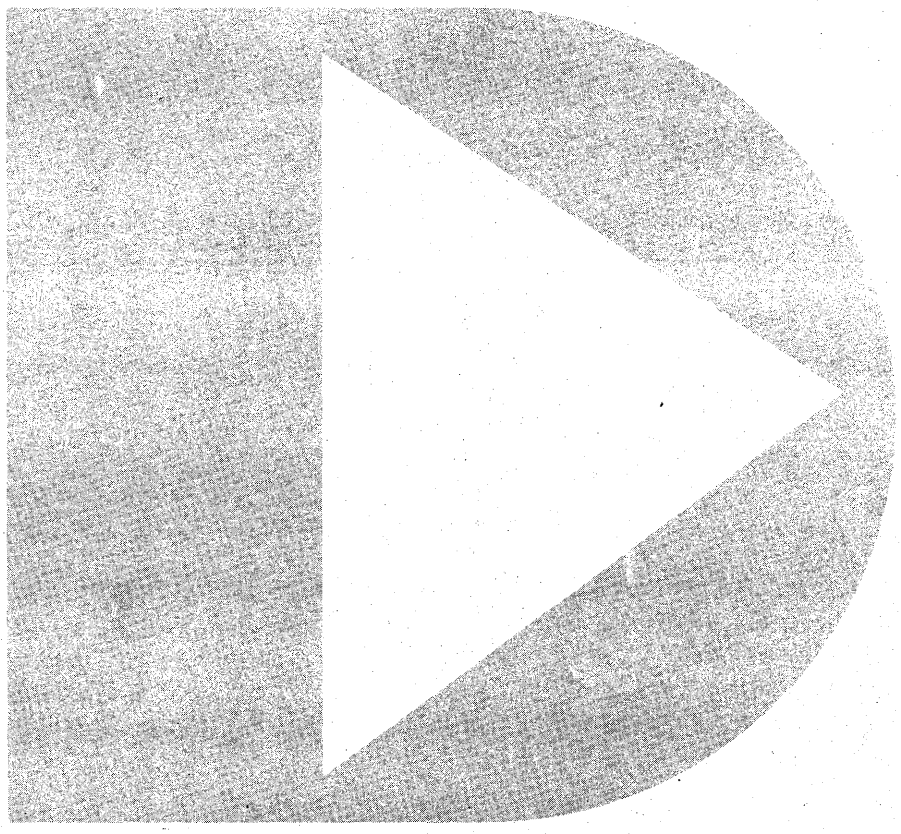
DISK OPERATING SYSTEM SORT

SORT 1.1

SEPTEMBER 5, 1973

Program Users Guide

Datapoint



S O R T

DATAPOINT DISK OPERATING SYSTEM SORT

Datapoint Corporation

September 5, 1973

INTRODUCTION

The Disk Operating System SORT enables any Datapoint Disk user to initiate rapid and sophisticated file sorts directly from a keyboard command simply by incorporating the SORT program within his command library.

Using a multi-train radix sort technique, the Datapoint 2200 achieves speeds comparable with much larger systems. The list of options also compares favorably with much more extensive systems. Nevertheless, since it uses the full dynamic nature of the 2200 Disk Operating System, it is extremely easy to operate.

For more sophisticated uses, SORT may be called from other programs through DOS CHAIN. Using CHAIN also enables complicated sort options to be reduced to a single macro name then callable either from the keyboard or another program by that name. CHAIN also extends the SORT package to operate as a merge, as well.

SORT is available on a single cassette consisting of two programs: SORT/CMD and SORT/OV1. Once cataloged, the SORT is immediately available to run.

TABLE OF CONTENTS

1.0 GENERAL INFORMATION

1.1 Physical requirements

SORT is designed to operate with a Datapoint 2200 model 126, 16k version 2, and a 2200 series 350 disk with one to four drives. It is necessary to have Datapoint DOS resident and SORT cataloged within it.

SORT will optimize its speed through allocation of its working files on the available drives. During this process it attempts to ascertain the availablility of sufficient disk space to achieve the desired sort. The program will abort at this point should the disk space be inadequate.

1.2 Bringing up the system

The two programs comprising sort are distributed on a cassette in the 'CMF' format meaning that they are cataloged on a CTOS tape. This is only a means of distribution and the Cassette Tape Operating System has no other connection with the SORT programs.

To incorporate the two programs from the CTOS tape into the Disk Operating System directory, the following procedure proves least time consuming: By hand, advance the cassette beyond the loader (about 6 inches of actual tape). Bring up the DOS. Place the cassette in the front deck. Type to the DOS IN SORT/CMD.N2. This will bring in the main SORT program. If no error messages are issued, proceed with the next step. Type IN SORT/OV1.N3. SORT is now ready to operate. (Of course, the above procedure could be replaced with the usual 'OUT' the CTOS file and 'IN' the disk file, but it takes much longer).

2.0 SIMPLE SORT CONCEPTS

2.1 What the files look like

All Datapoint systems use a universal text file structure - Databus, Datashare RPG II, Basic, Scribe, Editor, Assembler, Terminal emulators, etc. Therefore, any file generated by or for any of the above may be sorted. It must be on disk, however.

There are two sub-formats the Datapoint file can take: Indexed or Sequential. Sequential files are very free form and have little restriction on their relation to the physical records on the DISK. Indexed files, however, are required to have a fixed relationship of a single 'string' or 'record' of data per disk record. Nevertheless, indexed files can be read sequentially in the identical way that sequential files are read. In fact, both types, when read sequentially, are indistinguishable. Indexed files are used for achieving random access to records. They generally require a great deal more disk space than sequential files for the same amount of data.

When sorting, consider that the result of the sort is not restructuring of the original file. It is a restructured COPY of the original file. The original file is never changed.

Therefore, SORT produces a file which is a sorted version of the original. This gives the user the added opportunity of specifying the type of file to be the output regardless of the input file format (with one restriction - see section 3.4).

2.2 The key options

The KEY of a sort is the FIELD or that part of the record which is to ORDER the sequence of records. For instance, it can be a person's name, state, employee number, amount in debt or any aspect of the data base identifiable by a fixed position in the record based upon the column count from the beginning of the record.

Consider the following record (column count scale below for reference only):

```
Mule, Francis A.        242219 123 BARN    SAN ANTONIO      TX
12345678901234567890123456789012345678901234567890123456789012345678901234567890
```

The name begins in column 1 and goes to 22. The employee number spans columns 24-29. The street address is 31-42. The

city is 43-59. The State is 59-60.

If each person had a record in the file exactly in the above format, SORT could order the sequence of records in the file by any of the above fields. For instance, to get an alphabetical list of the records by name, the KEY would be 1 to 22 (hereafter referred to as 1-22). The KEY for sequencing the file in order of employee number would be 24-29. The key for ordering the records by state then city and then employee number would be 59-60,43-59,24-29.

It should be obvious that any part of the record can be used as a key. It may not be obvious, however, that the larger the key, the slower the sort - it is the case and is approximately proportional.

2.3 How to sort a file

Sorting a file is done from the keyboard of the DOS. All the operator must know is the NAME of the file to be sorted, the name desired for the sorted output file, and the definition of the KEY.

For instance, the keyboard issued command for the above example to sort on the name field (1-22), would be:

SORT EMPLFILE,SORTFILE:1-22

This is assuming that the name of that file was EMPLFILE. It is also the operators decision as to what the resultant sorted file is called. The command could have been: SORT EMPLFILE,EMPSRT:1-22, naming the resultant file EMPSRT. The second file named is where the resultant sort will be placed.

More complicated keys may be stated as well, and the command to list by state and then name would be:
SORT EMPFILE,SORTFILE:59-60,1-22 .

This is simplified sorting. Most systems have source code for a Databus or Assembly language program on file on the disk. Such programs can be sorted, for instance, by label field and produce an interesting look at the dictionary: SORT INFILE,OUTFILE:1-6.

Sorting an assembly or Databus source file by op-code can provide an interesting analysis of the usage of each instruction type:
SORT INFILE,OUTFILE:9-12.

Sort can provide an interesting, informative analytical tool to many other systems than simply the typical business

requirements. These other aspects of SORT's utility become increasingly significant when a sort function is as easily invokable and speedy as the Datapoint SORT system.

3.0 THE OTHER OPTIONS

3.1 Generalized command statement format

The following is a generalized statement format for the Datapoint DOS SORT:

SORT INFILE,OUTFILE<:DRk><;[<*><I><D><L>]<SSS<-EEE>>>

Information contained within a pair of corner brackets <> are optional. Default conditions are listed below. The items within square brackets [] are not order dependent. Typical statements obeying this format are:

(1) SORT INFILE,OUTFILE
(2) SORT INFILE,OUTFILE;1-3,7-20
(3) SORT INFILE,OUTFILE;ID1-3
(4) SORT INFILE,OUTFILE;*IDL7-20

All the above statements will invoke a sort. Each will provide different results. However, notice that in (1) there are no other parameters than the file specifiers. That is because all the specifiable parameters have a given value in case there is no specification for it.

The following list defines the parameters which can be specified:

<:DRk>.....This specifies the drive for the sort key file. This is only a working scratch file needed during the sort. It will go, in default, on drive 0 in a one-drive system, and drive 1 in a multi-drive system. Removing it from the same drive as the input file will speed up the sort.

<*>.....This will cause a debug display of various internal parameters to occur during the sort and is only useful in helping systems programmers understand a difficult-to-explain behavior on the part of SORT. If * is not present, no debug display will occur.

<I>.....Without typing the I, the output file will be SEQUENTIAL no matter what the input file. Only if the input file is an INDEXED file, can the I parameter be included and cause the output file to be indexed.

<D>.....Without typing the D, the collating sequence

order is considered ASCENDING. Including the D parameter will cause the collating sequence to operate in DESCENDING order.

<L>.....Normally the sort transfers the entire records of the input file to the output file. It is possible, not only to transfer part of each record, but to include constant literals in each record as well. Including the L parameter in the list of parameters will cause another question to be asked wherein you may specify the limitations and constants.

<SSS<-EEE>>.....This is the sort key specification. It may be repeated with comma separators. If no key is specified, the system will assume the identical sort to that which would occur if the operator typed 1-10, i.e. the first ten characters. The maximum number of keys is that which can be typed without exceeding the input line.

3.2 Keys-overlapping and in backwards order

The key specification does not need to be forward only. A specification of 17-12 will cause the 6 delimited characters to be a key but in the order of 17,16,15,14,13,12. This is extremely valuable, clearly, in data which has the most significant digit or character last.

Key specifications may also be overlapping: 1-20,30-15 overlaps 15 to 20. When this occurs, the system will optimize the sort and save time over re-sorting on those columns again.

3.3 Ascending and Descending sequences

Changing the collating sequence from ascending to descending is the same as 'reversing' the file, or placing the last first etc. Sorting a telephone directory in ascending sequence on name produces the familiar order. Should it be sorted in descending sequence, then Mr. Zyk would be first and Mr. Aardvark would be last. The order of collation follows the ASCII order when alphabetic, numeric, and punctuation characters can all occur in a column together.

3.4 Input/output file format options

SORT accesses each file sequentially. Due to the techniques used in the Datapoint standard file structure, the sequential reading technique will provide SORT with all of the records in the file whether the file was originally indexed or sequential. Therefore, the file format options only allow specification of the OUTPUT file's format.

If the input file is INDEXED, that is one logical record or string per physical disk record, then you have a choice of output formats. If 'I' is chosen, that is INDEXED, then each output disk record will contain an exact copy of the appropriate input file record. If 'S' is chosen, that is SEQUENTIAL, then the input file, reordered, will be reblocked, space compression imposed, and appear, generally much more compactly, in the output file in sequential format.

If the input file is SEQUENTIAL in its original format, then there is only one choice for the output format. The output file format for a sort on an input file which is sequential MUST be SEQUENTIAL.

3.5 Limited output format option.

In many cases, especially when making reports, directories etc. from the data base, it is not necessary to have the entire record transferred from the input file to the output file during a sort. For instance, an entire personnel data base can be sorted by name to produce an internal company telephone directory. However, it is obvious that all that is needed is the name and telephone number, NOT all the other payroll information. Therefore, SORT permits transferring only that part of the data base desired.

In the same manner that the key of the records is specified by fixed column number, i.e. 1-10 for the first ten characters, the limited output feature specifies the part of the record to be transferred. Should the response 1-10 be given to the limited output format request, only the first ten characters of each record will be transferred to the output file. Also, in the same manner that the key permits multiple discontiguous fields to be specified, the limited output format specifier operates. For instance, 1-10,50-70 would transfer thirty characters from each record of the input file to the output file. The eleventh character in the output record would be the fiftieth character of the input record, etc.

To invoke the limited output format option, the operator includes the 'L' parameter in the specifier list. (see section 3.1). If and only if the L is specified during

the SORT call, will there be a second question asked of the operator on the next line:

LIMITED OUTPUT FILE FORMAT:

This question requires at least one non-trivial field specification or constant (see next paragraph). The number of field and constant specifications is only limited by that which can fit on the keyed in line.

To permit even more utility in report generation, SORT allows inclusion of constants in the output record that did not occur in the input record. For instance, assume that the personnel data base was a full record of approximately 240 characters and that the employees name appears in columns 80 to 110 and his telephone number was in columns 171 to 180. To make a telephone directory in alphabetical order, one could answer the following to the limited file output format request:

80-110,' - ',171-180

Note that this would put out the name followed by four spaces, a hyphen, four more spaces and the number. Any number of input file fields and constants can be placed in the output file up to the limit of the line on which the specification is typed.

Also note that the output file requires proportionally less room than the input file when limited. Often this fact can be applied when the disk file space is nearly exhausted and a sort is required.

3.6 Key file drive number.

There are three file systems associated with a sort. The first is, of course, the input file. The second is the output file. The third is the keyfile system. (The user only uses the output file - the keyfile system is a scratch file used by the system during sorting). There are actually two files which are opened during the sort for the keyfile system. They are *SORTKEY/SYS and *SORTMRG/SYS. These two files can grow to considerable sizes during the sorting procedure since they are proportional to the number of records and the size of the key field.

There are two considerations for the location of the keyfile system. The first is the problem of room. The keyfile must be on a drive with sufficient room to hold it. The second is speed. The greatest increase in speed occurs in removing the keyfile system from the same drive as the input file. Greater speeds can occur if it is, as well, not

on the same drive as the output file. Normally the SORT does a good job of determining the best location of the two keyfile files, and it should not be necessary to specify anything for this. However, under complex circumstances, it may be desirable for the operator to specify the drive number for the keyfile. Should this be the case, the user should type in the <:DRk> specification as indicated in the general command format in section 3.1.

4.0 THE USE OF CHAIN WITH SORT

The reader should first familiarize himself with CHAIN by thoroughly reading the CHAIN User's Guide.

Chain is a system whereby the operator of a Datapoint Disk Operating System may pre-define a procedure sequence of his own programs, system commands and utilities (including keyboard answers to questions requested by these programs) and have them called and sequentially executed by a single name. This is especially powerful when using SORT since there may be a repetitive sequence of routines with complex parameterizations which would make good use of a simplification.

4.1 How to set up a chain file for sort

The author of a chain file needs to remember that ALL questions that the system requests INCLUDING those initiated by the executing programs MUST BE ANSWERED from the chain file just as though they would be typed in from the keyboard.

For instance, the initiation of a sort 'SORT INFILE,OUTFILE;I3-42' could be done through chain. To do this, use the Editor to type in that exact sequence of characters into a file. Note that the file will, in this case, consist of a single line as typed above. The file can be any name, but for purposes of simplifying the explanation, it shall be referred to as CHAINFIL. If CHAINFIL consists of that single line, and if the operator types the command 'CHAIN CHAINFIL' to the DOS, the SORT specified above would be initiated. If the 'L' specification were included in the statement above, then SORT would ask for another line of information. In this case, the file CHAINFIL would have to have two lines in it with the first being the SORT command and the second being the limited output file format specification.

4.2 Naming a repetitive sort procedure

Frequently there are sorts and printouts and other proceedures which occur together and for which a name invoking the procedure would be a great simplification.

For instance, in the telephone directory example above, the process of sorting the file into a limited output file and then listing it on a local printer could be procedurized as follows:

```
SORT EMPFILE,TELFILE;L80-110
80-110,'  - ',171-180
LIST TELFILE;XL
TELEPHONE DIRECTORY FOR XXXXXXXXX CORPORATION
```

Note that there are four statements. The first is the SORT command. The second is the answer to the limited format initiated by the 'L' in the SORT command. The third is the DOS LIST command with the specifiers of 'X' which says 'without line numbers' and the 'L' which, here, means local printer. Then there is a fourth line which the LIST command requests - the heading. This question must also be answered in the chain file. If the above four statements were placed in a file by the editor (or by any other means, for that matter) and then chain were invoked with that file specified, the result would be a sorted telephone directory from the personnel files appearing on the printer.

4.3 Initiating a sort from another program

The chain file (CHAINFIL above) could have been created by any Datapoint system which can write a file. This makes the concept even more powerful since programs can create or modify subsequent procedures of itself, other programs, system commands and utilities. RPG II and Databus 7 can make good use of this.

4.4 Using chain to cause a merge

Consider a situation wherein a system has a master file called 'MASTER' and a file of records to be added, in sequence, to the master file called 'ADDFILE'. To merge these two files in sorted sequence at the end of each day would normally require a sequence of keyed in operations which are somewhat complicated and error prone. CHAIN can cause an effective MERGE and assign it a single name as follows:

```
SAPP MASTER,ADDFILE,MASTER
SORT MASTER,SCRATCH;1-20
KILL MASTER/TXT
NAME SCRATCH/TXT,MASTER/TXT
```

Note that the procedure:
1) appends the ADDFILE to the MASTER file.
2) Sorts the extended MASTER file into a SCRATCH file.
3&4) Renames the SCRATCH file as the new MASTER file. Thus, it is apparent that a merge can be effectively achieved using SORT by using chain to pre-define the procedure.