QSORT™

USER'S MANUAL

Structured Systems Group
INCORPORATED
5208 CLAREMONT AVENUE
OAKLAND, CALIFORNIA 94618
(415) 547-1567

Q S O R T

U S E R ' S M A N U A L

GENERAL PURPOSE
SORT/MERGE PACKAGE
FOR DATA PROCESSING APPLICATIONS
IN A CP/M ENVIRONMENT

STRUCTURED SYSTEMS GROUP
OAKLAND, CA. 94618

Q S O R T

G E N E R A L I N F O R M A T I O N

This manual assumes that the reader has a running floppy disk based computer system that uses the CP/M operating system. It further assumes that the user has read the CP/M documentation manuals and has a working knowledge of data processing fundamentals. It is strongly suggested that the user have a copy of either BASIC-E or CBASIC as an aid in building parameter files.

QSORT is a CP/M compatible sort/merge utility program designed for use in data processing applications. It will sort up to a full diskette of data Many user controllable options make QSORT a versatile tool.

QSORT will attempt to fit the entire input file into the available storage space. If the file is small enough (or memory large enough; See Appendix B), the entire file will be sorted internally and written to the output file upon completion.

If a file to be sorted exceeds the size of available RAM the QSORT program will automatically create temporary workfiles on diskette to hold overflow data. Up to 20 of these files will be created as needed. When the entire input file has been processed the MERGE portion of QSORT will combine the data on all workfiles onto the single output file specified. All workfiles are erased at the end of the QSORT program.

QSORT is initiated by entering the transient name followed by a single required parameter in the form of a filename with or without the optional drive reference. For example:

```
A>QSORT B:PARMS
        or
A>QSORT PARMS
```

The QSORT program will expect to find all control information stored on the file whose name is specified on the command-line (in the example, 'PARMS') and which has an assumed file type of 'SRT'. The parameter file must be built before execution of the sort program. This can be accomplished by the use of DDT, ASM, QPARM or any user program. There may be many parameter files on line during a sort but only the one specified will be used.

Each parameter file must contain all information for the control of the QSORT program. The format and contents of the parameter file will be described in detail in the Parameter Field Contents section. What follows is a brief overview of the parameters.

Parameter information falls into one of three general categories:

1. FILE INFORMATION
2. KEY INFORMATION
3. CONTROL INFORMATION

FILE INFORMATION:

The drive, filename and filetype of the input and output files must be specified in this section. The drive reference may indicate the currently-logged drive, A or B. The filename and filetype of the input and output files must be in the form of a CP/M unambiguous file reference. The input file will be assumed to be a file of fixed length records. The records do NOT have to be separated by a carraige-return/line-feed sequence or any other record delineator. All records in the file must be the same length as that specified in the parameter file. The output file will be created with the same format as the input file.

KEY INFORMATION:

The data may be sorted on from one to five separate key fields as needed. Keys are specified by displacement and length in bytes. Each key may be specified as either ascending or descending independently from any other key. One of two sorting orders, numeric and alpha, are available independently on each key. Numeric order is normal binary sequence and should be used on all numeric, single-case alpha or non-ASCII data. The alpha sorting sequence is used for sorting text that contains a mixture of upper and lower case alpha characters.

All keys specified will be independently evaluated in descending order of importance. It is the user's responsibility to assure that the specified keys do not overlap other keys.

CONTROL INFORMATION:

The QSORT program must have other information to aid in its control. This includes the drive where intermediate work files are to be placed if external storage is required. If the file to be sorted is to be output on a separate diskette from both the disk containing the sort program and the input disk, the program must be instructed to stop processing immediately before the output file is written, issue a message to the console and allow the operator to change diskettes. This is used where a file is too large for both the sorted and unsorted versions to occupy the same diskette. When the sort program creates an output file, it will normally first erase any existing file of the same name. It is possible to request that a previous version of the output file be backed up according to CP/M conventions, ie., rename it to filetype 'BAK'. If requested, after execution, several statistical messages are printed to the system console.

THE QPARM PROGRAM

Distributed with the QSORT program package is a program written in BASIC-E that will interactively construct a parameter file. The program must be compiled by the BASIC-E compiler and then executed by the BASIC-E interpreter program. When executed, the program will request items of infor-

mation from the user in a conversational mode and build the parameter file. In use, the QPARM program will greatly ease the burden of making parameter files. Note that the QPARM program is upward compatible to CBASIC format for use in commercial environments. A more detailed description of the QPARM program along with complete instructions can be found in Appendix C.

PARAMETER FILE FIELDS

The following chart identifies on a byte-by-byte basis the position and number of all data that must be included in a parameter file. In this section each field will be explained in detail. The FIELD NUMBER should be used to cross-reference between this table and the text. Note that all field contents are shown in hexadecimal format.

FIELD NO.	FIELD NAME	START	END	# OF BYTES
1	not used	1	1	1
2	input file drive	2	2	1
3	input file name	3	10	8
4	input file type	11	13	3
5	output file drive	14	14	1
6	output file name	15	22	8
7	output file type	23	25	3
8	not used	26	26	1
9	record length	27	27	1
10	not used	28	32	5
11	backup override flag	33	33	1
12	disk change flag	34	34	1
13	console output flag	35	35	1
14	work file drive	36	36	1
15	key #1 start	37	37	1
16	key #1 length	38	38	1
17	key #1 ascend/descend	39	39	1
18	key #1 alpha/numeric	40	40	1
19	key #2 start	41	41	1
20	key #2 length	42	42	1
21	key #2 ascend/descend	43	43	1
22	key #2 alpha/numeric	44	44	1
23	key #3 start	45	45	1
24	key #3 length	46	46	1
25	key #3 ascend/descend	47	47	1
26	key #3 alpha/numeric	48	48	1
27	key #4 start	49	49	1
28	key #4 length	50	50	1
29	key #4 ascend/descend	51	51	1
30	key #4 alpha/numeric	52	52	1
31	key #5 start	53	53	1
32	key #5 length	54	54	1
33	key #5 ascend/descend	55	55	1
34	key #5 alpha/numeric	56	56	1
35	not used	57	64	8

DISKCHANGE: (Field 12)

The diskchange operation is necessary when data files are very large. The operation allows the input file and output file to reside on separate diskettes that are held in the same drive. The diskette containing the workfiles must remain on-line in the same drive at all times.

When the diskchange parameter is set in the parameter record the sequence of operation is as follows:

[User action shown in brackets]

Input file is read
Input file is sorted
Diskchange request is issued to console
Program temporarily stops execution
[User changes diskette]
[User types return key]
Program opens output file
Program writes output file

If the data file is large enough to cause intermediate workfiles to be created, they will be sorted and created before the diskchange request is issued.

The format of the diskchange message is:

PLACE DESIRED OUTPUT DISKETTE ON DRIVE a THEN TYPE RETURN

Where a is the output file drive (A or B).

RESTRICTIONS TO DISKCHANGE:

If no intermediate workfiles are created it is possible to change either diskette in a two-drive system. If the sorted file is large enough to cause intermediate files to be opened, the drive specified for output must be different from the drive specified for work files if the diskchange option is requested.

EXAMPLES:

Diskette 1 contains system, QSORT and work area
Diskette 2 contains input file
Diskette 3 will contain sorted output

Step 1) Insert #1 in drive A, #2 in drive B
Step 2) Reboot, initiate QSORT program
Wait for change message
Step 3) Remove diskette #2 from drive B
Step 4) Insert diskette #3 in drive B
Step 5) Type return key
Wait for program completion

Diskette #1 contains system, QSORT
Diskette #2 contains input file and work area
Diskette #3 will contain output file

Step 1) Insert #1 in drive A, #2 in drive B
Step 2) Reboot, initiate QSORT program
Wait for change message
Step 3) Remove diskette #1 from drive A
Step 4) Insert #3 in drive A
Step 5) Type return key
Wait for program completion

BACKUP FLAG: (Field 11)

If this flag is not set, any file on the output disk drive with the same name as the output file will be replaced by the sorted output file. If the Backup flag is set, however, the previous version (if any) will be renamed to filetype 'BAK'. If a file of type 'BAK' already exists, it will be replaced. Note that if the names and drives of both input and output files are identical, the unsorted input file will be renamed to type 'BAK' if the flag is set, but it will be lost if the flag is not set.

CONSOLE OUTPUT FLAG: (Field 13)

If this flag is set several statistical messages will be printed at the end of execution. If the flag is off, all console output will be suppressed with the exception of the logon message and the sort completion message at EOJ. This flag will NOT suppress error messages. The format of the QSORT logon message is:

QSORT VER:n.n

Where n.n is the current version number.

The format of the QSORT end of job message is:

SORT COMPLETED

Statistical messages that are printed are of the form:

```
INPUT RECORDS READ:          2544
WORKFILES USED:                 4
OUTPUT RECORDS WRITTEN:      2544
BYTES IN BUFFER:            23698
BYTES PER LOGICAL RECORD:      35
KEYS SPECIFIED:                 3
```

'INPUT RECORDS READ' is the total number of logical records that were read off the input file.

'WORKFILES USED' is the total number of temporary files that were open-

ed during execution to hold the intermediate merge strings. All temporary files are named 'SRTWRKnn.$$$' where the nn is a two digit number ranging from 00 to 19. These files are automatically deleted at the end of the program.

'OUTPUT RECORDS WRITTEN' is the total number of logical records that were written to the output file. This number should match the number of input records.

'BYTES IN BUFFER' is the total number of bytes available to the QSORT program to use for sorting purposes; ie., TOTAL STORAGE-(CP/M+QSORT).

'BYTES PER LOGICAL RECORD' is the number of bytes per record specified on the parameter file (Field 9).

'KEYS SPECIFIED' is the total number of keys that were specified on the parameter file.

KEY FIELDS:

Each file sorted by the QSORT program can be sorted on one through five separate keys. Each key field is independent of the others and must be of a fixed length in a fixed position in the record. The records are sorted in descending priority from first to fifth key regardless of the position of each key within the record.

There are four items associated with each key. Although only one key is mandatory, all four items are required for each key specified.

1) KEY STAZTING POSITION: (Fields 15,19,23,27,31)

This is the displacement of the key in each record counted from the beginning of the record. It may range from 01H to 0FFH. If the first byte of a record is to be the star| of a key, a 01H is specified for starting position. If the starting position is specified as 00H the key and all subsequent keys are assumed to be unused. If the first key specified has a starting position of 00H an error will occur.

2) KEY LENGTH: (Fields 16,20,24,28,32)

This parameter specifies the number of contiguous bytes that compose the key field including the starting position and extending to the right. This parameter may range from 01H to 0FFH but the last indicated byte may not extend past the end of the record. This parameter must not be zero if the key is used. Note that it is the user's responsibility to assure that key fields do not overlap.

3) ASCEND/DESCEND FLAG: (Fields 17,21,25,29,33)

Each field may be individually specified as to whether it is sorted lowest value first or highest value first. The ordering is based on the hexadecimal (binary) value of each byte of data (see Appendix A) comparing from left to right (high order to low order).

4) ALPHA/NUMERIC FLAG: (Fields 18,22,26,30,34)

The alpha specification should be used only when sorting fields that are composed of both upper and lower case ASCII alphabetic or numeric characters and it is desired that both upper and lower cases be treated as having the same value for sort purposes. This is useful when sorting a name field, for example. The effect of the Alpha designation is to zero out bit 5 of each byte of data in the key before it is compared; ie., 'a' and 'A' will be treated as the same.

If the numeric specification is made, all comparisons will be made according to the binary values of the data (See Appendix A).

DRIVE SPECIFICATIONS: (Fields 2,5,14)

Both the input data file and sorted output file need a drive specification so the QSORT program knows where to locate the files. In addition, the parameter record must contain the drive on which to place the sortwork temporary files. All three drive specifications are made in the same manner. Each drive may be specified as being on drive A, B or the currently logged disk. The currently logged disk concept is explained in your CP/M documentation.

PARAMETER FIELD CONTENTS

Field 1 Unused

Note that this field will contain a double-quote (") when the parameter file is written by a BASIC-E program (QPARM).

Field 2 Input file drive

This field must contain either:

00H = Currently logged disk
01H = Drive A
02H = Drive B

Field 3 Input file name

This field must contain an unambiguous file name of eight or less characters, left-justified and padded with blanks on the right.

Field 4 Input file type

This field must contain an unambiguous file type of three or less characters, left-justified and padded with blanks on the right.

Field 5 Output file drive

This field must contain either a:

00H = Currently logged disk
01H = Drive A
02H = Drive B

Field 6 Output file name

This field must contain an unambiguous file name of eight or less characters, left-justified and padded with blanks on the right.

Field 7 Output file type

This field must contain an unambiguous file type of three or less characters, left-justified and padded with blanks on the right.

Field 8 Unused

This field is assumed to contain a:

00H

Field 9 Record length

This field must contain the number of bytes per logical record in hexadecimal form. This number may range from 01H through 0FFH inclusive.

Field 10 Unused

Field 11 Backup flag

This field must contain either a:

00H = Backup not wanted
01H = Backup wanted

Field 12 Disk change flag

This field must contain either a:

00H = Disk change not necessary
01H = Disk change is necessary

Field 13 Console output flag
This field must contain either a:
00H = Console output not wanted
01H = Console output wanted

Field 14 Work file drive
This field must contain either a:
00H = Currently logged disk
01H = Drive A
02H = Drive B

Fields 15,19,23,27,31
Key starting position
These fields must contain a:
00H = Key is not used
Or range from 01H to 0FFH inclusive indicating in hexadecimal form the displacement in the record of the leftmost (high order) byte of the key. This field must not exceed the record length (Field 9). The first key (Field 15) MUST be non-zero.

Fields 16,20,24,28,32
Key length
These fields must contain a:
00H = Key is not used
Or range from 01H to 0FFH inclusive indicating in hexadecimal form the length in bytes of the key. The contents of the corresponding Start field minus one added to the contents of the length field for any one key must not exceed the record length (Field 9). If the corresponding Start field is non-zero, this field must also be non-zero.

Fields 17,21,25,29,33
Ascend/Descend flag
These fields must contain either a:
041H = ASCII 'A' for ascending
044H = ASCII 'D' for descending

Fields 18,22,26,30,34
Alpha/Numeric flag
These fields must contain either a:
041H = ASCII 'A' for alpha
04EH = ASCII 'N' for numeric

Field 35 Unused

ERROR MESSAGES

If the QSORT program encounters a problem during processing it will print a numbered error message to the system console and stop running. The error message will contain a descriptive phrase that should usually be enough to indicate to the user what the problem is. If more information is desired the message can be found in the following section by using the error number that preceeds each error message in the format:

QSnn msg

Where nn is the error number and msg is the descriptive phrase.

QS01 MISSING OR INVALID SORT PARAMETERS

This message indicates one of the following:

a) The parameter file name was omitted from the command-line when the QSORT program was initiated.
b) The parameter file name on the command-line was incorrectly typed, was too long, had embedded blanks, had the wrong drive specification or the file is not cataloged with the file type of 'SRT'.
c) The key specification parameters are incorrectly formed. See the previous section of key field parameters.

QS02 OPEN ERROR

This message is issued if the QSORT attempts to open a file that does not exist on the specified drive. Check the directory for the name of the input file.

QS03 CLOSE ERROR

This message is issued if the QSORT program attempts to close a file that is not present. If this error occurs it may be due to a system error or to a diskette being changed during the sort process.

QS04 NO DIR SPACE

This message indicates that the program is attempting to create a new file and there is not enough room in the diskette directory. The file being created may be the output file or one of the sort work files. Remove any unnecessary files from the diskette.

QS05 INPUT TOO LARGE

This message will be issued if the QSORT program cannot fit the input data into the maximum allowable number of work files or if the available memory is too small for the merging routine. See Appendix B for a table of maximum input file sizes for different memory configurations.

QS06 not used

QS07 OUT OF DISK SPACE

This message is issued if the sort program runs out of room on a disk write. This can occur on the output file or one of the sort work files. Remove any unnecessary files from the diskette.

QS08 FILE ERROR

This message indicates that CP/M cannot locate a directory entry for an output or work file. This error only occurs on a write and is an indication of a possible serious system error.

QS09 NO DIR SPACE

This message is similar to QS04 except that it will only occur when a file, either output or sort work, is being extended by the system. A file must be extended every 16K bytes.

QS10 not used

QS11 READ ERROR

This message indicates a serious program error and should never occur. It means that the sort program is attempting to read past the end of file on an input or work file.

APPENDIX A

A S C I I C O D E C H A R T

GRAPHIC OR CONTROL	ASCII (HEXADECIMAL)	DECIMAL EQUIVALENT
NUL	00	00
SOH	01	01
STX	02	02
ETX	03	03
EOT	04	04
ENQ	05	05
ACK	06	06
BEL	07	07
BS	08	08
HT	09	09
LF	0A	10
VT	0B	11
FF	0C	12
CR	0D	13
SO	0E	14
SI	0F	15
DLE	10	16
DC1	11	17
DC2	12	18
DC3	13	19
DC4	14	20
NAK	15	21
SYN	16	22
ETB	17	23
CAN	18	24
EM	19	25
SUB	1A	26
ESC	1B	27
FS	1C	28
GS	1D	29
RS	1E	30
US	1F	31
SP	20	32
!	21	33
"	22	34
#	23	35
$	24	36
%	25	37
&	26	38
'	27	39
(	28	40
)	29	41
*	2A	42
+	2B	43
,	2C	44

GRAPHIC OR CONTROL	ASCII (HEXADECIMAL)	DECIMAL EQUIVALENT
-	2D	45
.	2E	46
/	2F	47
0	30	48
1	31	49
2	32	50
3	33	51
4	34	52
5	35	53
6	36	54
7	37	55
8	38	56
9	39	57
:	3A	58
;	3B	59
<	3C	60
=	3D	61
>	3E	62
?	3F	63
@	40	64
A	41	65
B	42	66
C	43	67
D	44	68
E	45	69
F	46	70
G	47	71
H	48	72
I	49	73
J	4A	74
K	4B	75
L	4C	76
M	4D	77
N	4E	78
O	4F	79
P	50	80
Q	51	81
R	52	82
S	53	83
T	54	84
U	55	85
^	56	86
W	57	87
X	58	88
Y	59	89
Z	5A	90
[	5B	91
\	5C	92
]	5D	93
^	5E	94
_	5F	95
`	60	96

GRAPHIC OR CONTROL	ASCII (HEXADECIMAL)	DECIMAL EQUIVALENT
a	61	97
b	62	98
c	63	99
d	64	100
e	65	101
f	66	102
g	67	103
h	68	104
i	69	105
j	6A	106
k	6B	107
l	6C	108
m	6D	109
n	6E	110
o	6F	111
p	70	112
q	71	113
r	72	114
s	73	115
t	74	116
u	75	117
v	76	118
w	77	119
x	78	120
y	79	121
z	7A	122
{	7B	123
!	7C	124
}	7D	125
~	7E	126
DEL	7F	127

APPENDIX B

MEMORY SIZE is the total RAM available in the system.

APP. BUFFER SIZE is the approximate size in kilobytes of the working space available to QSORT. If the input file does not exceed this figure no intermediate workfiles will be used.

Note that 24K of memory is required to sort a full, single-density diskette. At least 36K of memory is needed for a full, double-density diskette. The sort program will process a given input file in less time if more memory is made available.

A TABLE OF MAXIMUM FILE SIZES FOR DIFFERENT MEMORY SIZES

MEMORY SIZE	APP. BUFFER SIZE	MAXIMUM FILE SIZE
16K	7.5K	150K
20	11.5	230
24	15.5	310
28	19.5	390
32	23.5	470
36	27.5	550
40	31.5	630

APPENDIX C

Q P A R M

QPARM is a program distributed with the QSORT package. It is written in BASIC-E and is upward compatible to the CBASIC language. It is used to interactively create parameter files simply and conveniently. The program will request data in conversational mode and will check it for validity. It will also perform such chores as decimal/hexadecimal conversion. The QPARM program should be all the average user needs to construct a complete library of sort parameter files.

The following is a list of the questions that QPARM asks along with the type of response the program expects.

ENTER PARAMETER FILE NAME

Enter any unambiguous file name. QPARM will append the 'SRT' type automatically. This is the name that you will enter on the command line when executing QSORT.

ENTER INPUT FILE DRIVE

Enter either:
'@' to indicate the currently logged disk
'A' to indicate drive A
'B' to indicate drive B

ENTER INPUT FILE NAME

Enter any unambiguous file name. The name may range from one to eight numbers or upper-case alpha characters.

ENTER INPUT FILE TYPE

Enter any unambiguous file type. The type may range from one to three numbers or upper-case alpha characters.

ENTER OUTPUT FILE DRIVE

Same as for Input file.

ENTER OUTPUT FILE NAME

Same as for Input file.

ENTER OUTPUT FILE TYPE

Same as for Input file.

ENTER LOGICAL RECORD LENGTH IN DECIMAL

Enter the length of records on the input file. This number should be in decimal notation and must range from 1 to 255 inclusive. If

the file to be sorted contains carraige-return/line-feed characters as record delimiters (ie., BASIC-E output files) they must be added in the length.

WANT OUTPUT FILE BACKED UP

Respond with a 'Y' or 'N' to determine the procedure the sort program will use if a previous output file exists.

WANT TO CHANGE OUTPUT DISKETTE

Respond with a 'Y' or 'N' to instruct the program whether or not to pause before opening the output file

WANT CONSOLE OUTPUT

Respond with a 'Y' or 'N' to allow or not allow printing of statistical messages at program completion.

ENTER WORK FILE DRIVE

Enter either a:
'@' to indicate the currently logged drive
'A' to indicate drive A
'B' to indicate drive B

ENTER KEY #n STARTING POSITION

This prompt will be given five times or until it is given a zero response. The position must be entered in decimal and must not exceed the record length.

ENTER KEY #n LENGTH

Enter the length of this particular key in decimal. The length and starting position together must not exceed the record length.

ENTER KEY #n ASCEND/DESCEND FLAG

Enter either an:
'A' for ascending
'D' for descending

ENTER KEY #n ALPHA/NUMERIC FLAG

Enter either an:
'A' for alpha
'N' for numeric