MAGNETIC PLATTAPPEN

Title:

RC 3600 Data Entry Release 2

SORT PROGRAM

USERs GUIDE

A/S REGNECENTRALEN

RC SYSTEM LIBRARY, FALKONERALLE 1, DK-2000 COPENHAGEN F

RCSL No: 43 - GL 4913

Edition: September 1977

Author: Torben G. Rasmussen

Keywords: MUS, Data Entry, Release 2, Sort, Supervisor Programs, Batch Structures, Sort Parameters, Comparing, Examples, Specifications.

Abstract: The Sort Program for RC 3600 Data Entry Release 2 is developed in order to handle data batches with several record types forming units of various structures. This publication is a guide in structuring Data Entry Batches, which are to be sorted. The sort description parameters are explained and examples of batch structures are shown. The supervisor programs concerned are: RC36-00612, RC36-00613, RC36-00614, RC36-00615

Printed by A/S Regnecentralen, Copenhagen

CONTENTS PAGE

1. INTRODUCTION

The purpose of this manual is to explain the possibilities of the RC 3600 Data Entry Sort programs and to explain the relationship between the format program used to create a batch and the structure of such a batch. The understanding of this relationship is necessary when a format program is to be written as the format program define the structure of the batches to be keyed under control of that format program. Thus, the format program determines the possibilities of the sort programs.

The RC 3600 Data Entry Sort is developed with reference to the fact that a data-entry batch may contain several types of records and that the batch may have a structure. The sort is performed by activation of the Supervisor Program SORT. SORT consists of 4 programs SORT, SORT1, SORT2 and SORT3, which is automatically activated in the correct sequence when SORT is started.

Supervisor program SORT is handling preparations like setting up all parameters, checking of parameter values, creation of destination batch, and workfile and parameter file manipulation and reading the input batch and creating the special records (only the sort keys) to be sorted.

Supervisor program SORT1 is doing the actual sort, controlled by the parameters having been set up. The program produces a number of sorted runs.

Supervisor program SORT2 is merging the sorted runs in the workarea, and writes the sorted file which contains records with pointers to and recordnumbers in the input batch.

Supervisor program SORT3 generates the final sorted output batch by means of the sorted file generated in SORT2 and the input batch.

By the term structure mentioned in the first paragraph we will understand a relation between different types of records. To the sort programs a batch is simply a sequence of records. For this reason the structure of the batch must be explicitly defined to the sort programs, and sort criteria for each type of record must also be defined.

A batch holding keyed invoices is an example of a structured batch. Each invoice consists of e.g. one record holding customer information, one or more records each holding article information, and one record holding total price.

In this example the structure comes from the fact that a sequence of article records and one total price record is belonging to a specific customer record, and the sort programs must therefore know this structure in order to avoid mixing of article records belonging to different customer records. The batch is in fact holding a sequence of invoices, and each invoice is holding a sequence of article lines. In this example the sort programs enable the user to sort the article lines on each invoice, or to sort the invoices, or to do both at the same time.

In section 2 the relation between a format program and the possible structures of the batch will be explained. The idea of associating level numbers and sequence numbers with each type of record is introduced.

Section 3 explains the necessary sort program parameters, and in section 4 the criteria used to select a winner among two records are stated.

Section 5 contains a number of examples of how a certain batch could be sorted, i.e. the consequences of various sets of sort specifications are shown.

Section 6 is detailed list of the possibilities and restrictions in the sort programs. (See also DATA ENTRY RELEASE 2 SORT - OPERATING GUIDE - RCSL:43-GL 4912).

2. THE BATCH STRUCTURES

In this section the relation between the format program and the batch structure is explained. The various possible batch structures are described, and the possible ways in which the sort programs may handle the various structures are explained.

2.1 Format Program and Batch Structure

A batch is a disc file in which the keyed data is stored. The data stored may also be called a sequence of complete forms.

Each form is a unit, which is normally composed of several parts. An invoice is a typical form, composed of a head consisting of a customer name, date, conditions of payment, customer number, invoice number, one or more lines of article descriptions each consisting of article name, article number, price per item, number of items, total price for this number of items, and a total consisting of total price to be payed.

Each of these parts is a record in the batch, but the three record types are composed of different information, and they therefore differ in their internal structure.

When keying a batch the keyed information is controlled by a specific format program which corresponds to the kind of unit to be registrated.

The format program consists of a subformat program for each part of the unit. The subformat programs are activated in a sequence corresponding to the structure of the unit. Units are registrated in the batch in a sequence selected by the keying operator, and the records which form the unit in the batch are written in a sequence corresponding to the structure of the unit. In fig. 1 is shown one unit, which represents an invoice.

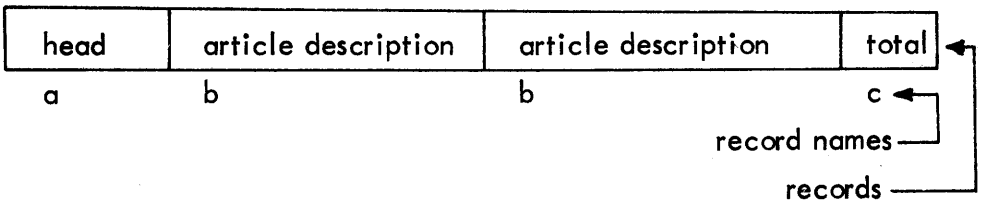


Fig. 1. An invoice composed of 4 records.

Each record written has a name which corresponds to the subformat program that controlled the keying of that particular record. In fig. 1 the record names are a, b and c. Each record name specifies one particular record type and the batch structure can then be defined by expressing a relation between pairs of record types. In fig. 5, section 2.3, is shown an example of a relation between two record types. From that figure can be seen that a unit can be regarded as a block, and the relation between record types can therefore be expressed by associating a block level or level number with each of the two record types.

The record types b and c are situated in inner blocks to the respective record types a. To define the relations between the three record types it is then sufficient to define record type a as belonging to level 1, to define record types b and c as belonging to level 2, and to define sequence number for record type b as 1 and to define sequence number for record type c as 2. The definitions of level and sequence numbers may be found in section 2.3.

2.2. Unstructured Batches.

Unstructured batches are characterized by the fact that they contain records as a sequence of records with no relations between the records. Unstructured batches may be divided into simple unstructured batches and complex unstructured batches.

2.2.1 Simple Unstructured Batches.

In this type of batch only one type of record occurs. This is the type of file on which sort programs normally work. The description of sort criteria is global to the file and consists of a set of field specifications to be used by the sort programs. If the sort criteria define the file to be sorted in one alphabetic field in ascending order the sorted file is characterized by the fact that for a given record N the sortfield is greater than or equal to the sortfield of record N-1, where $N \in [2, max]$.

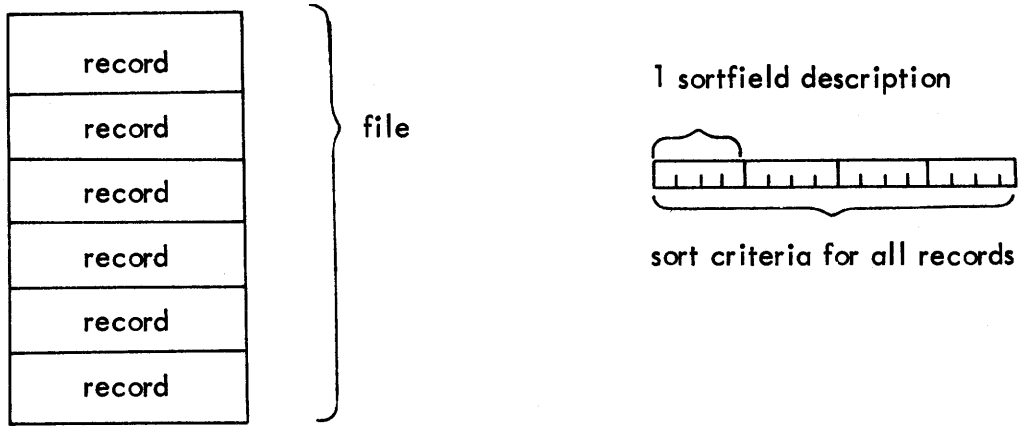


Fig. 2. A file holding a sequence of records, all with the same structure. Each sortfield description holds information about startindex, fieldlength, fieldtype and order of sorting.

Records with equal sort fields will occur in the sorted batch in the same mutual sequence as in the unsorted batch.

2.2.2 Complex Unstructured Batches

In this type of batch several types of records may occur. The sort programs are capable of sorting such a batch by associating a different set of criteria with each type of record.

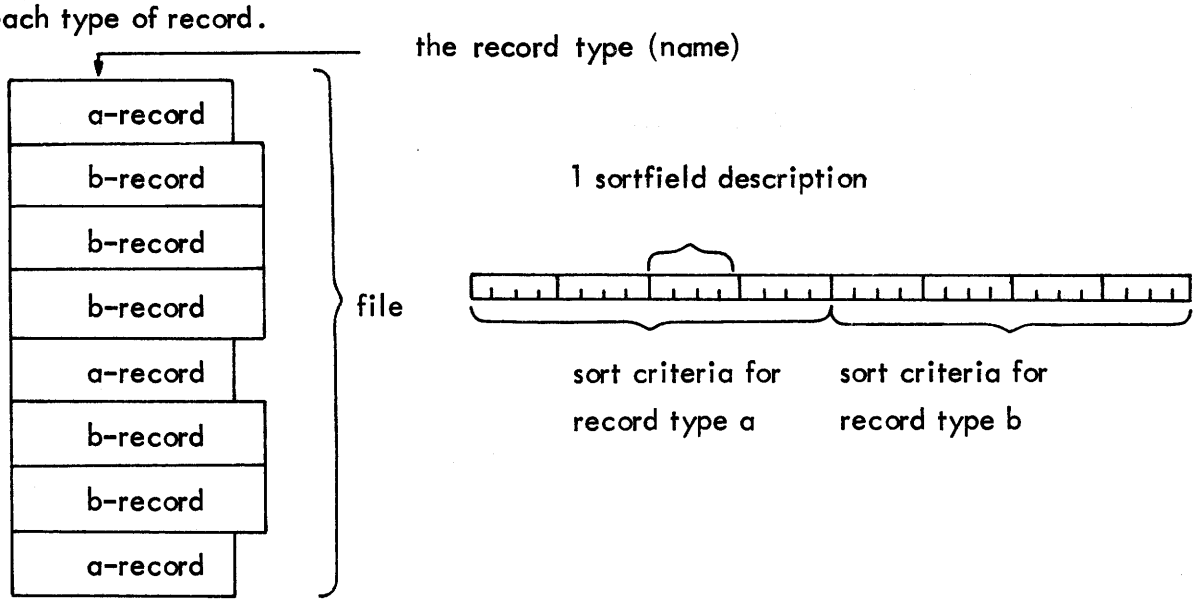


Fig. 3. A file holding a sequence of records. Two different record types with each their structure. One set of sort criteria for each type of record.

It should be noted that when setting up the sort criteria for data-entry batches, one should think in terms of field numbers rather than specific byte positions within the records. Thus, the sort programs also include the possibility of sorting fields of different lengths.

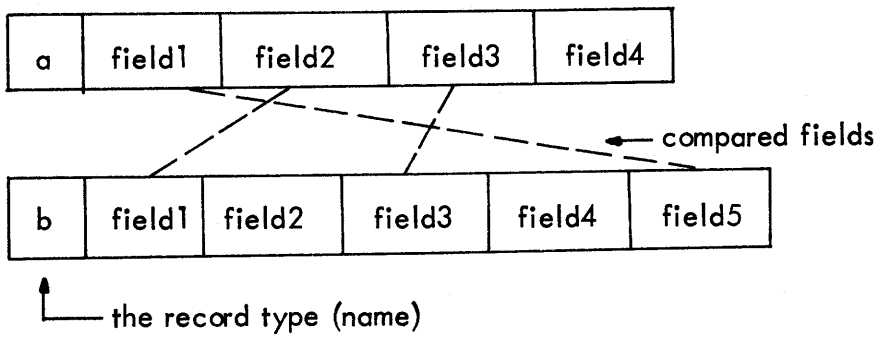


Fig. 4. Example of comparing of two different record types. Field 2 of record type a is compared to field 1 of record type b, field 3 of record type a is compared to field 3 of record type b and field 1 of record type a is compared to field 5 of record type b.

2.3 Structured Batches

Batches of this type contain several record types, each with its associated set of sort criteria, as for the complex unstructured batches, but in addition, each record type should be thought of as belonging to a certain level, and each record type has a sequence number to be used within the level to which it belongs.

Definition. The level number indicates how the record should be placed in the structured batch in relation to all other records in the batch.

Definition. The sequence number is just an extra sort criterion for any record. It may, or it may not, be used. This sort criterion preceedes the user-defined sortfields, i.e. if the sequence numbers of the records are identical they have no effect on the sort.

The structured batch may be thought of as a sequence of blocks, each block consisting of a sequence of records. Each of the records in a block may itself be the beginning of a new inner block.

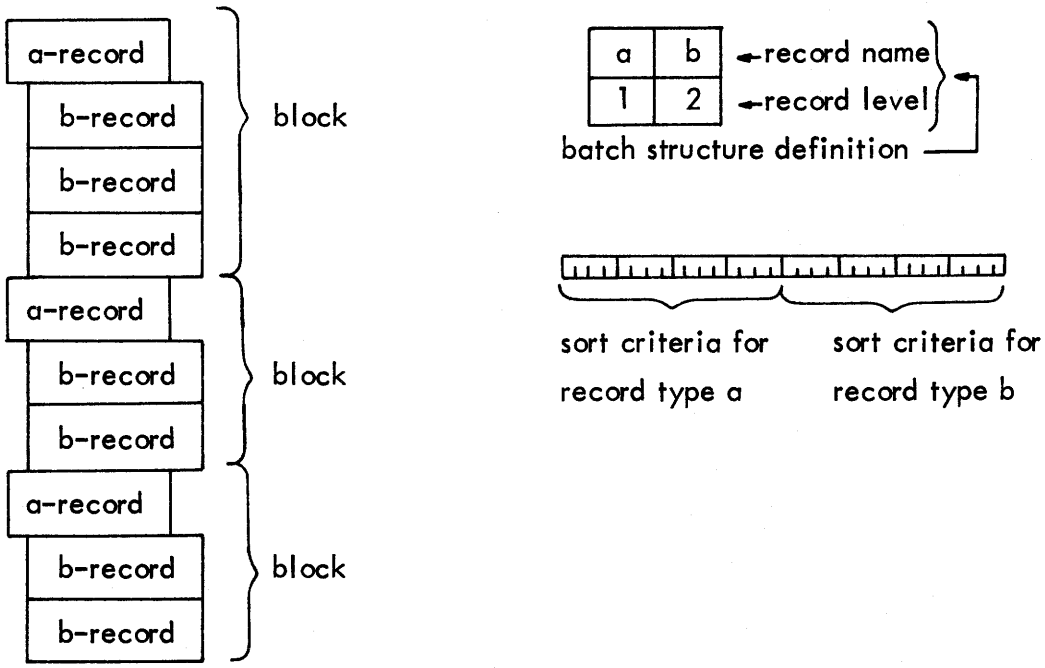


Fig. 5. In this example b-records are defined as belonging to the inner blocks of the a-records (b-record level greater than a-record level in the batch structure definition).

The sort programs are capable of sorting such batches by sorting the records within each innermost block and then sorting blocks rather than records, beginning with the innermost block level and ending up with the outermost block level.

The sort criteria of a block are then the sort criteria associated with the record which is the start of the block.

The outermost level is the lowest level defined by the user. The innermost block level is the highest level defined by the user. User defined levels between these two levels do not necessarily have to be present, as can be seen in section 5.

Definition. The start of a block is any record belonging to a level N followed by one or more records belonging to a level greater than N (e.g. N+1, N+7), and the block ending with the last of the records belonging to a level greater than that N.

In general, each record may be considered the start of an inner block which may, or may not, be empty.

When designing batch structures and subformat programs for each type of record to the Data Entry System it should be remembered, that a block can only be sorted on the sort criteria associated with the subformat which is forming the start of the block.

3. SORT PROGRAM PARAMETERS

The sort programs must be informed of the input batch, the output batch, the structure of the input batch, the record types in the input batch, and the fields to be used as sort keys.

This is done by transferring parameters to the SORT program, when it is activated from the Supervisor key-station.

Call:

SORT <INPUTBATCH> <OUTPUTBATCH> <PARAMBATCH> (<WORKAREASIZE> <RELEASE>)

<INPUTBATCH> : is the name of the batch to be sorted.

<OUTPUTBATCH> : is the name of the batch in which the sorted data are to be written. This batch must not exist at call-time, except in the case where the inputbatch is also used for the sorted output. An output batch is marked as sorted, this prevents editing, keying and rekeying of the sorted batch. For the sake of security it is recommended not to use the same batch for input and output, but the user is allowed to do so at his own responsibility.

<PARAMBATCH> : is the name of the batch containing information about the structure of the inputbatch, the record types and the fields to be used as sort-keys. This batch must be keyed under control of the standard-format SORTF. (see section 3.1).

<WORKAREASIZE> : this parameter should usually not be used in the call. The SORT-program calculates the size needed for workarea but in some "rare worst-case situations" it might calculate too little. In this case the sorting will be terminated with the message:
WORKAREA TOO SMALL NNNN
where NNNN is the calculated number of sectors of workarea.

Now SORT must be started again and this parameter must be set to a value greater than NNNN (max. NNNN = 1600).

<RELEASE> : if inputback contains invalid records.

3.1 SORT Program Parameter Batch

The parameter batch is a batch keyed under control of the standard format SORTF. The parameter batch contains information about the structure of the input batch, the record types in the input batch, and the fields to be used as sort keys.

If the user is working with several batch structures, a parameter batch may be installed for each batch structure. Similarly, if a batch has to be sorted in different ways from time to time, a parameter batch for each sort specification may be installed.

The information is for each record type (=subformat) NAME, LEVEL and SEQUENCE, where NAME is the subformat name. (LEVEL and SEQUENCE are described in section 2). LEVEL and SEQUENCE are both in the range 1 - 255. Only those record types specified in the parameter batch will be written in the output batch, but as a record specification might be omitted by mistake, the sort program will display any recordname, which has not been specified.

For each sort-criterie (=field) in each record FIELD, TYPE and ORDER must be specified. FIELD is the fieldnumber (1 - 255) in the record. TYPE is alphanumeric (A) or numeric (N). With the term alphanumeric is meant that the fields are compared by using the ASCII character values, and the fields are compared from left to right. With the term numeric is meant unsigned numeric fields. These fields are compared by using their binary values. ORDER is either ascending (A) or descending (D).

The maximum number of fields to be used as sortcriterie in a record is 30. The maximum length of the fields used as sort-criterie is 480 bytes. This means that a record can have for example 6 fields of the length 80 bytes each or 30 fields of the length 16 bytes each as sort-criterie.

4. RULES FOR SELECTING A WINNER

When defining a batch structure, the following rules for selecting a winner should be observed.

4.1 Records with Different Level

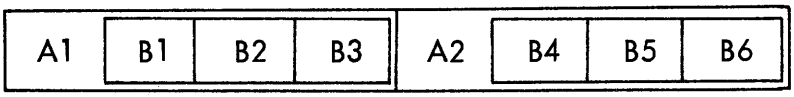


Fig. 6. Example of record types with different levels. The B-records are in inner blocks of the A-records.

The example in fig. 6 shows two types of records, A and B. A is on level 1 and B is on level 2. In this case the B-records are said to form inner blocks to the A-records.

When comparing records with different level number the lowest level i.e. the outermost block, is selected as winner.

Records in one inner block are never mixed with records from other inner blocks, i.e. B1 B2 B3 are sorted, B4 B5 B6 are sorted, but B1 B2 B3 will always belong to A1, and B4 B5 B6 will allways bel ong to A2.

When the records have the same level number, then no winner can be selected from the level numbers of the records. The sort programs will then use the sort criteria associated with each record (type) to find a winner.

4.2 Record Sequence

The following rules for selecting a winner all refer to comparisons of records having the same level number.

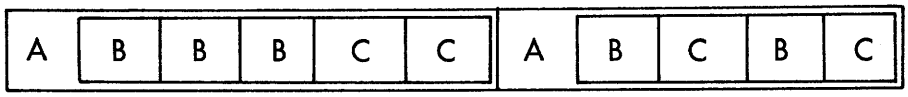


Fig. 7. An example of two record types in same inner block.

The example shows three types of records, A, B and C. A is on level 1, B and C are on level 2, C having sequence number 1 and B having sequence number 2.

When comparing records with the same level number the record with the lowest sequence number is selected as winner (the sequence number preceedes all the user-defined sort criteria). In fact, the sequence number may be considered as the first sort criterion to be used in the comparison. The records shown will thus appear in the following order after sorting of the inner blocks:

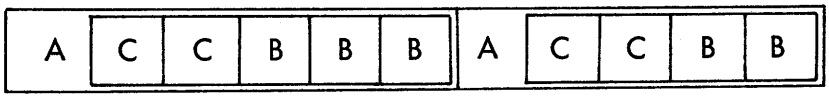


Fig. 8. The B- and C-records in the order determined by their sequence numbers.

When the records have the same sequence number (within the same level) no winner is selected, and the comparing will continue using the sort field criteria.

4.3. Record Fields.

When the batch structure involves comparison of different record types bearing the same sequence number within the same level, it is the users responsibility that the fields to be compared are of the same type and order.

However, conflicts are handled in the following way:

If the fields are of different types they will be handled as alphanumeric fields.

If it is indicated that the fields are to be handled in a different order, the records will be arranged in ascending order.

Finally, if the two records differ in number of sort fields and no field has selected a winner, the record with the fewer fields will be selected as winner.

4.3.1 Alphanumeric Compare

If the two fields to be compared are of different lengths, the shorter field will be regarded as if it was succeeded by a number of spaces giving it the same length as the longer field.

The term alphanumeric means a sequence of ASCII characters, and the sort program does not check the legality of a given character value.

The comparison is a byte-by-byte compare repeated until a difference is found, in which case a winner is pointed out according to the specified order (ascending/descending), or until the end of the fields is reached, in which case no winner is selected, and the comparison will continue with the next set of fields.

When the fields are compared in this way the ASCII character values of the characters are used during the comparing.

If the fields are the last in the sort criteria specification then if the fields have the same value, the records are ordered in the same sequence as in the input file.

4.3.2 Numeric Compare

If the two fields to be compared are of different lengths, then the shorter field will be regarded as if it was preceeded by a number of zeroes giving it the same length as the longest field.

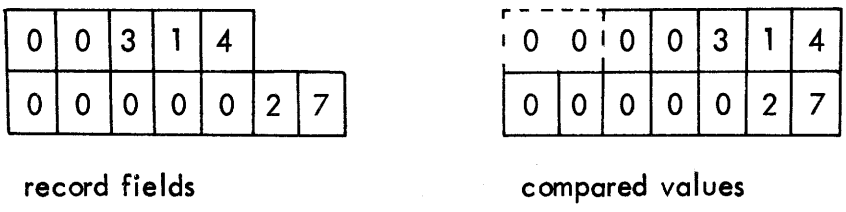


Fig. 9. Example of two fields with different lengths.

The term numeric means a sequence of ASCII characters in the way that a minus is regarded as a zero and a sign information. The characters U, J, K, L, M, N, O, P, Q, R are regarded as numeric characters (0 - 9 plus a sign information), and other characters e.g. asterisks, spaces, are regarded as zeroes.

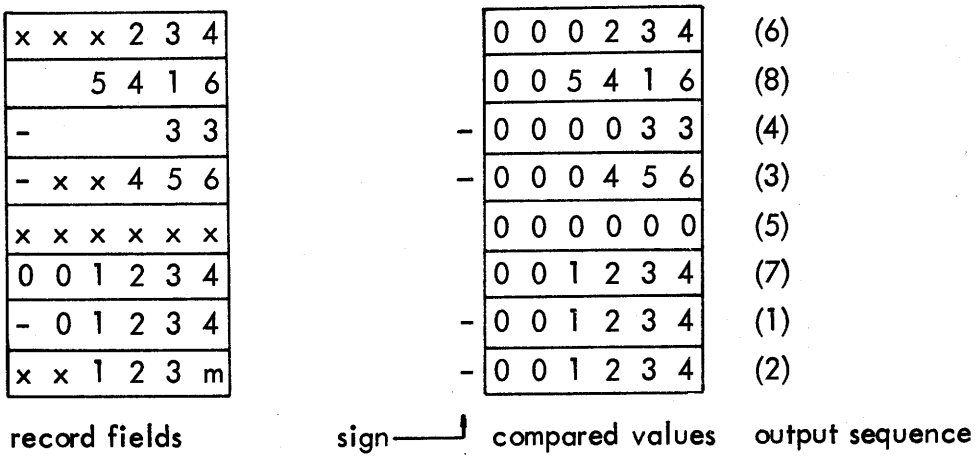


Fig. 10. Examples of contents of fields and how they are handled. In the column to the right is shown the sequence in which the fields would be ordered (ascending).

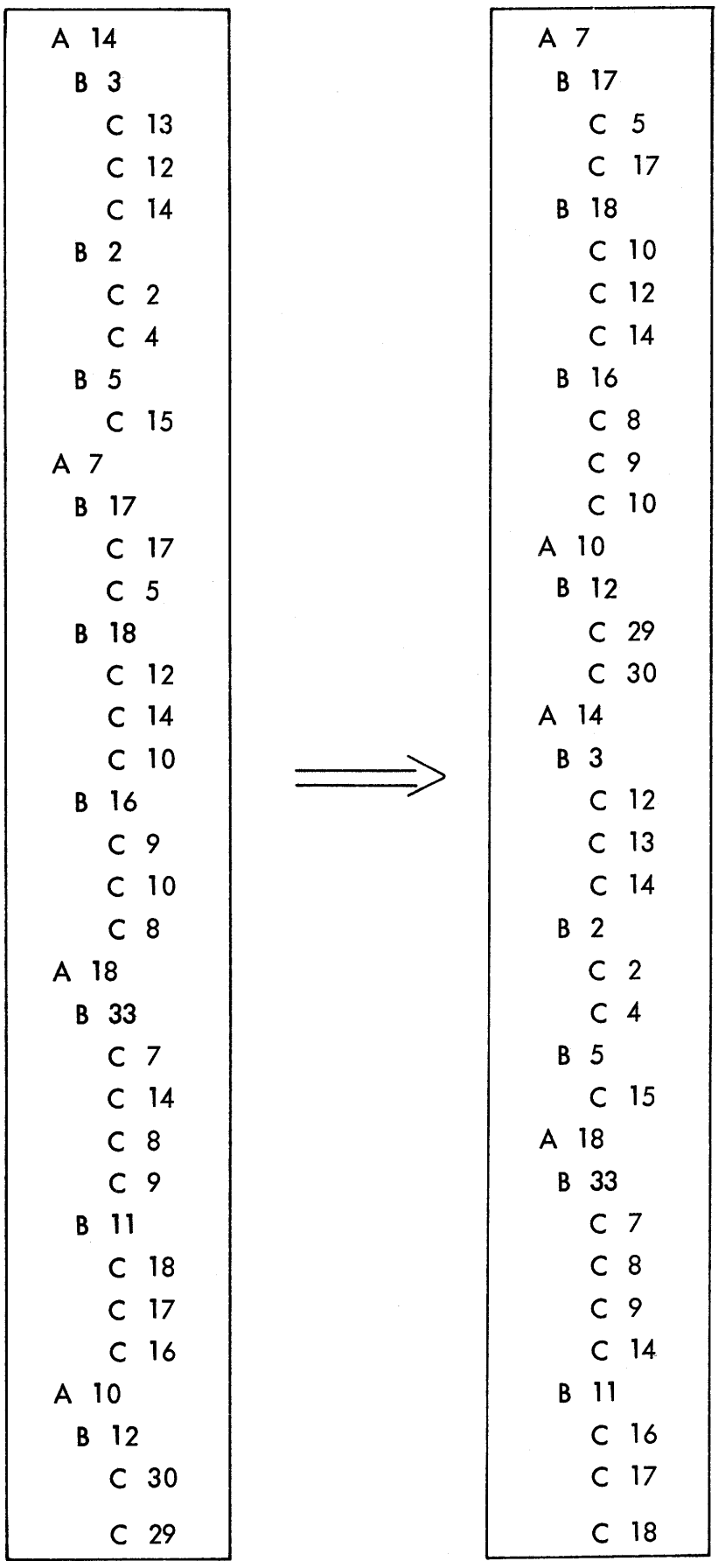
The comparison is a byte-by-byte compare running till the end of the fields. If the numeric values differ or the signs of the fields differ (or both) then a winner is selected. Otherwise comparison with the next set of fields will commence.

If the fields are the last in the sort criteria specification then if the fields have the same numeric value and same sign, the records are ordered in the same sequence as in the input file.

4.4 Record Serial Number

It has been mentioned in the previous sections that when no winner is found from the field comparisons then the records are ordered as in the input file. This is obtained by associating a record serial number i.e. the record number in the input batch, with each record. When two records have same level and sequence numbers, and the contents of the fields in the two records are identical then this record serial number as the last sort criterion of any record, and it is added automatically by the sort programs, and specified as ascending order.

In section 3.3 is mentioned that only those record types specified to the sort program will be written in the output batch. By means of the record serial number it is possible to let any record type take part in the sort without actually being sorted, simply by defining the record type name, level and sequence number, but without specifying sortfields. In fig. 11 is shown a batch holding the record types A, B and C, but only A and C are sorted. Fig. 11 shows the batch before and after sorting.



the input batch

the output batch

Fig. 11. Three record types. Only A- and C-records are sorted.

5. HOW A BATCH COULD BE SORTED

In this section a batch containing three types of records is shown. The record names and the sort keys are created at random i.e. the batch is used only to demonstrate the importance of defining the correct structure of a batch to the sort programs.

Three different definitions of the batch structure are used, and the consequences of each choice are shown.

In the examples each record is represented by a letter (denoting the record type name) and a number which is used as the sort key. Different margins are used to illustrate the block level to which each record belongs.

5.1 Sort Example 1

Structure: Record type A on level 1, sequence 1; Record type B on level 2, sequence 1; Record type C on level 3, sequence 1. The records are sorted on one field (the sort key) in ascending order.

A65	C13	C76
B55	C77	C48
B91	C50	C6
C93	C90	C43
C37	C0	A74
C65	B89	B67
A3	C10	B82
B59	B13	B71
B79	A2	B80
B82	C15	B87
C59	A56	B91
C46	B42	A20
B98	B25	B68
A75	C20	B19
B48	B5	C12
C76	C14	C63
A60	C41	C2
B67	C44	C61
C98	C99	B52
C65	A39	B21
C69	B68	C55
C55	B42	B23
A14	C91	C2
C39	C99	C16
A45	B23	C50
C15	C66	B21
C80	B23	C7
A47	B32	A47
B71	B62	A91
B53	B33	

⟹

A2	B23	C41
C15	B32	C44
A3	B33	C99
B59	C6	B25
B79	C43	C20
B82	C48	B42
C46	C76	A60
C59	B42	B67
B98	C91	C55
A14	C99	C65
C39	B62	C69
A20	B68	C98
B19	A45	A65
C2	C15	B55
C12	C80	B91
C61	A47	C37
C63	B13	C65
B21	B53	C93
C55	C0	A74
B21	C13	B67
C7	C50	B71
B23	C77	B80
C2	C90	B82
C16	B71	B87
C50	B89	B91
B52	C10	A75
B68	A47	B48
A39	A56	C76
B23	B5	A91
C66	C14	

5.2 Sort Example 2

Structure: Record type A on level 1, sequence 1; Record type B on level 2, sequence 1; Record type C on level 2, sequence 2. The records are sorted on one field (the sort key) in ascending order.

A65	C13	C76
B55	C77	C48
B91	C50	C6
C93	C90	C43
C37	C0	A74
C65	B89	B67
A3	C10	B82
B59	B13	B71
B79	A2	B80
B82	C15	B87
C59	A56	B91
C46	B42	A20
B98	B25	B68
A75	C20	B19
B48	B5	C12
C76	C14	C63
A60	C41	C2
B67	C44	C61
C98	C99	B52
C65	A39	B21
C69	B68	C55
C55	B42	B23
A14	C91	C2
C39	C99	C16
A45	B23	C50
C15	C66	B21
C80	B23	C7
A47	B32	A47
B71	B62	A91
B53	B33	

⟹

A2	B32	B42
C15	B33	C14
A3	B42	C20
B59	B62	C41
B79	B68	C44
B82	C6	C99
B98	C43	A60
C46	C48	B67
C59	C66	C55
A14	C76	C65
C39	C91	C69
A20	C99	C98
B19	A45	A65
B21	C15	B55
B21	C80	B91
B23	A47	C37
B52	B13	C65
B68	B53	C93
C2	B71	A74
C2	B89	B67
C7	C0	B71
C12	C10	B80
C16	C13	B82
C50	C50	B87
C55	C77	B91
C61	C90	A75
C63	A47	B48
A39	A56	C76
B23	B5	A91
B23	B25	

5.3 Sort Example 3

Structure: Record type A on level 1, sequence 1; Record type B on level 1, sequence 1; Record type C on level 1, sequence 1. The records are sorted on one field (the sort key) in ascending order.

A65	C13	C76
B55	C77	C48
B91	C50	C6
C93	C90	C43
C37	C0	A74
C65	B89	B67
A3	C10	B82
B59	B13	B71
B79	A2	B80
B82	C15	B87
C59	A56	B91
C46	B42	A20
B98	B25	B68
A75	C20	B19
B48	B5	C12
C76	C14	C63
A60	C41	C2
B67	C44	C61
C98	C99	B52
C65	A39	B21
C69	B68	C55
C55	B42	B23
A14	C91	C2
C39	C99	C16
A45	B23	C50
C15	C66	B21
C80	B23	C7
A47	B32	A47
B71	B62	A91
B53	B33	

C0	A39	B67
A2	C41	B67
C2	B42	B68
C2	B42	B68
A3	C43	C69
B5	C44	B71
C6	A45	B71
C7	C46	A74
C10	A47	A75
C12	A47	C76
C13	B48	C76
B13	C48	C77
A14	C50	B79
C14	C50	C80
C15	B52	B80
C15	B53	B82
C16	B55	B82
B19	C55	B87
C20	C55	B89
A20	A56	C90
B21	B59	B91
B21	C59	C91
B23	A60	B91
B23	C61	A91
B23	B62	C93
B25	C63	B98
B32	A65	C98
B33	C65	C99
C37	C65	C99
C39	C66	

6. SORT PROGRAM SPECIFICATIONS

6.1 Disc and Storage Requirements

The Supervisor Programs SORT, SORT1, SORT2 and SORT3 take up 65 sectors on the disc, and they require 8 K bytes of core storage each when they are loaded.

6.2 Workarea

The workarea is a discfile created at runtime. The size is from 20 to 1600 sectors and is calculated by the program. The size depends on the structure of the input batch, number of records in the input batch and maximum length of sort-keys in one record.

6.3 Parameter Batch

The user may have as many parameter batches installed as he likes. The size of a parameter batch is 2 sectors.

6.4 Batch Size and Batch Head

The size of the input batch must not exceed 761 sectors. The output batch is created with the same size as the input batch. The batch-head of the input batch is copied to the output batch and the output batch status is set to sorted, which means that editing, keying and rekeying on the output batch is impossible.

6.5 Record and Sortcriteria Size

Maximum record size is 20000 bytes of which max. 480 bytes can be used as sort-keys. Maximum number of fields in a record is 255 of which max. 30 can be used as sort-keys.

The area holding the sort-criteria is 200 bytes. Each record type definition requires 10 bytes, and each sortfield definition requires 6 bytes.