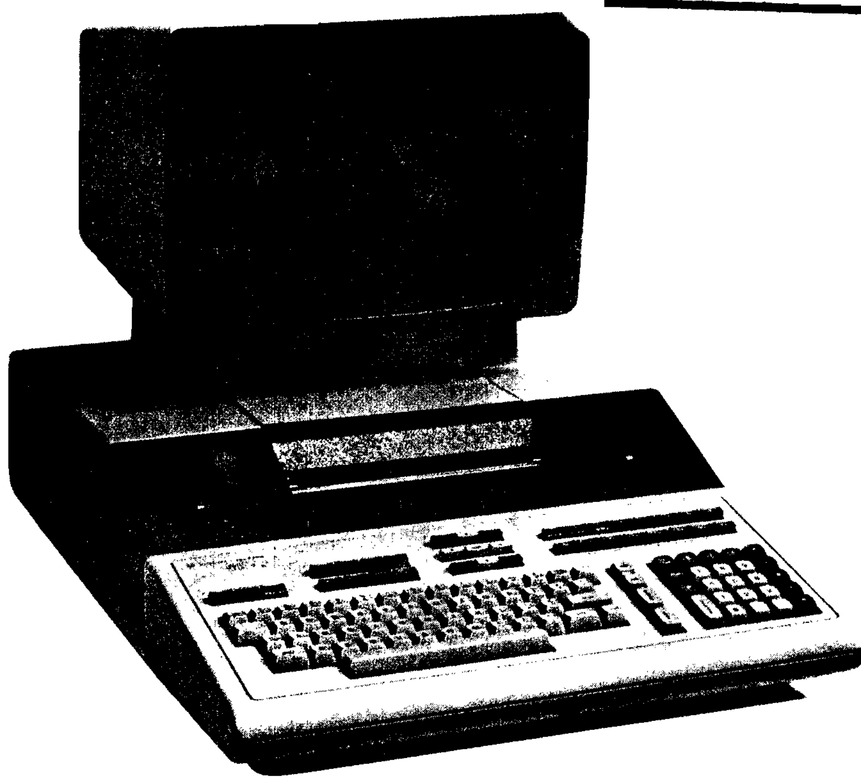
Advanced Programming ROM

TECHNICAL COMPUTER
GROUP – MELBOURNE
LIBRARY COPY



HP System 45B Desktop Computer

Computer
Museum

Hewlett-Packard Desktop Computer Division
3404 East Harmony Road, Fort Collins, Colorado 80525
(For World-wide Sales and Service Offices see back of manual.)
Copyright by Hewlett-Packard Company 1979

Table of Contents

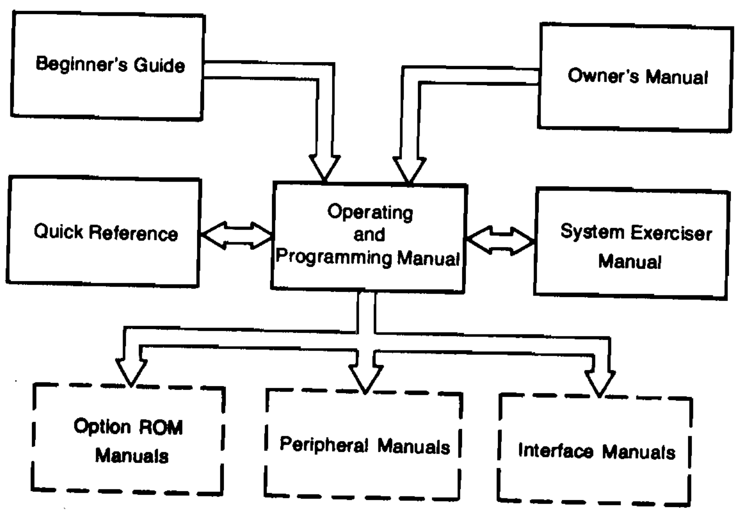
Printing History

Periodically, this manual is updated. Each new edition of this manual incorporates all material updated since the previous edition. Each new or revised page is indicated by a revision date.

The date on the back cover changes only when each new edition is published.

First Printing...May 1, 1979

Manual Map



General Information

Overview

Your HP System 45B Advanced Programming ROM provides extended programming capabilities such as sorting list data in numerical or lexical order, defining a lexical order and searching lists for conditions which you specify. The ASCII, French, German, Spanish and Swedish lexical orders are provided.

You should be familiar with the basic operation of your System 45B before attempting to use the Advanced Programming ROM and this manual. Refer to your System 45B Operating and Programming manual for this information.

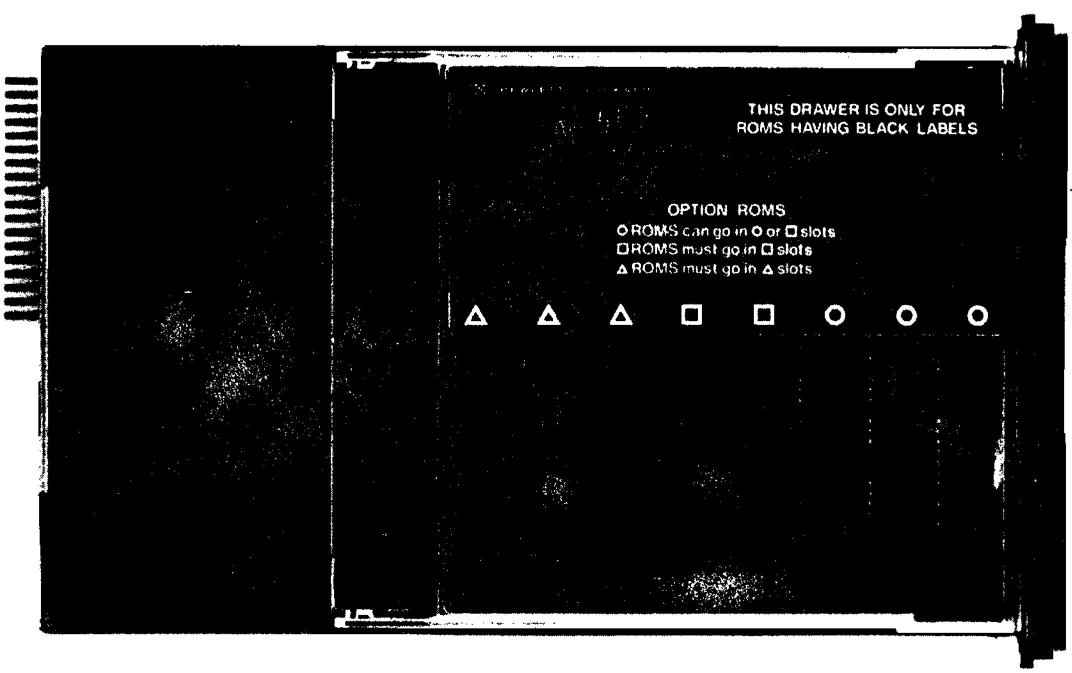
Chapter 1

The System 45B Advanced Programming ROM provides extended computing capabilities for your System 45B Desktop Computer. This ROM enables you to perform such functions as ordering list data in numerical or lexical order and searching lists for conditions which you specify. These features can be very useful in the areas of mathematics, statistics, and information processing. This manual explains and demonstrates the programming features provided by the Advanced Programming ROM.

ROM Installation

The Advanced Programming ROM (HP Part Number 09845-81314) is plugged into the right ROM drawer (black-labeled ROMs). The installation procedure is as follows:

1. Turn off computer power.
2. Slide the right ROM drawer all the way out (see photo).



3. Open the plastic cover by squeezing the sides of the cover and lifting to gain access to the ROM connectors.
4. Press the Advanced Programming ROM onto any available O drawer connector so that it seats all the way down.
5. Close the plastic cover and slide the ROM drawer back into the computer until the drawer is flush with the computer housing.
6. Turn the computer on.

CAUTION

POWER TO THE COMPUTER MUST BE TURNED OFF WHILE INSTALLING ROMS OR DAMAGE TO THE ROM OR TO THE COMPUTER MAY RESULT.

Lexical Tables Cartridge

An Advanced Programming ROM Lexical Tables Cartridge (P/N 09845-90448) is provided for use with the Advanced Programming ROM. The cartridge contains ASCII and local language collating tables which can be modified for particular collating applications. Descriptions of the tables and instructions for their use are found in Appendix B.

Manual Requirements

Before using this manual or the Advanced Programming ROM, you should be familiar with the basic operating procedures of the System 45B Desktop Computer as explained in its Operating and Programming Manual.

Error Messages

The Advanced Programming ROM adds 15 additional error messages (330 through 344) to the System 45B mainframe error message list. Explanations of Advanced Programming error messages as well as mainframe error messages are found in Appendix C of this manual.

Manual Syntax

The following conventions apply to the syntax used in this manual:

Dot Matrix –	All items in dot matrix must be entered as shown.
[ ] –	All items in brackets are optional.
... –	Three dots indicate that successive parameters are allowed, when each is separated by a comma.

Data Manipulation

Terms

- **Record** – represents data which is manipulated as a unit within an array.
- **Key** – a data item within a record used to identify the record for sorting purposes.
- **Key specifier** – a format used in a syntax to specify primary and/or secondary keys within a record.
- **Pointer array** – a one-dimensional numeric array which contains the sorting order of specified records.
- **Location specifier** – a format used within a MAT SEARCH statement syntax to specify the locations to be searched.

Statement Syntax

MAT SORT source array ⟨key specifier⟩ [[substring specifier]]
[DES] [, ⟨secondary key specifier⟩ [[substring specifier]] [DES]...] [TO pointer array]

MAT REORDER object array BY pointer array [, dimension specifier]

MAT SEARCH source array ⟨location specifier⟩, condition; variable [, starting address]

condition:

LOC ⟨relational operator-expression⟩
#LOC ⟨relational operator-expression⟩
MAX
MIN

Chapter 2

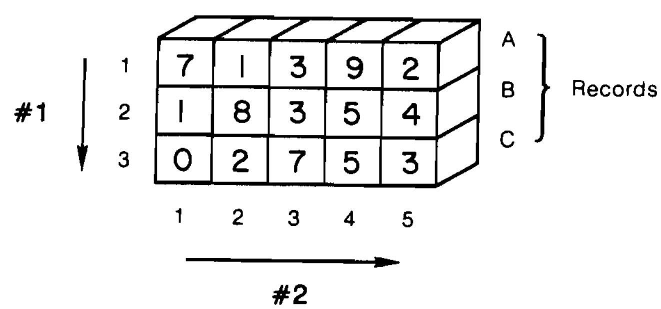
Introduction

The statements described in this chapter are used as programming aids to manipulate data. They concern the sorting and searching of numeric and string arrays. Each of these statements can be executed from within a program or from the keyboard. Examples of each statement are included to demonstrate its use.

Array Structure and Terminology

For a better understanding of the sorting and searching processes introduced in this manual, a brief description of array structure and terminology is helpful.

The following illustration represents a two-dimensional array containing numeric data.



Array X

It is dimensioned 3 rows by 5 columns as follows:

```
OPTION BASE 1
DIM X(3,5)
```

The subscripts within parentheses (3,5) are written in the order in which the array is dimensioned (i.e., in the order of the numbered arrows). Throughout this manual dimensioning assumes OPTION BASE 1 (i.e., numbering begins with 1, not 0). The dimension statement specifies that the array name is X and that it is three rows by five columns.

An array **record** represents data (string or numeric) which is manipulated as a unit within an array. For example, if you arrange a list of names in a specified order, each name is considered a record within that list. Similarly, an array might contain a list of social security numbers. If you arrange them in a particular order, each social security number is considered a record within that array.

In the previous example, if array X is to be rearranged by rows, each row is considered a record and ordering is performed along the first dimension. That is, the positions of the rows in relationship to each other along the first dimension are rearranged. Likewise, if the array is to be reordered by column, each column is considered a record and reordering is performed along the second dimension.

In the previous illustration, assume that each row is a record (A,B, and C). If the records are to be sorted, a **key** is needed to determine how each row is to be ordered in relationship to the others. Assume that the records are to be sorted in ascending order according to the value of the first number in each record. The first column, then, contains the keys for this sort. The sorting process sorts the keys in ascending order along the dimension in which they lie. In so doing, the records in which the keys lie are rearranged.

The keys selected are described by the following format:

```
(*,1)
```

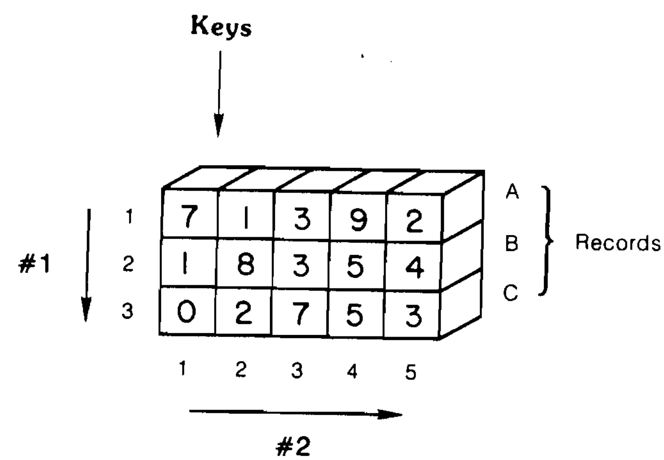
The asterisk indicates a varying subscript. The first dimension subscript is varied over its range of values (1 to 2 to 3) to designate a key in all three records (rows). The second dimension subscript is fixed at 1 to indicate that the keys lie in the first column of each record.

The format (*, 1) is called a **key specifier**. The asterisk replaces the subscript which corresponds to the dimension along which the records lie, in this case, the rows. The number 1 indicates where the key is located within each record. In this way, the key specifier indicates how the array is partitioned into records, and where the keys are located within those records.

If all the combinations of the key specifier in the previous example are listed, a description of the individual keys is obtained as follows.

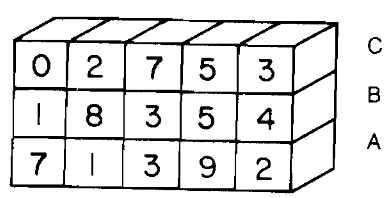
1st Subscript	2nd Subscript	Key
1	1	1st
2	1	2nd
3	1	3rd

The shaded cells in the next figure represent the keys described by the key specifier.



ARRAY X

Upon execution, the sorting process arranges the records in ascending order along the first dimension according to the values of their respective keys. The sorted array is shown next.



ARRAY X AFTER SORTING

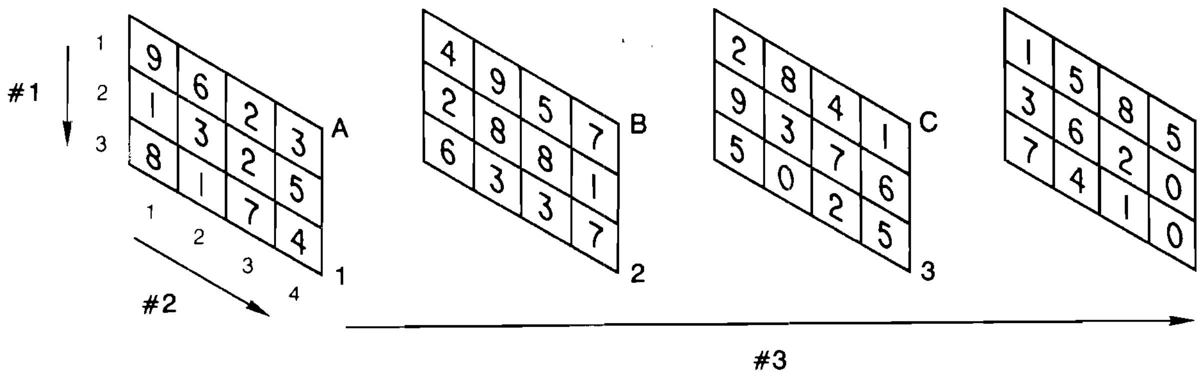
The records, or rows, are repositioned in relationship to each other according to the value of their keys. The contents of the records are left unchanged.

Note that any corresponding numbers within the rows could be selected as keys. The column containing the keys would be described by the key specifier to reflect the proper keys.

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

As another example, assume array Y is a three dimensional numeric array as shown.



ARRAY Y WITH KEYS

It is dimensioned in the order of the numbered arrows (3 rows by 4 columns by 4 planes):

```
OPTION BASE 1
DIM Y(3,4,4)
```

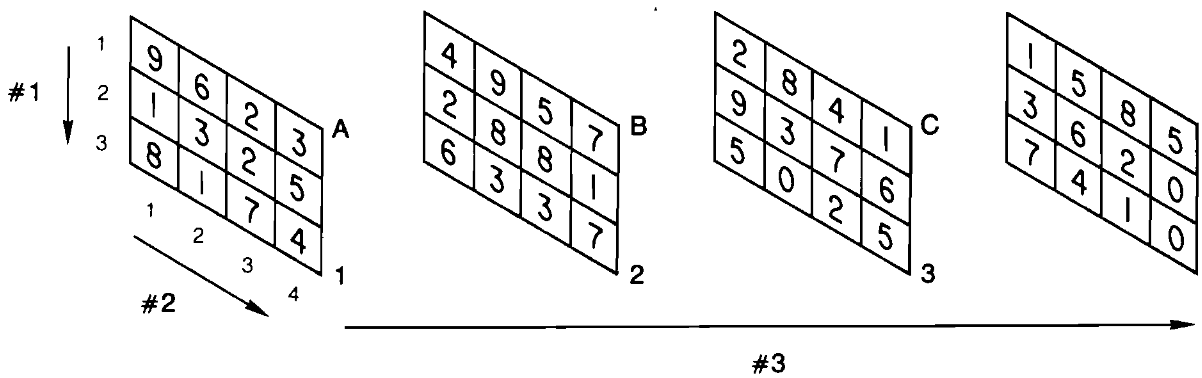
Assume that the array is divided into records (A,B,C, & D) along the third dimension, that is, each plane is considered a record. The keys selected must also lie along the third dimension since there must be one for each record. If the upper right number in each record is designated as a key, the key specifier is

```
(1,4,*)
```

where the asterisk indicates that the third subscript is varied over its entire range of values (1 through 4). In this way, a total of four keys is described as shown.

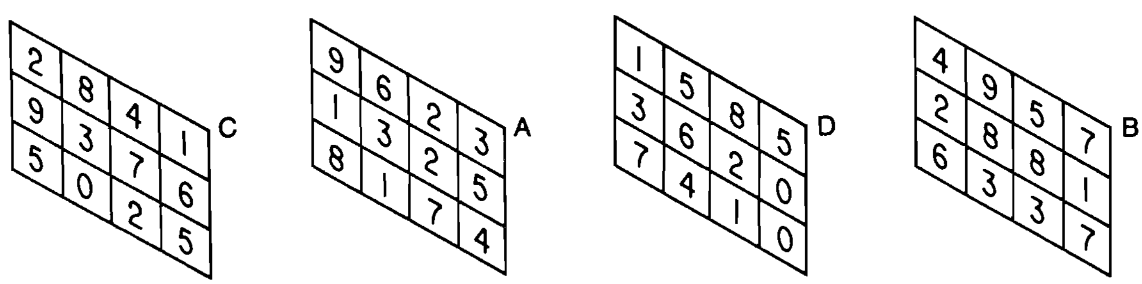
1st Subscript	2nd Subscript	3rd Subscript	Key
1	4	1	1st
1	4	2	2nd
1	4	3	3rd
1	4	4	4th

The shaded cells in the following illustration represent the four keys described by the key specifier (1,4,*).



ARRAY Y WITH KEYS

If the array records are sorted in ascending order, they are rearranged along the third dimension according to the values of their keys. The following illustration represents the array Y after it is sorted.



ARRAY Y AFTER SORTING

Note that the records have been rearranged according to the ascending values of their keys.

MAT SORT Statement

Sorting Numeric Data

The MAT SORT statement is used to order data records in an array. If the data is numeric, it can be sorted in either ascending or descending order. If the array contains string data, it can be sorted in either lexical (alphabetical) or reverse lexical order.

Syntax:

MAT SORT numeric source array (key specifier)

The source array represents the array in which sorting is performed. The key specifier allows you to specify the keys by which the records of data are sorted. If the source array is one-dimensional, no key specifier need be included.

Example:

```
MAT SORT A(1,*,3)
```

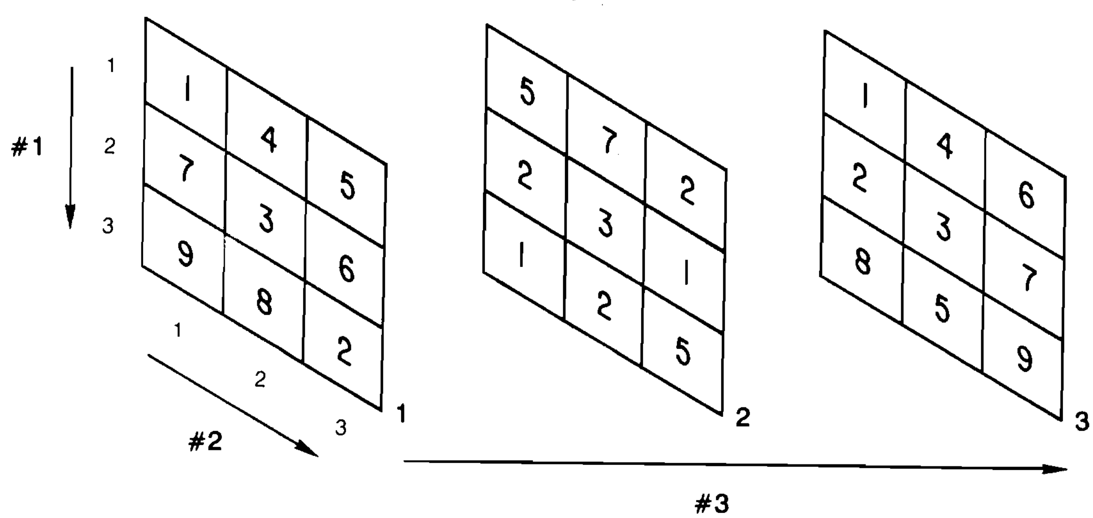
In this case, A is a numeric array in which sorting is performed. The key specifier indicates that A is a three dimensional array and that its records lie along the second dimension. The position of the asterisk within the key specifier determines which subscript is varied, and therefore, how the array is divided into records. If the first subscript were to identify records instead of the second, `MAT SORT A(*,2,3)` would be entered. The array in which sorting is performed can have no more than six dimensions. Numeric sorting comparisons are performed in the INTEGER mode for integer-precision arrays and in the REAL mode for short or real-precision arrays.

The default order of sorting is ascending. In the previous example statement, the records are sorted in ascending order. If descending order is desired, the correct entry is

```
MAT SORT A(1,*,3) DES
```

where the letters DES inclued after the key specifier indicate descending order.

The following example serves to explain the MAT SORT process. Assume array Data is a three dimensional numeric array containing random numbers.



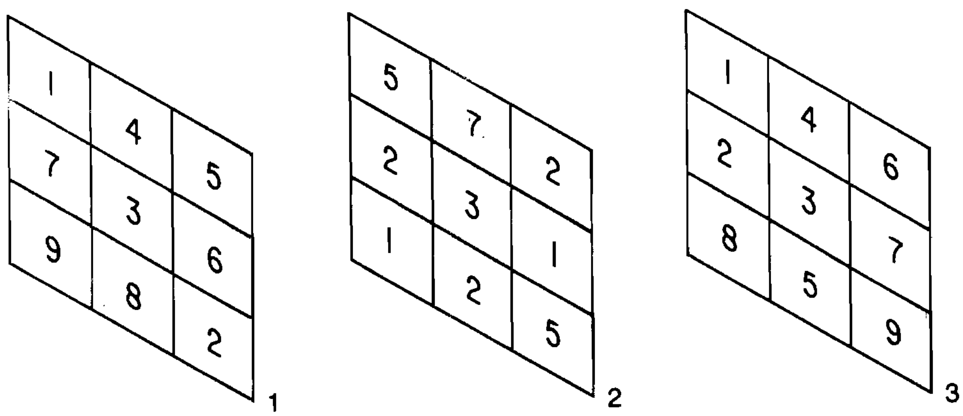
ARRAY Data BEFORE SORTING

This illustration is a graphical representation of the array and its contents. Each record is numbered as are its rows and columns. The numbered arrows represent the order in which the array was originally dimensioned.

Assume that the records are to be sorted in ascending order and that the numbers in the upper right location in each record are designated as keys. Since the records all lie along the third dimension, the third subscript is replaced by an asterisk in the key specifier. The proper sorting statement is

```
MAT SORT Data(1,3,*)
```

where the first two numbers indicate the upper right location of each record. The asterisk indicates that the records are selected by varying the third subscript over its range of values (1 to 2 to 3). After the sort is performed, the array is rearranged as shown next.



ARRAY Data AFTER SORTING

Note that the order of the records is changed according to the value of their keys.

A pointer array can be specified in a sorting statement to maintain a record of how the source array should be rearranged.

Syntax:

`MAT SORT` source array (key specifier) `TO` pointer array

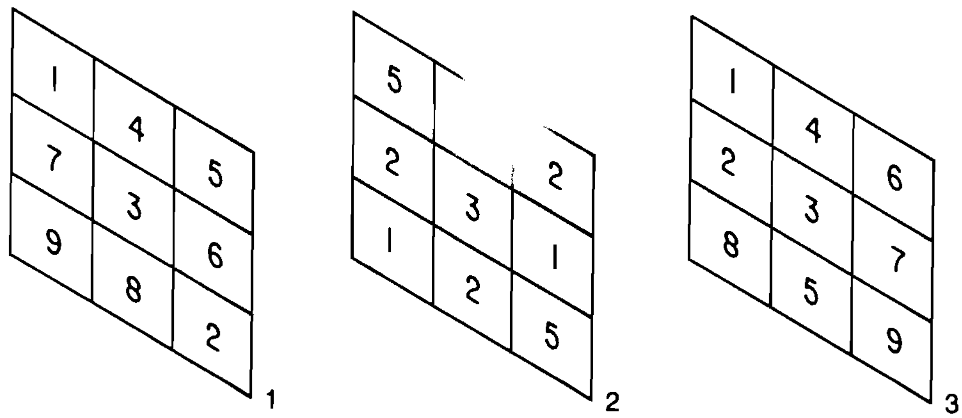
When a pointer array is included in a MAT SORT statement, the sorting process does **not** rearrange the source array. Instead, it fills the pointer array with a series of numbers representing the order of the source array records as if they were sorted. In this way, the source array is not disturbed, but the order in which its records should be sorted is maintained for future use.

The pointer array must be a one-dimensional numeric array and it must be the same length as the range of records to be sorted. The pointer array does not contain the contents of the records, but rather, a series of numbers representing the order in which the records would be sorted.

For example, assume that the sorting statement in the previous example is modified to include the pointer array Point. The pointer array must be dimensioned to be three elements in length to accommodate the number of records in the source array (`DIM Point(3)`). The sorting statement is modified as follows:

```
MAT SORT Data (1,3,*) TO Point
```

Upon execution, the source array remains u nged as shown.



ARRAY Data AFTER SORTING

However, the pointer array now contains the order of the records as if they were rearranged.

2	1	3
1	2	3

POINTER ARRAY Point AFTER SORTING

Note that the numbers in the pointer array correspond to the sorted order of the records in the previous example. However, the source array is not rearranged.

The numbers in the pointer array are the values that the varied subscript would assume in designating each record. For example, if Data was dimensioned DIM Data (3,3,−1:1), then Point would contain

0	-1	1

This allows the pointer array to be used for indirect reference into the original array. Thus, after sorting this example, Data (3,1, Point(2)) = 9.

If two items described by the key specifier are identical, a secondary key specifier can be used to complete the sort. The sorting process utilizes the order of the data described by the secondary key specifier to arrange the records containing identical primary keys. Should further identical data be described by the secondary key specifier, additional key specifiers can be included. The asterisks must appear in the same respective positions in all related key specifiers.

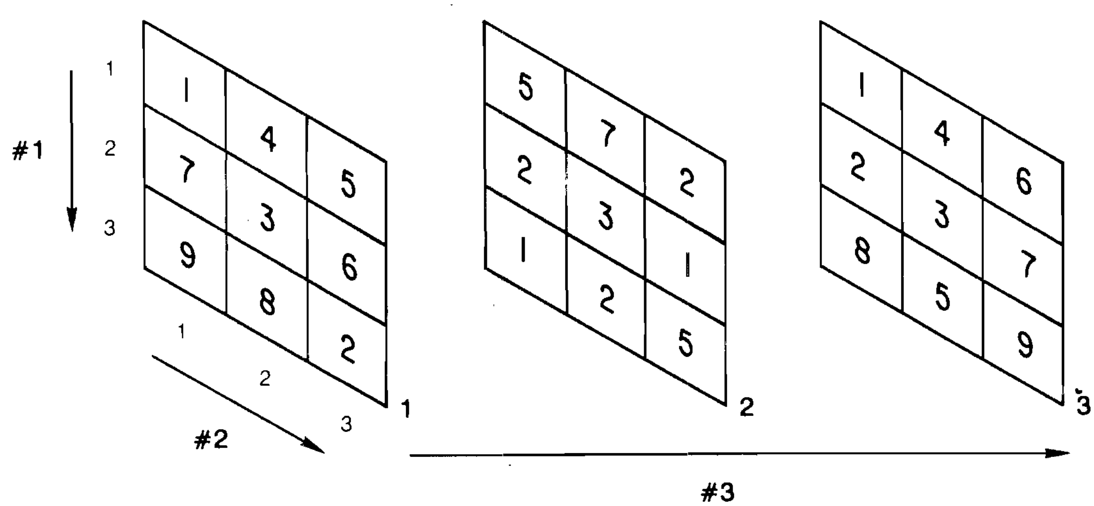
Syntax:

`MAT SORT` source array (key specifier) [, (secondary key specifier)...]

Note that commas are used to separate individual key specifiers. All key specifiers in a given MAT SORT statement must partition the array into records in the same way. That is, the asterisk must always appear in the same position.

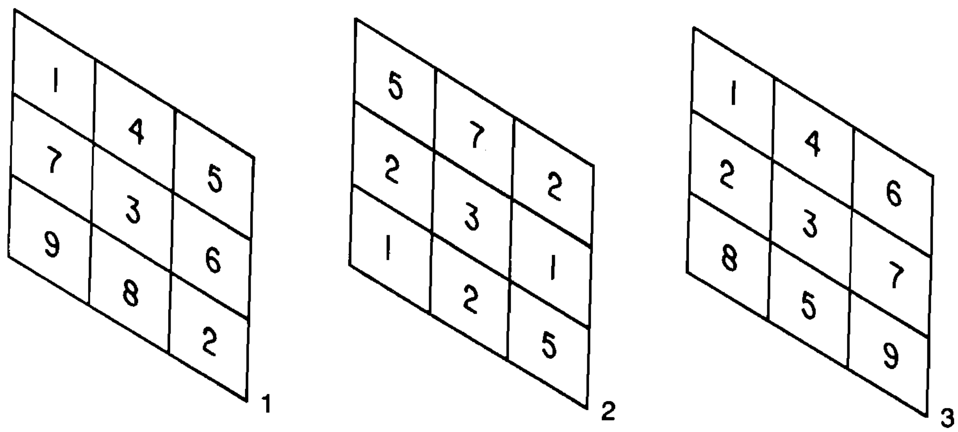
Referring to the previous array example, assume that the records are to be sorted using their upper left location contents as keys. The correct sorting statement is `MAT SORT Data(1,1,*)`, where (1,1,*) specifies the upper left numbers as keys. An ambiguity would develop, however, since the number "1" appears in the keys of both records one and three. The values of another series of keys can be used to determine which "1" should be ordered first. Assume that the numbers in the lower right locations are described by a secondary key specifier. The proper sorting statement is:

```
MAT SORT Data (1,1,*),(3,3,*)
```



ARRAY Data WITH PRIMARY & SECONDARY KEY SPECIFIERS

Since identical numbers have been encountered in records one and three, the sorting process examines the lower right keys of the records (as described by the secondary key specifier(3,3,*)) to complete the sort. Since the "2" of record one precedes the "9" of record three, record one is ordered before record three. Upon execution of the sorting statement, array Data is rearranged as shown.



ARRAY Data AFTER SORTING

If identical items cannot be differentiated, then the relative order of the sorted records is indeterminate (due to the nature of the sort).

The program shown next demonstrates a use of numeric sorting. A list of salespeople, their districts, and their sales volumes is sorted according to district and volume.

```
10  ! ************************************************************
20  ! ***       This program demonstrates the use of the       ***
30  ! ***                  MAT SORT statement.                 ***
40  ! ************************************************************
50                                       !
60    OPTION BASE 1                      ! Select OPTION BASE.
70    DIM Salespeople$(9)[20],Sales(12,2,9),Rank(9) ! Dimension
80                                       ! source and pointer arrays.
90                                       !
100 ! ************************************************************
110 ! *** Data represents district, sales volume, salespeople. ***
120 ! ************************************************************
130                                      !
140     DATA 3,3000,1,6000,2,1900,1,1200,3
150     DATA 1400,1,2500,3,900,2,2200,2,1700
160     DATA JILL TANDY,DON DEEDS,RICK HILL,SAM SPADE,NICK DANGER
170     DATA JOE JONES,HARRY WHITE,BARB SMITH,H.J. STEED
180                                      !
190 ! ************************************************************
200 ! ***             Print headings and sort.                 ***
210 ! ************************************************************
220                                      !
230    PRINT TAB(10);"   DATA FOR MONTH";Month;LIN(1)
240    PRINT "SALESPEOPLE";TAB(17);"DISTRICT";TAB(33);"SALES";LIN(1)
250                                      !
```

```
260    Month=6                          ! Select month of sales survey.
270                                     !
280      FOR I=1 TO 9
290        READ Sales(Month,1,I),Sales(Month,2,I)  ! Input district
300      NEXT I                         ! and sales volume.
310                                     !
320    MAT READ Salespeople$            ! Input salespeople.
330                                     !
340      FOR I=1 TO 9
350        PRINT Salespeople$(I);TAB(19);Sales(Month,1,I);TAB(32);
360        PRINT Sales(Month,2,I)
370      NEXT I                         ! Print the unsorted sales data.
380                                     !
390    WAIT 5000                        ! Wait 5 seconds before continuing.
400    PRINT PAGE                       ! Clear the screen.
410                                     !
420    PRINT TAB(10);"  RESULTS FOR MONTH";Month;LIN(1)  ! Print
430                                     ! a new heading.
440    MAT SORT Sales(Month,1,*),(Month,2,*) DES TO Rank ! Sort
450                                     ! the sales by district; use
460                                     ! sales volume as a secondary key.
470                                     !
480  ! **********************************************************
490  ! *** Establish FOR/NEXT loop for printing sales results.   ***
500  ! **********************************************************
510                                     !
520      FOR I=1 TO 9
530        IF District=Sales(Month,1,Rank(I)) THEN Skip ! Test: Has
540                                     ! district changed?
550                                     ! If so, skip print routine.
560                                     ! If not, continue.
570        District=Sales(Month,1,Rank(I)) ! Set District to new value.
580                                     !
590        PRINT TAB(10);"  DISTRICT";District;"RESULTS" ! Print heading.
600 Skip:  PRINT Salespeople$(Rank(I));TAB(30);"  ";Sales(Month,2,Rank(I))
610      NEXT I                         ! Go on to next salesperson.
620    END
```

The program first prints the unsorted list. The sort is performed by district using sales volume as the secondary key specifier. If the program is run, the results shown next are obtained.

DATA FOR MONTH 0

SALESPEOPLE	DISTRICT	SALES
JILL TANDY	3	3000
DON DEEDS	1	6000
RICK HILL	2	1900
SAM SPADE	1	1200
NICK DANGER	3	1400
JOE JONES	1	2500
HARRY WHITE	3	900
BARB SMITH	2	2200
H.J. STEED	2	1700

```
                    RESULTS FOR MONTH 6

                    DISTRICT 1 RESULTS
DON DEEDS                                   6000
JOE JONES                                   2500
SAM SPADE                                   1200
                    DISTRICT 2 RESULTS
BARB SMITH                                  2200
RICK HILL                                   1900
H.J. STEED                                  1700
                    DISTRICT 3 RESULTS
JILL TANDY                                  3000
NICK DANGER                                 1400
HARRY WHITE                                 900
```

Sorting String Data

The sorting process for string data follows that for numeric data except that ordering is lexical rather than numeric.

Syntax:

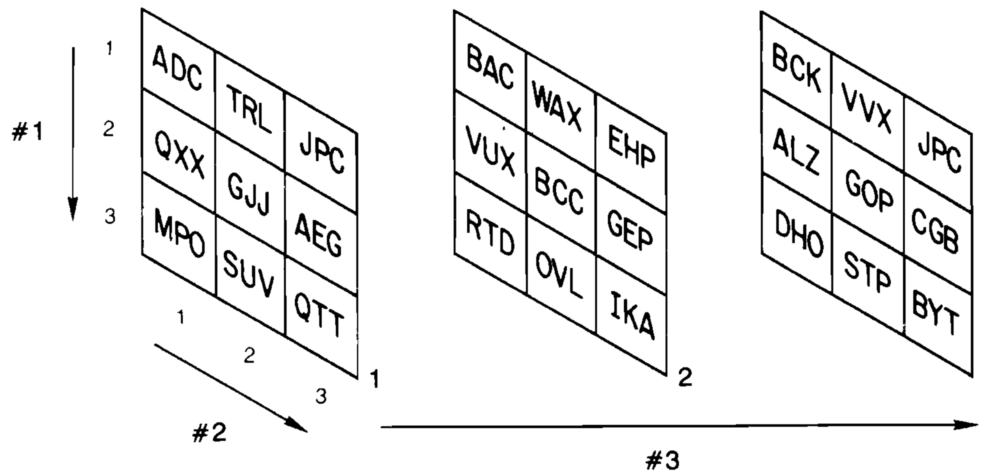
`MAT SORT` string source array (key specifier) [`TO` pointer array]

The source array in this case contains string data. The key specifier specifies which locations are designated as keys. The optional pointer array is a one-dimensional numeric array which contains the order in which the records should be sorted. As with numeric arrays, the source array is not actually rearranged when a pointer array is used.

String sorting uses the LEX function on the string data to perform the ordering (see Chapter 3). The default order is lexical (ascending), but DES can be used to specify reverse lexical (descending) order. The order can also be determined by a user-defined table as described in the LEXICAL ORDER IS section.

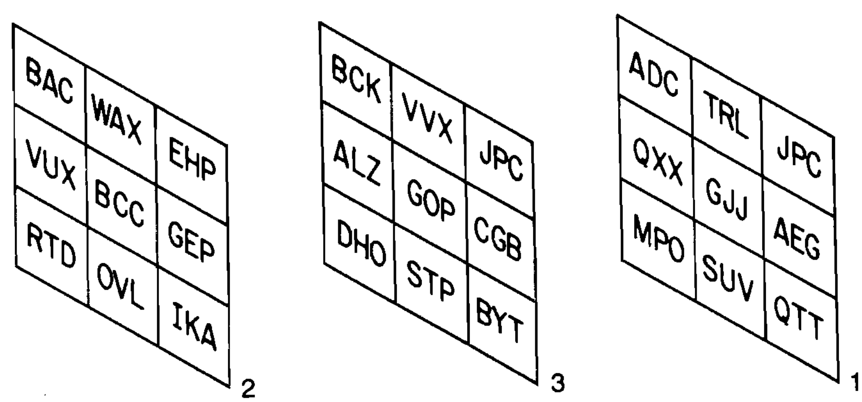
For example, assume array A$ is dimensioned to be a three-dimensional string array containing three characters per element (`DIM A$(3,3,3)[3]`). Assume that the middle location in the right-hand column of each record (plane) is designated as a key. If the records are to be sorted along the third dimension in reverse lexical order, the correct sorting statement is:

```
MAT SORT A$(2,3,*) DES
```



ARRAY A$ WITH KEYS

This illustration represents the source array. The sorting process sorts the records according to the ASCII values of the data in the key locations. The sorting process begins comparing key characters until dissimilar characters are found. Since the first characters of all the data in the keys are different, sorting can be performed without comparing further characters. When the sort is complete, the array is rearranged as shown.

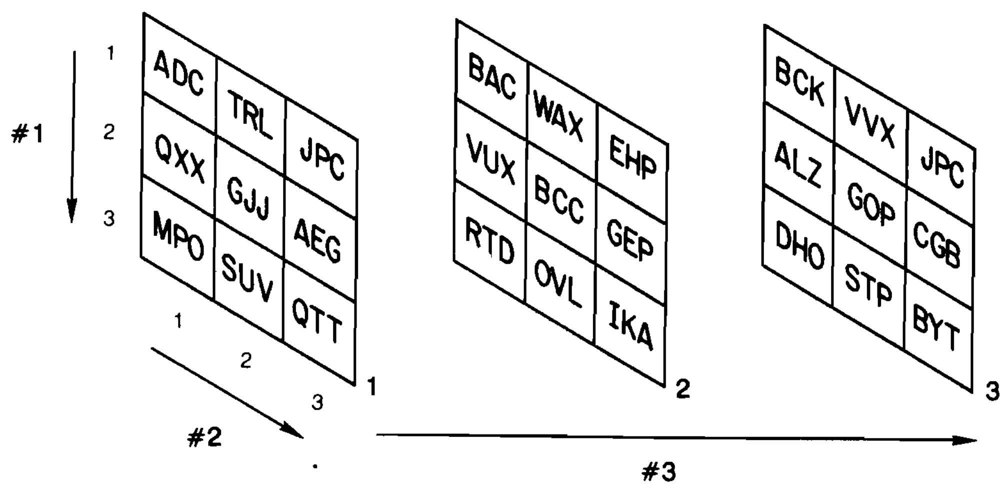


Note that the string data within each memory location has not been rearranged. The records, however, have been rearranged to reflect the new order of the keys of string data. It is important to note that strings of unequal length present no problem. Any unused dimensioned character spaces are filled with the null string and are sorted accordingly. (Example: AB precedes ABC.)

As with numeric sorting, a secondary key specifier can be used to order records which contain identical keys. In the previous array example, assume that the characters in the upper right locations of the records are designated as keys and that the records are to be sorted along the third dimension in lexical order. The sorting process begins comparing characters until dissimilar characters are found. After comparing all the characters in the primary keys of records one and three, the sort recognizes identical data. The sort process could then utilize a secondary key specifier to perform the sort. (All characters must match in order to be considered identical data).

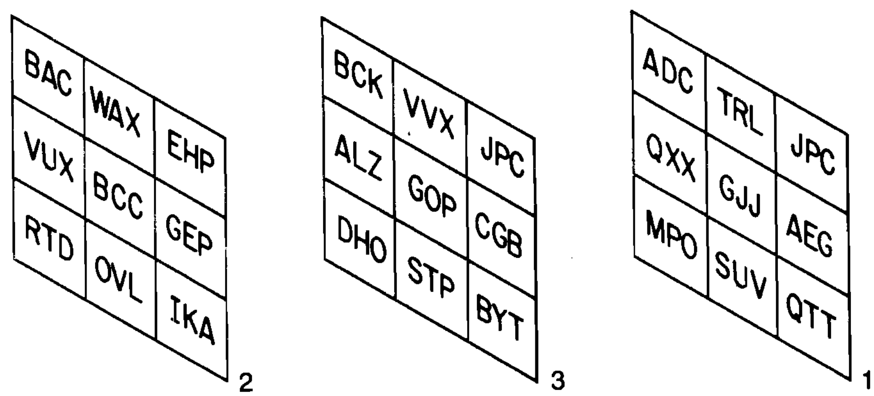
Assume that the data in the lower right locations is described by a secondary key specifier. The correct sorting statement is:

```
MAT SORT A$(1,3,*),(3,3,*)
```



Comparing the data described by the secondary key specifier, the sorting process would order record three before record one since the letter "B" precedes the letter "Q" in the STANDARD lexical order.

Upon execution of the sort statement, the array A$ is rearranged as shown.



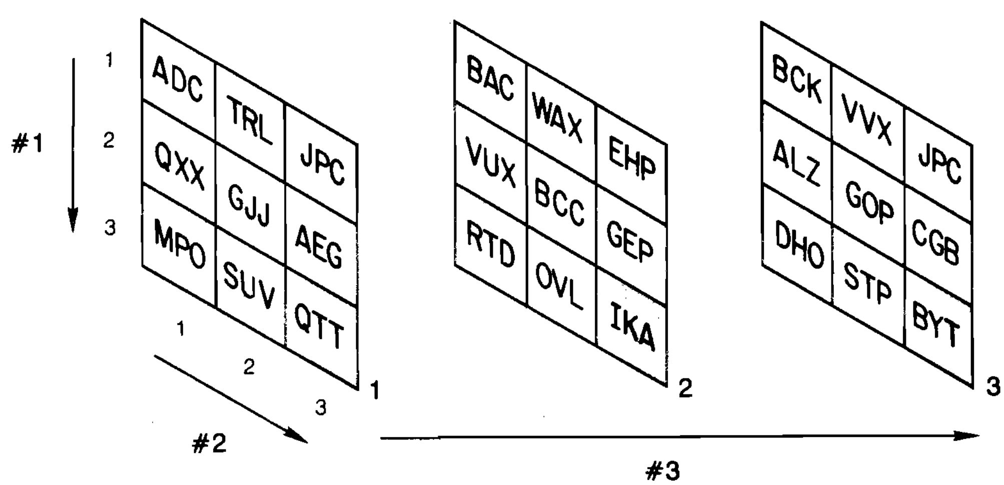
ARRAY A$ AFTER SORTING

When sorting string data, a substring specifier can be used to describe partial key specifier strings.

Syntax:

MAT SORT string source array (key specifier) [[substring specifier]]

The optional substring specifier indicates which portion of the data string in the specified keys is used to order the records. Referring to the previous example, assume the second and third characters in the strings located in the lower right memory locations are designated as keys.



A$ KEYS WITH SUBSTRING SPECIFIER

The correct sorting statement is

```
MAT SORT A$(3,3,*) [2,3]
```

where [2,3] specifies that portion of each key string which begins at the second character and ends with the third. In this example, the sorting process begins sorting the records according to the values of the second and third characters in each key string. Sorting can be completed at the second characters since none of them are identical. After execution of the MAT SORT statement, the rearranged record order is 2-1-3.

A substring specifier can also be written in the form [2;3], where the substring described in this case starts at the second character and continues for three characters.

A substring specifier can be used with a secondary key specifier also. The following is a typical entry.

```
MAT SORT A$(1,*,2),(4,*,6) [4;4]
```

The substring specifier ([4;4]) indicates that the secondary key specifier (4,*,6) describes a substring which starts at the fourth character and extends for four characters.

Substring specifiers can be used with primary and secondary key specifiers simultaneously, as can descending specifiers. Example:

```
MAT SORT A$(1,*,3) [4;4] DES,(4,*,6) [3;3] DES
```

When used with multiple key specifiers, substring specifiers need not be of equal length.

A pointer array can also be included:

```
MAT SORT A$(1,*,3) [4;4] DES,(4,*,6) [3;3] DES TO B
```

Note that no punctuation is included between a key specifier and its related specifiers. A comma is included, however, between complete individual specifier entries.

The program shown next demonstrates string sorting. A list of names and phone numbers is read into a string array and then sorted according to last name.

```
10   ! ************************************************************
20   ! ***        This program demonstrates the use of          ***
30   ! ***              the MAT SORT statement.                 ***
40   ! ************************************************************
50                                          !
60     OPTION BASE 1                        ! Select OPTION BASE.
70     DIM Phones$(7,2)[20]                 ! Dimension source array.
80                                          !
90       DATA RICHARD  HILL,818-8086
100      DATA MICHAEL  HILL,916-6502
110      DATA BARB     SMITH,917-6802  ! Name and phone
120      DATA HARRY    RULE,818-8048   ! number data.
130      DATA JULIE    TANDY,818-8080
140      DATA SAMUAL   SPADE,818-6809
150      DATA MICHAEL  SMITH,916-6800
160                                         !
170      MAT READ Phones$                   ! Read data into source array.
180                                         !
190      MAT SORT Phones$(*,1)[10],(*,1)[1,9]  ! Sort data by last name;
200                                         ! use first name as secondary key.
210                                         !
220  ! ************************************************************
230  ! ***       Print heading and sorted directory.            ***
240  ! ************************************************************
250                                         !
260      PRINT TAB(4);"THE SORTED DIRECTORY:";LIN(1)   ! Print heading.
270        FOR I=1 TO 7
280          PRINT Phones$(I,1)[10];",";Phones$(I,1)[1,9];TAB(20);
290          PRINT Phones$(I,2)      ! Print the sorted directory.
300        NEXT I
310    END
```

Notice that the data are sorted by last names using the first names as a secondary key. If the program is run, the results shown next are obtained.

```
    THE SORTED DIRECTORY:

HILL,MICHAEL       916-6502
HILL,RICHARD       818-8086
RULE,HARRY         818-8048
SMITH,BARB         917-6802
SMITH,MICHAEL      916-6800
SPADE,SAMUAL       818-6809
TANDY,JULIE        818-8080
```

MAT REORDER Statement

The MAT REORDER statement is used to order an array according to the contents of an existing pointer array.

Syntax:

`MAT REORDER` object array `BY` pointer array [, dimension specifier]

The object array in this syntax is rearranged in the order specified by the contents of the pointer array. The maximum allowable number of dimensions of the object array is six. The optional dimension specifier selects the dimension of the object array along which records are ordered. The dimension specifier is a number from 1 to 6 or an expression which represents this number. If it is not specified, a value of 1 is assumed by the computer. The dimension specifier describes the object array dimensions in the manner shown next.

```
DIM Object_array(A,B,..., F)
                 ↑ ↑      ↑
Dimension Specifier: 1 2      6
```

For example, assume that the array Point is a pointer array containing a range of values determined by a previous sorting process as shown.

4	2	1	3

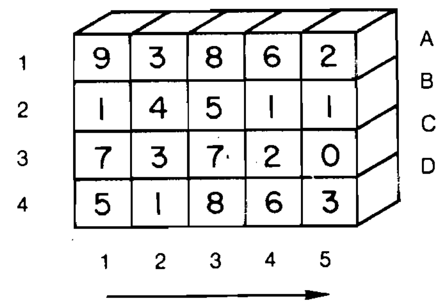
It is dimensioned as follows:

```
OPTION BASE 1
DIM Point (4)
```

Also, assume that array Object is a two-dimensional object array which contains numeric data and is dimensioned as shown.

```
OPTION BASE 1
DIM Object (4,5)
```

The following illustration represents array Object.



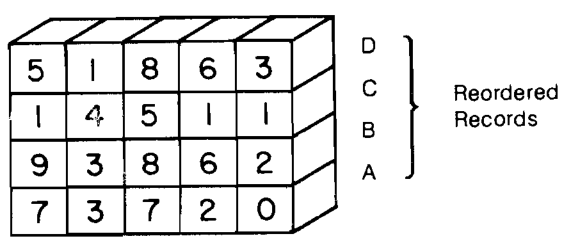
#2

ARRAY Object

If array Object is to be rearranged along the first dimension according to the contents of array Point, the correct reordering statement is

```
MAT REORDER Object BY Point,1
```

where Object is the object array, Point is the pointer array, and 1 specifies that the object array is reordered along its first dimension (i.e., each row is considered a record). Upon execution of the reorder statement, array Object is rearranged as shown next.



ARRAY Object AFTER REORDERING

The records are rearranged in the order specified by the pointer array. Note that they are not necessarily rearranged in ascending or descending order. There is no necessary relationship among the records. They are merely rearranged according to the pointer array.

The pointer array must be dimensioned the same size as the dimension of the object array along which reordering is performed. In the previous example, the pointer array Point is dimensioned four elements in length as is the first dimension of array Object as shown next.

```
DIM Object(4,5), Point(4)
```

NOTE

If the pointer array contains duplicate numbers, unpredictable results occur. Also, if the pointer array contains numbers which are out of range for the specified dimension of the object array, an error message results when reordering is attempted.

As you may recall, a MAT SORT statement which contains a pointer array does not rearrange the source array upon execution. It merely fills the pointer array with the order of the records as if they were rearranged. The MAT REORDER statement can be used to rearrange that source array at a later time. For example,

```
MAT SORT A(1,*,3) TO B
```

fills the pointer array B with the proper sorted order of records, but it does not actually rearrange the source array A. In order to rearrange the source array, a MAT REORDER statement can be used. Example:

```
100 MAT SORT A(1,*,3) TO B

200 MAT REORDER A BY B, 2
```

The execution of line 100 fills the pointer array but does not rearrange the source array A. Line 200, however, does rearrange the source array according to the order defined earlier by the MAT SORT statement.

A program is now presented that demonstrates a possible use of the MAT SORT and MAT REORDER statements. Original$ is a one-dimensional array list of names and grade point averages.

The program fills the array list and sorts the data both by name and grade point average.

```
10 ! ********************************************************
20 ! ***          This program demonstrates the use       ***
30 ! ***               of the MAT SORT statement.         ***
40 ! ********************************************************
50                                      !
60    OPTION BASE 1                     ! Select OPTION BASE.
70    DIM Original$(6)[21],B(6),C(6),D(6) ! Dimension source
80                                      ! and pointer arrays.
90      MAT READ Original$              ! Read data into source array;
100                                     ! data in Lines 330-380.
101                                     !
110     PRINT "THE ORIGINAL SEQUENCE IS:",LIN(1)
120     PRINT Original$(*)              ! Print the unsorted list.
130                                     !
140     MAT SORT Original$(*)[1,16] TO B  ! Sort list by last names.
150     MAT SORT Original$(*)[18,21] DES,(*)[1,16] TO D ! Sort list by
160                                     ! grade points; use names
170                                     ! as secondary keys.
180       FOR I=1 TO 6
190         C(B(I))=I                   ! Compute the inverse permutation.
200       NEXT I
210                                     !
220     MAT REORDER Original$ BY B      ! Reorder list by name.
230     PRINT "REORDERED BY NAME:",LIN(1)
240     PRINT Original$(*)              ! Print the reordered list.
250                                     !
260     MAT REORDER C BY D              ! Combine permutations C and D
270                                     ! into C.
280     MAT REORDER Original$ BY C      ! Reorder list by grade point
290                                     ! using the combined permutation.
300     PRINT "REORDERED BY GPA:",LIN(1)
310     PRINT Original$(*)              ! Print reordered list.
320                                     !
330     DATA SMITH BILL H      3.75
340     DATA JONES BOB R       2.00
350     DATA BROWN MARY A      2.00
360     DATA SMITH GLEN C      2.90
370     DATA TAYLOR RALPH E    3.50
380     DATA JONES BONNIE R    3.50
390                                     !
400   END
```

Notice that the inverse permutation of pointer array B is computed. The use of this permutation allows two sorts to be performd on the original object array thereby eliminating the need for a duplicate array. If the program is run, the results shown next are obtained.

```
THE ORIGINAL SEQUENCE IS:

SMITH BILL H      3.75
JONES BOB R       2.00
BROWN MARY A      2.00
SMITH GLEN C      2.90
TAYLOR RALPH E    3.50
JONES BONNIE R    3.50


REORDERED BY NAME:

BROWN MARY A      2.00
JONES BOB R       2.00
JONES BONNIE R    3.50
SMITH BILL H      3.75
SMITH GLEN C      2.90
TAYLOR RALPH E    3.50


REORDERED BY GPA:

SMITH BILL H      3.75
JONES BONNIE R    3.50
TAYLOR RALPH E    3.50
SMITH GLEN C      2.90
BROWN MARY A      2.00
JONES BOB R       2.00
```

It is important to note that several arrays can be reordered by the same pointer array. The program shown next utilizes this feature. Several "parallel" arrays are reordered according to the contents of a single pointer array.

```
10   ! ************************************************************
20   ! ***         This program demonstrates the use          ***
30   ! ***           of the MAT REORDER statement.             ***
40   ! ************************************************************
50                                   !
60     OPTION BASE 1                 !Select OPTION BASE.
70     DIM Id_no(4),Jobs$(4),Clock_no(4),Order(4) ! Dimension source and
80                                   ! pointer arrays.
90      MAT READ Id_no               ! Read appropriate data into arrays.
100     MAT READ Jobs$               ! Data is in Lines 230,240,250.
110     MAT READ Clock_no
120                                  !
130  ! ************************************************************
140  ! ***       Print the heading and reorder the arrays.     ***
150  ! ************************************************************
160                                  !
170  PRINT TAB(5);"ID NUMBER";TAB(20);"JOB";TAB(30);"CLOCK NUMBER";LIN(2);
180  CALL Print(Id_no(*),Jobs$(*),Clock_no(*)) ! Call the Print subroutine
190                                  ! using the array data as parameters.
200       PRINT LIN(1)
210                                  !
```

```
220      MAT SORT Id_no(*) TO Order ! Sort ID numbers to pointer array.
230                                 !
240      MAT REORDER Jobs$ BY Order
250      MAT REORDER Id_no BY Order ! Reorder all arrays by the same pointer.
260      MAT REORDER Clock_no BY Order
270                                 !
280      PRINT "     ------------------------------------ ";LIN(1)
290      CALL Print(Id_no(*),Jobs$(*),Clock_no(*)) ! Call subroutine
300                                 ! for printing the reordered arrays.
310      DATA 2325,4157,9020,1025
320      DATA MANAGER,GUARD,TYPIST,NURSE ! Data is entered in the proper order
330      DATA 307,118,476,534       ! to create "parallel" arrays.
340                                 !
350     END
360  !
370  ! ****************************************************************
380  ! ***   This subroutine prints data in the specified order.   ***
390  ! ****************************************************************
400                                 !
410     SUB Print(Id_no(*),Jobs$(*),Clock_no(*))
420       FOR I=1 TO 4
430         PRINT TAB(6);Id_no(I);TAB(19);Jobs$(I);TAB(33);Clock_no(I);
440       NEXT I
450     SUBEND
```

Notice that the individual object arrays are reordered by the same pointer array. If the program is run, the results shown next are obtained.

```
ID NUMBER      JOB       CLOCK NUMBER

  2325       MANAGER         307
  4157       GUARD           118
  9020       TYPIST          476
  1025       NURSE           534

 ------------------------------------

  1025       NURSE           534
  2325       MANAGER         307
  4157       GUARD           118
  9020       TYPIST          476
```

Computer Museum

MAT SEARCH Statement

Searching Numeric Arrays

The purpose of the MAT SEARCH statement is to provide information about user-defined conditions within an array. This information is returned to a variable for recall and examination.

Syntax:

MAT SEARCH source array (location specifier), condition ; variable
[, starting address]

The source array represents the array in which searching is performed. The location specifier defines the locations within the souce array which are searched. No location specifier need be included if the source array is one-dimensional. The condition is that value or location for which you are searching. When the condition is satisfied, the memory content or location which satisfies it is returned to the variable. The optional starting address specifier allows you to select a location within the range defined by the location specifier at which you want searching to begin. Numeric searching comparisons are performed in the INTEGER mode for integer-precision arrays and in the REAL mode for short or real-precision arrays.

The conditions available in the MAT SEARCH process are entered in the following form:

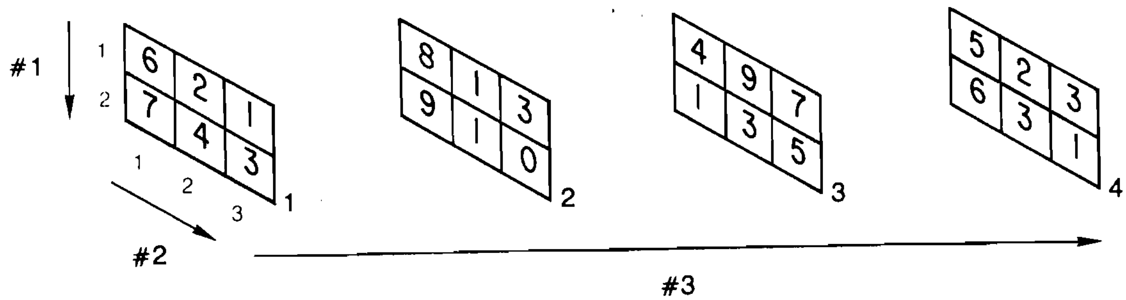
LOC (relational operator – expression)
#LOC (relational operator – expression)
MAX
MIN

The list of relational operators includes: >,<,=,#,>=,<=,<>. For an integer search, the expression is rounded before the comparison is performed. A LOC condition causes the search process to scan the specified locations until it finds the first value which satisfies the condition. The default value for a relational operator is =.

A #LOC condition causes the search process to scan the specified locations and return the total number of locations whose contents satisfy the condition.

MAX and MIN conditions do not include an argument in parentheses because they automatically cause the search process to scan all specified locations to find the maximum and minimum values. To return the result of a MAX or MIN search, a string variable is used with string data, and a numeric variable with numeric data.

To demonstrate the MAT SEARCH process, assume that array A is a three dimensional numeric array containing random numbers as shown below.



ARRAY A WITH SEARCH LOCATIONS

Assume that the shaded locations are to be searched until one is found which contains a value greater than 5. Let B represent the variable to which the result is returned.

The shaded memory locations must be described by a location specifier. Since they lie along the third dimension, the records in which they lie are defined accordingly. The correct location specifier is:

```
(1,3,*)
```

Note that the location specifier is written in the same format as a key specifier for a MAT SORT statement. The subscripts are written in the same order as the array dimensions (the numbered arrows). The first two subscripts define the upper right location, and the asterisk indicates that the third subscript is varied over its range of values to describe all three records.

The correct search statement for this example is

```
MAT SEARCH A(1,3,*),LOC(>5);B
```

where A is the source array, (1,3,*) defines the locations to be searched, LOC(>5) is the condition, and B is the variable to which the result is returned.

After execution of the search statement, the number 3 is returned to the variable B. This is the first record whose specified location satisfies the condition(i.e., it contains a value greater than 5). Searching begins at the location in that record described by the smallest number in the range of varied subscripts and continues to the largest.

If a condition is not satisfied upon completion of the searching process, a value one greater than the upper limit of the varied subscript is returned to the numeric variable. For example, if the specified locations in the previous example are searched for a value greater than 8, none is found. Therefore, a value of 5 representing a number one greater than the record containing the last searched location is returned to B. Note that if the upper limit of the varied subscript is 32 767, the value returned by an unsuccessful search is − 32 768.

The location specifier initially defines the range of locations to be searched. If you do not wish to search the entire range, a starting address specifier can be used to designate where the search is to begin. In the previous example, assume that the search process is to scan only the last three records. The correct search statement is

```
MAT SEARCH A(1,3,*),LOC(>5);B,2
```

where the number 2 directs the search to begin at the specified location in the second record and proceed to the last record. Upon execution, the number 3 indicating the third record is returned to the variable B because the content of its specified location is the first to satisfy the condition.

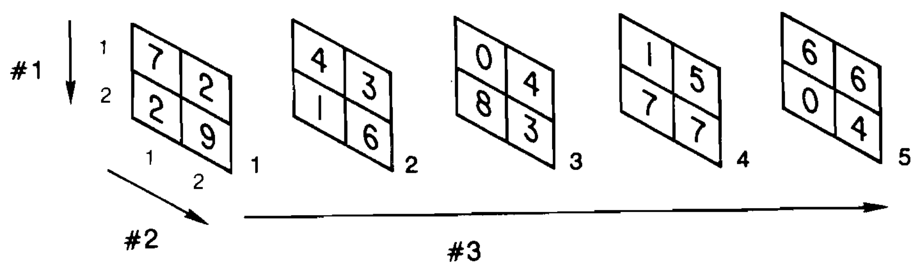
The following program is included to demonstrate a search for maximum and minimum values and number of occurences. The array Numbers is filled with random numbers from which the maximum and minimum values are selected.

```
MAXIMUM = 9          MINIMUM = 1
```

If this program is run, the following results are obtained.

```
10    ! **************************************************************
20    ! ***      This program demonstrates the use of the          ***
30    ! ***      MAT SEARCH statement to find maximum and          ***
40    ! ***           minimum values within an array.              ***
50    ! **************************************************************
60                                        !
70      OPTION BASE 1                     ! Select the OPTION BASE.
80      DIM Numbers(11)                   ! Dimension source array.
90                                        !
100         FOR I=1 TO 11                 ! Establish input loop.
110           DISP "NUMBER";I;            ! Display entry number.
120           INPUT Numbers(I)            ! Input data to source array.
130         NEXT I                        ! Loop to input next data item.
140                                       !
150   ! **************************************************************
160   ! *** The data used in this example: 6,1,9,2,8,3,8,9,1,7,5 ***
170   ! **************************************************************
180                                       !
190       MAT SEARCH Numbers(*),MAX;Max   ! Search for maximum value.
200       MAT SEARCH Numbers(*),MIN;Min   ! Search for minimum value.
210                                       !
220       PRINT "MAXIMUM =";Max,"MINIMUM =";Min
230     END
```

Normally, a LOC search ends at the first location which satisfies the specified condition. However, additional satisfactory values may exist beyond that location. By setting the starting address to be one greater than the content of the variable, a search can be continued from the first location which satisfied the LOC condition. This automatically continues a search from where a previous search left off. All satisfactory values can be obtained in this way. As an example, assume that array B is a three dimensional numeric array containing random numbers as shown.



ARRAY B

Assume that the shaded memory locations are to be searched for values greater than 2. A single MAT SEARCH scan would stop at the second record since it is the first one encountered whose memory location satisfies the condition.

However, by constructing a loop and using the proper starting address specifier, the search can be made to continue through the range of specified locations. The following program demonstrates this feature.

```
10   ! *******************************************************
20   ! ***       This program demonstrates the use         ***
30   ! ***          of the MAT SEARCH statement.           ***
40   ! *******************************************************
50                                  !
60     OPTION BASE 1                ! Select OPTION BASE.
70     DIM B(2,2,5)                 ! Dimension numeric array B.
80                                  !
90       MAT READ B                 ! Read data into array B. Data is
100                                 ! found in Line 240.
110      C=0                        ! Initialize C so first search begins
95                                  ! at record 1 of array B.
120                                 !
```

```
130 ! ************************************************************
140 ! ***  Search for locations of numbers greater than 2.  ***
150 ! ************************************************************
160                                   !
170 Search: MAT SEARCH B(1,2,*),LOC(>2);C,C+1
180                                   !
190        IF C+1>6 THEN 260          ! Test: Have all locations been searched?
170                                   !       If yes, stop.
200                                   !
210        DISP C;                    ! Display most recent search results.
220        WAIT 500                   ! Wait slightly before further display.
230        IF C<5 THEN Search         ! Test: Has search reached last location?
220                                   !       If not, continue.
240                                   !
250        DATA 7,4,0,1,6,2,3,4,5,6,2,1,8,7,0,9,6,3,0,4
260     END
```

The variable C is initially set to zero. Line 90 begins the search routine. Since C was set to zero, the starting address specifier (C + 1) directs the search to begin at record one. Line 110 tests to see if the specified locations have all been searched. (Remember, if all locations have been searched, a number one greater than the last record searched is returned to the variable. In this case, that number is six). If all locations have not yet been searched, line 120 displays the contents of C (it now contains a satisfactory value). If C contains a value less than 5, the search is not finished and line 140 directs the program to search again. Due to the nature of the starting address specifier, the search begins this time at the next memory location beyond that which satisfied the condition previously. In other words, the search resumes where it left off. If you run this program, the following should be displayed:

```
2  3  4  5
```

Searching String Arrays

Searching string arrays follows the same format as searching numeric arrays.

Syntax:

MAT SEARCH source array (location specifier), condition; variable
[, starting address]

The source array, location specifier, condition, variable, and starting address specifier serve the same functions for string arrays as for numeric arrays. If a relational operator is included, the LEXICAL ORDER IS statement determines the ordering of the string data. The order can be STANDARD (according to the ASCII table) or user-defined (see Appendix B).

For example, assume array List$ contains a list of names and dollar amounts. The program shown next inputs the data to the source array. It then searches for a particular name and outputs the corresponding dollar amount.

```
10   ! ***********************************************************
20   ! ***         This program demonstrates the use of        ***
30   ! ***              the MAT SEARCH statement.              ***
40   ! ***********************************************************
50                                   !
60    OPTION BASE 1                  ! Select OPTION BASE.
70    DIM List$(4)[20]               ! Dimension source array.
80                                   !
90      FOR I=1 TO 4                 ! Establish loop for data entry.
100       READ List$(I)              ! Read data into List$.
110     NEXT I                       ! Read next data item.
120                                  !
130     MAT PRINT List$              ! Output the original list.
140                                  !
150  ! ***********************************************************
160  ! ***         Search List$ for a particular name.         ***
170  ! ***********************************************************
180                                  !
190     MAT SEARCH List$(*)[1,5],LOC("HAYS ");B   ! Search the proper
200                                               ! portion of each
210                                               ! string.
220                                  !
230     PRINT List$(B)[1,5];":  ";List$(B)[15,18] ! Output specified
240                                               ! name and dollar
250                                               ! amount
260                                  !
270     DATA BROWN  FRANK  $100  ,SMITH  BOB   $75
280     DATA HAYS   SUE    $150  ,JONES  ED    $200
290   END
```

In this program a MAT SEARCH is used to find the string which contains the required name. Once that string is found, the portion of it containing the dollar amount is displayed. Note that the substring specifier is used in the search and display statements. If you run this program, the following results are obtained.

```
$150
```

```
BROWN  FRANK  $100
SMITH  BOB    $75
HAYS   SUE    $150
JONES  ED     $200

HAYS :  $150
```

MAX and MIN values can also be obtained from a string search as demonstrated by the program shown next.

```
10   ! ***********************************************************
20   ! ***  This program demonstrates use of the MAT SEARCH   ***
30   ! ***         statement in conjunction with strings.      ***
40   ! ***********************************************************
50                                      !
60     OPTION BASE 1                    ! Select the OPTION BASE.
70     DIM Stringsearch$(3)[8]          ! Dimension the string array.
80                                      !
90       MAT READ Stringsearch$         ! Read data into the array.
100        DATA "ABC","BCD","DEF"
110                                     !
120      MAT SEARCH Stringsearch$(*),MAX;A$
130                                     ! Search string array for
140                                     ! maximum string value.
150                                     !
160      MAT SEARCH Stringsearch$(*),MIN;B$
170                                     ! Search string array for
180                                     ! minimum string value.
190                                     !
200      PRINT "MAXIMUM = ";A$,"MINIMUM = ";B$
210                                     ! Output the results.
220    END
```

The array Stringsearch$ is filled with string data. MAT SEARCH statements are then used to find the maximum and minimum values of the data. Note that the search statements use the current lexical order of the data to perform the search comparisons (see LEXICAL ORDER IS statement). If this program is run, the results shown next are obtained.

```
MAXIMUM = DEF        MINIMUM = ABC
```

Extended Character Sets

Statement Syntax

LEXICAL ORDER IS lexical order designator

lexical order designator: an integer array containing a local language collating table or the word **STANDARD**

LEX (string expression 1, string expression 2)

Dot Matrix – All items in dot matrix must be entered as shown.

[ ] – All items in brackets are optional.

... – Three dots indicate that successive parameters are allowed, when each is separated by a comma.

Chapter 3

Introduction

The statements described in this chapter are used as programming aids to define and access comparison and case functions on the ASCII and Roman Extension Character Sets. Each of these statements or functions can be executed from within a program or from the keyboard. Examples of each statement are included to demonstrate its use.

LEXICAL ORDER IS Statement

The LEXICAL ORDER IS statement allows you to select the collating sequence for string sorts and determine the results of string comparisons. It also determines the results of UPC$ and LWC$ transformations for Roman Extension characters.

Syntax:

`LEXICAL ORDER IS` lexical order designator

The lexical order designator must be an integer array containing a local language collating table, or the word STANDARD which indicates the standard ASCII character sequence. The following local language collating tables are available on the Lexical Tables Cartridge for use with the Advanced Programming ROM.

FRENCH	SWEDSH (Swedish)
GERMAN	SPANSH (Spanish)

Refer to Appendix B for information on the lexical order collating tables.

The STANDARD lexical order designator selects the ASCII table as the collating sequence (i.e., string sorting is performed according to ASCII values). STANDARD is the default lexical order designator at power ON and after SCRATCH A. As such, you need not include a LEXICAL ORDER IS statement if you wish to use ASCII as the collating sequence. RESET does not clear the current lexical order. However, if you wish to change to STANDARD from some other collating sequence, you can do so by entering:

```
LEXICAL ORDER IS STANDARD
```

The program shown next demonstrates how a local language collating sequence is selected.

```
10  INTEGER A(400)
20  ASSIGN #1 TO "FRENCH"
30  MAT READ #1;A
40  LEXICAL ORDER IS A(*)
```

Line 10 dimensions an integer array to hold the specified collating table. Here, A is chosen as the array and is dimensioned to a length of 400 which accommodates any of the local language character collating tables listed previously.

Line 20 assigns #1 to the specified file on the cartridge; here, FRENCH.

Line 30 reads the contents of the file (the specified collating table) into the array.

Finally, line 40 establishes the contents of the array (which now contains the collating table) as the lexical order.

Note that when a LEXICAL ORDER IS statement is executed, the information in the array is transferred into an internal buffer. This allows you to use the array for something else. Care must be taken, however, to insure that enough memory exists for the buffer when the statement is executed. Otherwise, an error occurs.

The LEX Function

The LEX function allows you to compare two strings and determine which precedes the other according to the current lexical order. The LEX function can be executed from the keyboard or from within a program.

Syntax:

LEX(string expression 1, string expression 2)

The LEX function begins comparing string characters until a difference is found or until one or both strings terminates. The function returns the number −1 if string 1 precedes string 2. If the current lexical order cannot distinguish between the two strings, 0 is returned. The number 1 is returned if string 2 precedes string 1.

The program shown next demonstrates the LEX function. Two strings are compared and the result is displayed.

```
10  A=LEX("XEF","XBC")
20  DISP A
30  END
```

Notice that the program utilizes the default STANDARD lexical order. The LEX function compares the strings up to their second characters since these are the first respective characters that differ. Upon execution, the number 1 is displayed because B (string 2) precedes E (string 1) according to the current lexical order (ASCII).

The previous example program compared strings of equal length. The example program shown next demonstrates a comparison of strings of unequal length.

```
10  A=LEX("A","A ")
20  B=LEX("A","AB")
30  DISP A,B
40  END
```

In this case, both A and B are −1. In both comparisons, the first string terminates before the second.

Uppercase and Lowercase Functions

The Advanced Programming ROM extends the capabilities of the mainframe uppercase (UPC$) and lowercase (LWC$) functions to correctly handle the upper and lowercase transformations of the various local language characters.

The results of the UPC$ and LWC$ functions are determined by the entries in the upper and lowercase sections of the lexical order array for Roman Extension characters (refer to Appendix B). The upper and lower cases of characters in the main ASCII set are fixed.

Descriptions of the French, German, Spanish and Swedish upper and lowercase transformations which are provided in the respective files on the cartridge are explained in Appendix B.

The UPC$ function can be used to obtain a proper LEX comparison if it is not known if the two strings are in the same case (i.e., if "ABC" should equal "abc"). Example:

```
X = LEX (UPC$ (A$),UPC$ (B$) )
```

NOTE

If a program containing UPC$ or LWC$ statements is STORE'd in a System 45B which has an Advanced Programming ROM, and is then LOAD'ed into a System 45B which does not have an Advanced Programming ROM, a MISSING ROM error message occurs.

If a program is STORE'd in a System 45B which has no Advanced Programming ROM, and is then LOAD'ed into a System 45B which has an Advanced Programming ROM, no error message results. However, if any lexical statements are added, the current UPC$/LWC$ lines must be re-STORE'd in order for the Advanced Programming ROM to affect them.

It is suggested that you SAVE rather than STORE programs which will be switched between those System 45B Desktop Computers which have an Advanced Programming ROM and those which do not.

Statement Summary

MAT SORT source array (key specifier) [[substring specifier]] [DES] [, (secondary key specifier) [[substring specifier]] [DES]] [TO pointer array]

Arranges specified numeric or string data in ascending (lexical) or descending (reverse lexical) order. Optional pointer array stores sorting order.

MAT REORDER object array BY pointer array [, dimension specifier]

Reorders the object array according to the order specified by the pointer array. Optional dimension specifier selects the object array dimension along which reordering is performed.

MAT SEARCH source array (location specifier), condition; variable [, starting address]

Searches the specified condition of the source array for the specified condition. The satisfactory condition is returned to the variable. The optional starting address specifier selects the location at which the search begins.

LEXICAL ORDER IS lexical order designator

Establishes the particular collating table by which sorting and searching is performed.

LEX(string expression 1, string expression 2)

Compares two strings and determines which precedes the other according to the current lexical order.

UPC$ and LWC$

Extends the capabilities of the uppercase and lowercase transformations to the various local language characters.

Appendix A

User-Defined Lexical Order

It is recommended that you use the existing collating tables as they appear on the tape cartridge. However, an experienced programmer can create a lexical order table to accommodate his particular application. The following information describes the procedures involved. A knowledge of binary and octal numbers is necessary to utilize these procedures.

To modify a local language collating table, it is first input to an integer array. Once in the array, the specified table can be accessed for modifications.

Along with the local language tables, a special ASCII table ("ASCII") is included on the cartridge for the purpose of modifying the ASCII collating sequence. The LEXICAL ORDER IS STANDARD statement automatically uses the standard ASCII sequence. But in order to modify it, the "ASCII" file on the cartridge may be input to an integer array in the same manner as is a local language table. Example:

```
10   ! **************************************************
20   ! ***     This program segment illustrates the     ***
30   ! ***          LEXICAL ORDER IS statement.         ***
40   ! **************************************************
50
60     OPTION BASE 1                    ! Select OPTION BASE.
70     INTEGER B(400)                   ! Dimension integer array.
80       ASSIGN #1 TO "ASCII"           ! Select ASCII file.
90       MAT READ #1;B                  ! Read in the ASCII table.
100      !
110      ! User modifications.
120      !
130      LEXICAL ORDER IS B(*)          ! Establish contents of
140                                     ! array B as lexical order.
150    END
```

Line 40 inputs the "ASCII" file contents to the integer array B. Lines 50 through 70 represent the user modifications to the ASCII table and line 80 establishes the modified table as the lexical order. Notice that in order for the table in the given array to dictate lexical comparison results, a LEXICAL ORDER IS statement must be included after any modifications to the array have been made.

Appendix B

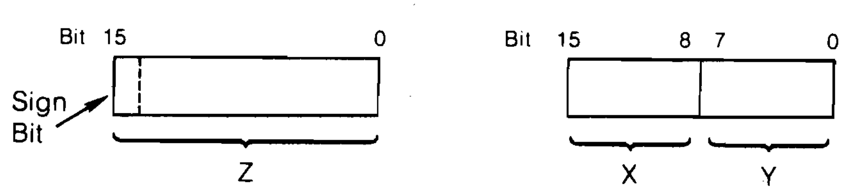
Once a LEXICAL ORDER IS statement has been executed, the integer array can be used for other purposes. Be sure, however, to save all necessary modifications on the tape cartridge before using the array for other purposes. The program lines shown next can be used to maintain a copy of the lexical table modifications for the previous example.

```
100  ASSIGN #3 TO "NEW"
110  MAT PRINT #3;B
```

The lexical order tables give you the capability to define the collating sequence (collating section), define the uppercase/lowercase transformation of Roman Extension characters (UPC/LWC section), and prescribe "special handling" of certain characters (mode section). The lexical table with its individual sections is an integer array organized as shown in the following illustration.

1	Total Length
2	Mode Section Length
3 … 258	Collating Section (Length = 256)
259 … 354	UPC/LWC Section (Length = 96)
355 … 354 + N	Mode Section (Length = N)

Each element of the lexical table consists of 16 bits. This arrangement can accommodate one number or an integer combination of two (unsigned) numbers as shown next.



If one number is stored, a binary two's complement storage format is used. In this manner, each element has the ability to contain two distinct values.

The first element of the array contains the complete length of the lexical table. This length is 354 plus the number of elements (N) needed in the user-defined mode section (see explanation of mode section).

The second element of the array contains N, the length of the mode section of the table. Once the length of the mode section is defined, the first two elements of the array table can be filled with the proper information.

Elements 3 – 258 of the table contain the actual collating sequence and the appropriate pointers into the mode section, if one exists.

Elements 259 – 354 define the uppercase/lowercase transformations for characters in the Roman Extension Set (refer to the ASCII and Roman Extension chart at the end of this appendix).

The mode section consists of elements 355 to (354 + N) and provides facilities for handling certain special case characters.

Collating Sequences

When the lexical order is STANDARD, the ASCII codes define the collating sequence. Thus, A lexically precedes B (i.e., LEX("A", "B") is −1) because the ASCII code for A is 65, the ASCII code for B is 66, and 65 is less than 66.

For computer processing, each character has been assigned an ASCII code. The ASCII codes are fixed, however, and cannot be changed to reflect a new user-defined lexical order. The lexical order tables give you the capability of redefining the lexical order by assigning each character a sequence number. Characters are still represented internally by their ASCII codes, but a lexical comparison compares sequence numbers rather than ASCII codes. Now, LEX ("A", "B") is −1 if the sequence number of A is less than the sequence number of B. For example, the strings "ABC" and "XYZ" are stored in the computer as strings of ASCII codes:

ASCII code for A	ASCII code for B	ASCII code for C

ASCII code for X	ASCII code for Y	ASCII code for Z

The calculation of LEX ("ABC","XYZ") involves a comparison of sequence numbers:

Sequence # for A	Sequence # for X
Sequence # for B	Sequence # for Y
Sequence # for C	Sequence # for Z

By properly assigning each character a sequence number, you can define any lexical ordering of the characters.

Collating Section

The collating section of the lexical order table (elements 3-258) defines the actual collating sequence by assigning a sequence number to each character. There are 256 possible 8-bit character codes (refer to the ASCII and Roman Extension chart). The user-defined sequence number for the character with ASCII code 0 goes in element 3 of the lexical table (element 1 of the collating section), sequence number for character with ASCII code 1 goes in element 4 of the lexical table (element 2 of the collating section), and so on.

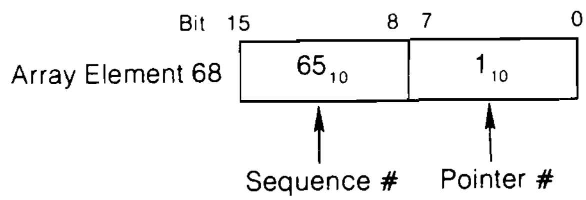
For example, the letter A has ASCII code 65_{10}. When an A is encountered, the manipulation routine goes to the 66th position of the collating section to fetch a sequence number. Note that the sequence number is fetched from the collating section element number equal to the ASCII value of the character +1. This is because the first character in the collating section is null, and its ASCII value is zero. Usually, the sequence number equals the ASCII value. However, if a different collating order is selected, the sequence number may differ from the ASCII value.

In addition to the sequence number, each entry of the collating section may contain a pointer into the mode section of the lexical table. If no pointer is specified, 0 is entered. The mode section contains instructions governing special case letters. The pointer number indicates in which position of the mode section a special case is handled.

The sequence number and mode pointer number for each character are combined to form a single integer entry in the lexical table:

Bit	15 8	7 0
	Sequence #	Mode Pointer #

Assume that A is a special case letter with the sequence number 65. Also, assume that the special case is handled in position 1 of the mode section. The collating section entry containing this information is represented by:



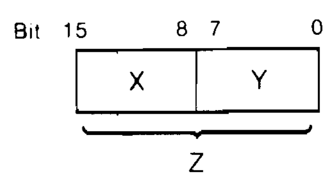
The actual content of this entry is $16\ 641_{10}$. The number is obtained as shown next.

	65	1
Binary	01000001	00000001
Octal	4 0	4 0 1_8
Decimal	1 6	6 4 1_{10}

Both the sequence number (65) and the pointer number (1) are combined to form one integer value within the table element, in this case, $16\ 641_{10}$.

An alternate method of converting the binary and octal elements of the lexical table entries to integer form is given below. For example, if the sequence number you are using is greater than 127, you may find it more convenient to use the formulas shown next.

X and Y are each 8-bit unsigned integers.
Z is a 16-bit signed integer formed by concatenating the bit patterns of X and Y.



1. Given X and Y, calculate Z:
   Z = X*256 + Y − (X>127)*65 536
2. Given Z, calculate X and Y:
   X = INT(Z/256) + (Z<0)*256
   Y = Z MOD 256

Uppercase/Lowercase Sec

Elements 259-354 of the lexical table are used for uppercase/lowercase information for characters residing in the Roman Extension Set. Data contained in these elements is in the form:

Bit 15 — 8	7 — 0
Uppercase Character	Lowercase Character

The UPC/LWC section contains an element for each character with an 8-bit code of 160 through 255. The first element of the UPC/LWC section (element 259 of the lexical table) contains the upper/lowercase for the character with 8-bit code 160, the next element contains the transformations for the character with 8-bit code 161, and so on.

The purpose of this section of the lexical table is to allow you to define the upper and lowercase versions of each letter in the Roman Extension Set for use by the UPC$ and LWC$ functions. All characters in the main ASCII table have fixed upper and lowercases, and as such, are not changed. If both the uppercase and lowercase 8-bit codes in the table are 0 for a given character, the character is unchanged by the UPC$ and LWC$ functions.

For example, if a word containing a lowercase é (code 197_{10}) is to be converted by French conventions, the accent is omitted when the letter is converted to uppercase. That is, é→E. But under German conventions, an umlaute in the letter remains so that ä→Ä. Therefore, for French, the 38th entry of the uppercase/lowercase section (296th entry of the table) is

	E	é
Array Element 296	69_{10}	197_{10}

while the 45th entry in the German uppercase/lowercase section is

	Ä	ä
Array Element 303	216_{10}	204_{10}

Mode Section

The mode section (starting at element 355) of the lexical table handles special case characters. The three special cases shown next are possible.

- Accent Priority
- 1 For 2 Character Replacement
- 2 For 1 Character Replacement

Accent Priority

You may specify the priority of letters with accent marks. Example:

e<ê<ë<é<è

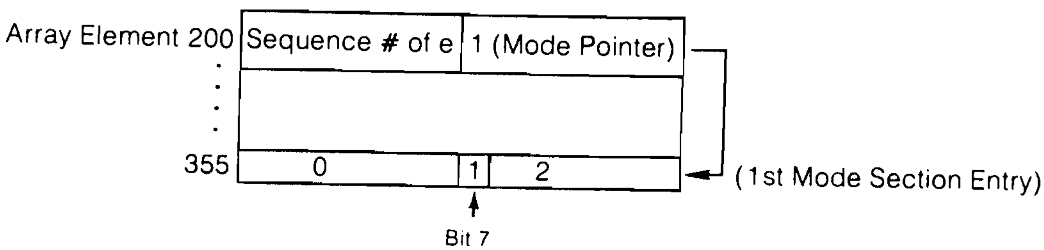
The symbol (<) means "precedes alphabetically". In this case, these letters have the same sequence number in the collating section, but different pointer numbers. That is, each has an individual entry in the mode section.

An accent priority requires only one entry in the mode section (some special cases require more than one). The format of the mode section entry for this special case is:

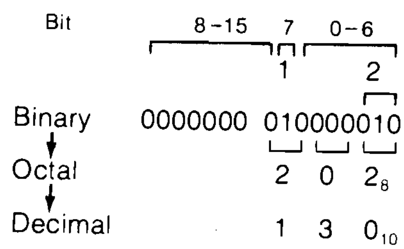
Bit 15 8	7	6 0
0	1	Priority

The zero indicates that the eight bits of this special case mode entry are not used, and must contain zero (0) so as not to be confused with one of the other special cases. A 1 must be present in bit seven to indicate that this special case is an accent priority. The remaining bits describe the user-defined priority number.

For example, assume that the letter é in the collating section has a pointer number which specifies the first entry of the mode section (or element 355 of the table). Also, assume that this mode entry assigns a priority of 2 to the letter é. The collating and mode sections contain the information shown next.



The following scheme is used to find the correct mode section entry.



130_{10} is the proper entry for the mode section element. Bits 6 through 0 are used to describe the priority number. Bit 7 is always 1 for an accent priority special case. Bits 15 through 8 are not used, but must contain zeros to insure proper results.

Note that the value of an accent priority entry in the mode section is always 128 plus the assigned priority. This means that the maximum accent priority is 127.

Accent priority mode section entries assign a relative priority to characters having the same sequence number. The accent priority is used only to distinguish two otherwise identical strings which differ only in their accent marks.

For example, assume ê and ë have the same sequence number and each has been assigned an accent priority; ê has priority 5, and ë has priority 2. The routine to calculate LEX ("ê","ë") first compares the sequence numbers of ê and ë.

Since the sequence numbers are the same (and each character has been assigned an accent priority), the accent priorities are next compared. Five is greater than 2, so LEX ("ê","ë") is 1 (i.e., ê> ë, or ê comes after ë alphabetically).

A character which is not assigned an accent priority in the mode section lexically precedes any other character with the same sequence number and which has an accent priority. For example, assume e and ê have the same sequence number, but that ê has been assigned an accent priority in the mode section while the mode pointer of e is 0. Since e has not been assigned an accent priority, LEX ("e","ê") is −1, or e precedes ê alphabetically.

In the following example, the two strings have the letter e (but with different accent marks) as the first character. Since the last characters of each string differ, a mismatch is encountered without using any accent priority information from the mode section.

LEX ("êz"," ëa") = −1

In the next example, both letters in one string match those in the other; the only difference being the accent marks on the first letters. In this case, the strings match and accent priority information is used to make a distinction. If the accent priority for ë has been specified to be less than the accent priority for ê, the results would be:

LEX ("êb"," ëb") = 1

1 For 2 Character Replacement

Another special case that you can specify is a 1 for 2 character replacement. This specifies that for collating purposes, a given combination of two characters is to be treated as a single character. That is, two characters are assigned a single sequence number.

An example of this are the letters "CH" together in the Spanish standard collating sequence. The correct alphabetical sequence in Spanish is A, B, C, CH, D... All words beginning with C followed by any letter other than H come before words beginning with C followed by H. Therefore, the two letters CH can be taken as a single letter coming between C and D in the alphabetical sequence.

The string "CHA" is to be stored as

ASCII Code for C	ASCII Code for H	ASCII Code for A

but for collating purposes, the sequence numbers fetched for this string would be:

Sequence # for CH
Sequence # for A

The mode section format is:

Seq. # of 2-cha combination	ppercase of d character
Seq. # of 2-char combination	wercase of d character
0	128_{10}

The front, or upper part of the first (and possibly second) entry, contains the sequence number of the 2-character combination. The last, or lower part of the entry, contains the ASCII value of the 2nd character.

Usually an entry for both the uppercase and lowercase versions of the 2nd character are specified. This is to catch both occurrences of the given character combination; for example, CH and Ch. However, this is not required. If the special 1 for 2 character replacement is to specify CH only (and not Ch), the lowercase entry would be omitted.

In the case that both an uppercase and lowercase entry appears, the sequence numbers in the upper part can be the same in both entries. They are the same in the Spanish table. However, different sequence numbers for each combination are acceptable also.

No matter how many entries are given, the last entry for this special case is equal to an octal 200, or a decimal 128. It is a terminator that indicates there are no further character combinations.

When a lexical calculation on a string encounters a character with a mode pointer to a mode entry of this type, the next character in the string is compared one-by-one with the possible second characters given in the mode entries. If a match is found, the two characters together are represented by the single sequence number given in the corresponding mode table entry. If no match is found, the first character is assigned the sequence number given in its collating section entry and processing continues with the next character in the string. Note that "don't care" characters in the string are not regarded as such for this type of replacement.

For example, the sequence numbers fetched for the string "CAH" (even if "A" is a "don't-care" character) would be:

Sequence # for C
Sequence # for A
Sequence # for H

Note that this special case does not cause any actual substitutions to be made in a string containing CH; CHA is still stored as:

ASCII Code for C	ASCII Code for H	ASCII Code for A

This means that for collating purposes, the character pair CH causes a single sequence number to be fetched.

Consider again the example of CH for Spanish. Assume that the letter C has sequence number 67 and D has sequence number 68. To include the combination CH, a new sequence number must be created between 67 and 68. To do this, the collating sequence is rearranged so that C has sequence number 67 and D has 69. Sequence number 68 is reserved for CH and the remaining sequence numbers are adjusted accordingly.

Assume that the 4th, 5th, and 6th elements in the mode sequence contain the information governing the CH special case. The mode sequence entries for this example are shown next.

Array Element 70	67	4	
71	69		
⋮			
358	Sequence # for CH	H	← (4th Mode Section Entry)
359	Sequence # for CH	h	
360	0	128_{10}	

Numerically, the mode section entries for this example would contain:

68	72
68	104
0	128

Note that 72 is the ASCII value for uppercase H and 104 is the ASCII value for lowercase h.

Whenever a C is encountered in collating and H or h immediately follows in the string, the pair of characters (CH or Ch) is assigned a single sequence number. Processing the string "CHA" would fetch the sequence numbers:

Sequence # for CH
Sequence # for A

In the original example, the actual contents of the mode sequence entries are:

4th Mode Section Entry:

	68	72
Binary ↓	01000100	01001000
Octal ↓	4 2	1 1 0_8
Decimal	1 7	4 8 0_{10}

5th Mode Section Entry:

	68	104
Binary ↓	01000100	01101000
Octal ↓	4 2	1 5 0_8
Decimal	1 7	5 1 2_{10}

Therefore, lexical table element 358 (1+1+256+96+4) contains 17 480_{10}, element 359 contains 17 512_{10}, and element 360 contains the terminator value 128_{10}.

It is possible to specify additional character replacements for the same character. For example, if the combination CZ were to follow CH in the previous example, the mode section would be expanded to include the additional characters as shown below.

Sequence # for CH	H
Sequence # for Ch	h
Sequence # for CZ	Z
Sequence # for Cz	z
0	128

The terminator does not occur until all additional characters have been included.

2 For 1 Character Replacement

The third special case is a 2 for 1 character replacement. This type of mode entry specifies that a single character is to be assigned two sequence numbers for lexical comparisons. For example, the letter Ä in German is alphabetically equivalent to AE (or Ae). In a lexical comparison, Ä is assigned two sequence numbers; the sequence number for A and the sequence number for E (or e).

In general, if the mode entry specifies that a character is to be assigned two sequence numbers, the first sequence number is given in the normal collating section of the table. The second sequence number is given in the mode table entries. The format of the mode table entries is shown next.

Bit 15 … 8	7 … 0
Sequence # of 2nd character (upper)	0
Sequence # of 2nd character (lower)	0

Since there is only one number required for this case, the second portion of the element (bits 7-0) is not used, and must be set to zero to avoid confusion.

Note that mode entries of this special case require no terminator element. For example, assume Ä is a special case character that is alphabetically equivalent to AE (or Ae), and that the special case for Ä is handled in the first entry of the mode section. The collating section entry for Ä (which is character code 216) and the mode section entry are shown next.

Array Element			
219	Sequence # for A	1 (Mode Pointer)	
⋮			
355	Sequence # for E	0	← (Ist Mode Section Entry)
356	Sequence # for e	0	

Assuming that the sequence numbers of A, E, and e are the same as their ASCII values, the following illustration represents the numerical contents of the table entries.

219	65_{10}	1	16641_{10}
⋮			
355	69_{10}	0	17664_{10}
356	101_{10}	0	25856_{10}

Note that the sequence number for Ä in the collating section is equal to the sequence number for A. Differentiation occurs in the mode section information. That is, A probably will have no mode section pointer, while Ä will. The sequence difference between A and Ä, then, is handled by Ä's entry in the mode section.

The choice between lower and uppercase for the second sequence number is determined by the case of the character immediately following the special case letter. For ÄN, A is equivalent to AE. For Än, A is equivalent to Ae. The character Ä alone is equivalent to AE.

This special case does not cause any actual substitutions to be made in a string. In the previous example, the string "Än" is stored as:

ASCII CODE for Ä	ASCII Code for n

For collating purposes, the sequence numbers fetched for the string "Än" would be:

Sequence # for A
Sequence # for e
Sequence # for n

"Don't Care" Characters

There is a fourth special case which doesn't require a mode section entry. This is the "don't care" special case where the specified character is to be ignored for collating purposes. An example of such a case is the hyphen in hyphenated words.

A character is ignored alphabetically if it has a sequence number of 255 in the collating section. That is, wherever a "don't care" condition exists in the collating section, simply enter 255 as the sequence number in that element.

Note that once a character is specified to be a "don't care" character, it is always treated as if it did not appear for collating purposes. For example, if the hyphen is designated as a "don't care" character, then "user-defined" would be collated as "userdefined", and, consequently, "−2" would be collated as "2". Therefore, consideration should be given to the desired results before a character is designated as a "don't care" condition.

It should also be noted that the null string precedes a string containing only a "don't care" character. That is, if A$ = "" and B$ = "−", then A$ precedes B$ even if the hyphen is a "don't care" character.

ASCII Character Codes

ASCII Char.	EQUIVALENT FORMS			
	Binary	Oct	Hex	Dec
NULL	00000000	000	00	0
SOH	00000001	001	01	1
STX	00000010	002	02	2
ETX	00000011	003	03	3
EOT	00000100	004	04	4
ENQ	00000101	005	05	5
ACK	00000110	006	06	6
BELL	00000111	007	07	7
BS	00001000	010	08	8
HT	00001001	011	09	9
LF	00001010	012	0A	10
VT	00001011	013	0B	11
FF	00001100	014	0C	12
CR	00001101	015	0D	13
SO	00001110	016	0E	14
SI	00001111	017	0F	15
DLE	00010000	020	10	16
DC1	00010001	021	11	17
DC2	00010010	022	12	18
DC3	00010011	023	13	19
DC4	00010100	024	14	20
NAK	00010101	025	15	21
SYNC	00010110	026	16	22
ETB	00010111	027	17	23
CAN	00011000	030	18	24
EM	00011001	031	19	25
SUB	00011010	032	1A	26
ESC	00011011	033	1B	27
FS	00011100	034	1C	28
GS	00011101	035	1D	29
RS	00011110	036	1E	30
US	00011111	037	1F	31

ASCII Char.	EQUIVALENT FORMS			
	Binary	Oct	Hex	Dec
space	00100000	040	20	32
!	00100001	041	21	33
"	00100010	042	22	34
#	00100011	043	23	35
$	00100100	044	24	36
%	00100101	045	25	37
&	00100110	046	26	38
'	00100111	047	27	39
(	00101000	050	28	40
)	00101001	051	29	41
*	00101010	052	2A	42
+	00101011	053	2B	43
,	00101100	054	2C	44
–	00101101	055	2D	45
.	00101110	056	2E	46
/	00101111	057	2F	47
0	00110000	060	30	48
1	00110001	061	31	49
2	00110010	062	32	50
3	00110011	063	33	51
4	00110100	064	34	52
5	00110101	065	35	53
6	00110110	066	36	54
7	00110111	067	37	55
8	00111000	070	38	56
9	00111001	071	39	57
:	00111010	072	3A	58
;	00111011	073	3B	59
<	00111100	074	3C	60
=	00111101	075	3D	61
>	00111110	076	3E	62
?	00111111	077	3F	63

ASCII Char.	EQUIVALENT FORMS			
	Binary	Oct	Hex	Dec
@	01000000	100	40	64
A	01000001	101	41	65
B	01000010	102	42	66
C	01000011	103	43	67
D	01000100	104	44	68
E	01000101	105	45	69
F	01000110	106	46	70
G	01000111	107	47	71
H	01001000	110	48	72
I	01001001	111	49	73
J	01001010	112	4A	74
K	01001011	113	4B	75
L	01001100	114	4C	76
M	01001101	115	4D	77
N	01001110	116	4E	78
O	01001111	117	4F	79
P	01010000	120	50	80
Q	01010001	121	51	81
R	01010010	122	52	82
S	01010011	123	53	83
T	01010100	124	54	84
U	01010101	125	55	85
V	01010110	126	56	86
W	01010111	127	57	87
X	01011000	130	58	88
Y	01011001	131	59	89
Z	01011010	132	5A	90
[	01011011	133	5B	91
\	01011100	134	5C	92
]	01011101	135	5D	93
^	01011110	136	5E	94
_	01011111	137	5F	95

ASCII Char.	EQUIVALENT FORMS			
	Binary	Oct	Hex	Dec
`	01100000	140	60	96
a	01100001	141	61	97
b	01100010	142	62	98
c	01100011	143	63	99
d	01100100	144	64	100
e	01100101	145	65	101
f	01100110	146	66	102
g	01100111	147	67	103
h	01101000	150	68	104
i	01101001	151	69	105
j	01101010	152	6A	106
k	01101011	153	6B	107
l	01101100	154	6C	108
m	01101101	155	6D	109
n	01101110	156	6E	110
o	01101111	157	6F	111
p	01110000	160	70	112
q	01110001	161	71	113
r	01110010	162	72	114
s	01110011	163	73	115
t	01110100	164	74	116
u	01110101	165	75	117
v	01110110	166	76	118
w	01110111	167	77	119
x	01111000	170	78	120
y	01111001	171	79	121
z	01111010	172	7A	122
{	01111011	173	7B	123
|	01111100	174	7C	124
}	01111101	175	7D	125
~	01111110	176	7E	126
DEL	01111111	177	7F	127

Roman Extension Character Codes

ASCII Char.	Binary	Octal	Decimal
	EQUIVALENT FORMS		
CLEAR	10000000	200	128
IV	10000001	201	129
BL	10000010	202	130
IV/BL	10000011	203	131
UL	10000100	204	132
IV/UL	10000101	205	133
BL/UL	10000110	206	134
IV/BL/UL	10000111	207	135
	10010111	227	151
	10101000	250	168
	10101001	251	169
	10101010	252	170
	10001000	210	136
	10001001	211	137
	10001010	212	138
	10001011	213	139
	10001100	214	140
	10001101	215	141
	10001110	216	142
	10001111	217	143
	10010000	220	144
	10010001	221	145
	10010010	222	146
	10010011	223	147
	10010100	224	148
	10010101	225	149
	10010110	226	150
	10010111	227	157
	10011000	230	152
	10011001	231	153
	10011010	232	154
	10011011	233	155
	10011100	234	156

ASCII Char.	Binary	Octal	Decimal
	EQUIVALENT FORMS		
	10011101	235	157
	10011110	236	158
	10011111	237	159
	10100000	240	160
À	10100001	241	161
Î	10100010	242	162
Ò	10100011	243	163
Ù	10100100	244	164
Á	10100101	245	165
É	10100110	246	166
Ó	10100111	247	167
´	00101000	250	168
`	00101001	251	169
^	00101010	252	170
¨	10101011	253	171
~	10101100	254	172
Ê	10101101	255	173
Ô	10101110	256	174
£	10101111	257	175
—	10110000	260	176
Ā	10110001	261	177
ā	10110010	262	178
°	10110011	263	179
Ç	10110100	264	180
ç	10110101	265	181
Ñ	10110110	266	182
ñ	10110111	267	183
¡	10111000	270	184
¿	10111001	271	185
¤	10111010	272	186
£	10111011	273	187
º	10111100	274	188
§	10111101	275	189

ASCII Char.	Binary	Octal	Decimal
	EQUIVALENT FORMS		
´	10111110	276	190
+	10111111	277	191
â	11000000	300	192
ê	11000001	301	193
ô	11000010	302	194
û	11000011	303	195
á	11000100	304	196
é	11000101	305	197
ó	11000110	306	198
ú	11000111	307	199
à	11001000	310	200
è	11001001	311	201
ò	11001010	312	202
ù	11001011	313	203
ä	11001100	314	204
ë	11001101	315	205
ö	11001110	316	206
ü	11001111	317	207
Å	11010000	320	208
î	11010001	321	209
Ø	11010010	322	210
Æ	11010011	323	211
å	11010100	324	212
í	11010101	325	213
ø	11010110	326	214
æ	11010111	327	215
Ä	11011000	330	216
ì	11011001	331	217
Ö	11011010	332	218
Ü	11011011	333	219
É	11011100	334	220
ï	11011101	335	221
ß	11011110	336	222

ASCII Char.	Binary	Octal	Decimal
	EQUIVALENT FORMS		
	11011111	337	223
	11100000	340	224
	11100001	341	225
	11100010	342	226
	11100011	343	227
	11100100	344	228
	11100101	345	229
	11100110	346	230
	11100111	347	231
	11101000	350	232
	11101001	351	233
	11101010	352	234
	11101011	353	235
	11101100	354	236
	11101101	355	237
	11101110	356	238
	11101111	357	239
	11110000	360	240
	11110001	361	241
	11110010	362	242
	11110011	363	243
	11110100	364	244
	11110101	365	245
	11110110	366	246
	11110111	367	247
	11111000	370	248
	11111001	371	249
	11111010	372	250
	11111011	373	251
	11111100	374	252
	11111101	375	253
	11111110	376	254
	11111111	377	255

IV - Inverse video
BL - Blinking
UL - Underline

Advanced Programming Lexical Tables

The following listings are the Advanced Programming lexical tables. The tables are recorded on the Lexical Tables Cartridge under the names ASCII, FRENCH, GERMAN, SWEDSH and SPANSH. For your convenience, backup tables are included on the cartridge under the names BKUPAS, BKUPFR, BKUPSW and BKUPSP.

ASCII TABLE

ENTRY #1 -- LENGTH OF COLLATING TABLE = 354
ENTRY #2 -- LENGTH OF MODE TABLE = 0

------------- COLLATING SECTION -------------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR	ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
3	1	0	NUL	0	0	51	49	48	0	48	0
4	2	1	SOH	1	0	52	50	49	1	49	0
5	3	2	STX	2	0	53	51	50	2	50	0
6	4	3	ETX	3	0	54	52	51	3	51	0
7	5	4	EOT	4	0	55	53	52	4	52	0
8	6	5	ENQ	5	0	56	54	53	5	53	0
9	7	6	ACK	6	0	57	55	54	6	54	0
10	8	7	BEL	7	0	58	56	55	7	55	0
11	9	8	BS	8	0	59	57	56	8	56	0
12	10	9	HT	9	0	60	58	57	9	57	0
13	11	10	LF	10	0	61	59	58	:	58	0
14	12	11	VT	11	0	62	60	59	;	59	0
15	13	12	FF	12	0	63	61	60	<	60	0
16	14	13	CR	13	0	64	62	61	=	61	0
17	15	14	SO	14	0	65	63	62	>	62	0
18	16	15	SI	15	0	66	64	63	?	63	0
19	17	16	DLE	16	0	67	65	64	@	64	0
20	18	17	DC1	17	0	68	66	65	A	65	0
21	19	18	DC2	18	0	69	67	66	B	66	0
22	20	19	DC3	19	0	70	68	67	C	67	0
23	21	20	DC4	20	0	71	69	68	D	68	0
24	22	21	NAK	21	0	72	70	69	E	69	0
25	23	22	SYN	22	0	73	71	70	F	70	0
26	24	23	ETB	23	0	74	72	71	G	71	0
27	25	24	CAN	24	0	75	73	72	H	72	0
28	26	25	EM	25	0	76	74	73	I	73	0
29	27	26	SUB	26	0	77	75	74	J	74	0
30	28	27	ESC	27	0	78	76	75	K	75	0
31	29	28	FS	28	0	79	77	76	L	76	0
32	30	29	GS	29	0	80	78	77	M	77	0
33	31	30	RS	30	0	81	79	78	N	78	0
34	32	31	US	31	0	82	80	79	O	79	0
35	33	32	SP	32	0	83	81	80	P	80	0
36	34	33	!	33	0	84	82	81	Q	81	0
37	35	34	"	34	0	85	83	82	R	82	0
38	36	35	#	35	0	86	84	83	S	83	0
39	37	36	$	36	0	87	85	84	T	84	0
40	38	37	%	37	0	88	86	85	U	85	0
41	39	38	&	38	0	89	87	86	V	86	0
42	40	39	'	39	0	90	88	87	W	87	0
43	41	40	(	40	0	91	89	88	X	88	0
44	42	41	)	41	0	92	90	89	Y	89	0
45	43	42	*	42	0	93	91	90	Z	90	0
46	44	43	+	43	0	94	92	91	[	91	0
47	45	44	,	44	0	95	93	92	\	92	0
48	46	45	-	45	0	96	94	93	]	93	0
49	47	46	.	46	0	97	95	94	^	94	0
50	48	47	/	47	0	98	96	95	_	95	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR	ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
99	97	96	`	96	0	147	145	144		144	0
100	98	97	a	97	0	148	146	145		145	0
101	99	98	b	98	0	149	147	146		146	0
102	100	99	c	99	0	150	148	147		147	0
103	101	100	d	100	0	151	149	148		148	0
104	102	101	e	101	0	152	150	149		149	0
105	103	102	f	102	0	153	151	150		150	0
106	104	103	g	103	0	154	152	151		151	0
107	105	104	h	104	0	155	153	152		152	0
108	106	105	i	105	0	156	154	153		153	0
109	107	106	j	106	0	157	155	154		154	0
110	108	107	k	107	0	158	156	155		155	0
111	109	108	l	108	0	159	157	156		156	0
112	110	109	m	109	0	160	158	157		157	0
113	111	110	n	110	0	161	159	158		158	0
114	112	111	o	111	0	162	160	159		159	0
115	113	112	p	112	0	163	161	160		160	0
116	114	113	q	113	0	164	162	161	Á	161	0
117	115	114	r	114	0	165	163	162	Í	162	0
118	116	115	s	115	0	166	164	163	Ó	163	0
119	117	116	t	116	0	167	165	164	Ú	164	0
120	118	117	u	117	0	168	166	165	Á	165	0
121	119	118	v	118	0	169	167	166	É	166	0
122	120	119	w	119	0	170	168	167	Ó	167	0
123	121	120	x	120	0	171	169	168	´	168	0
124	122	121	y	121	0	172	170	169	`	169	0
125	123	122	z	122	0	173	171	170	^	170	0
126	124	123	{	123	0	174	172	171	¨	171	0
127	125	124	|	124	0	175	173	172	˜	172	0
128	126	125	}	125	0	176	174	173	É	173	0
129	127	126	~	126	0	177	175	174	Ö	174	0
130	128	127	DEL	127	0	178	176	175	₤	175	0
131	129	128	CLR	128	0	179	177	176	¯	176	0
132	130	129	IV	129	0	180	178	177	Ā	177	0
133	131	130	BL	130	0	181	179	178	ā	178	0
134	132	131	IV-B	131	0	182	180	179	°	179	0
135	133	132	UL	132	0	183	181	180	Ç	180	0
136	134	133	IV-U	133	0	184	182	181	ç	181	0
137	135	134	BL-U	134	0	185	183	182	Ñ	182	0
138	136	135	I-B-U	135	0	186	184	183	ñ	183	0
139	137	136		136	0	187	185	184	¡	184	0
140	138	137		137	0	188	186	185	¿	185	0
141	139	138		138	0	189	187	186	¤	186	0
142	140	139		139	0	190	188	187	£	187	0
143	141	140		140	0	191	189	188	º	188	0
144	142	141		141	0	192	190	189	§	189	0
145	143	142		142	0	193	191	190	²	190	0
146	144	143		143	0	194	192	191	•	191	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
195	193	192	â	192	0
196	194	193	ê	193	0
197	195	194	ô	194	0
198	196	195	û	195	0
199	197	196	á	196	0
200	198	197	é	197	0
201	199	198	ó	198	0
202	200	199	ú	199	0
203	201	200	à	200	0
204	202	201	è	201	0
205	203	202	ò	202	0
206	204	203	ù	203	0
207	205	204	ä	204	0
208	206	205	ë	205	0
209	207	206	ö	206	0
210	208	207	ü	207	0
211	209	208	Å	208	0
212	210	209	î	209	0
213	211	210	Ø	210	0
214	212	211	Æ	211	0
215	213	212	å	212	0
216	214	213	í	213	0
217	215	214	ø	214	0
218	216	215	æ	215	0
219	217	216	Ä	216	0
220	218	217	ì	217	0
221	219	218	Ö	218	0
222	220	219	Ü	219	0
223	221	220	É	220	0
224	222	221	ï	221	0
225	223	222	ß	222	0
226	224	223		223	0
227	225	224		224	0
228	226	225		225	0
229	227	226		226	0
230	228	227		227	0
231	229	228		228	0
232	230	229		229	0
233	231	230		230	0
234	232	231		231	0
235	233	232		232	0
236	234	233		233	0
237	235	234		234	0
238	236	235		235	0
239	237	236		236	0
240	238	237		237	0
241	239	238		238	0
242	240	239		239	0

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
243	241	240		240	0
244	242	241		241	0
245	243	242		242	0
246	244	243		243	0
247	245	244		244	0
248	246	245		245	0
249	247	246		246	0
250	248	247		247	0
251	249	248		248	0
252	250	249		249	0
253	251	250		250	0
254	252	251		251	0
255	253	252		252	0
256	254	253		253	0
257	255	254		254	0
258	256	255		255	0

---------- UPPERCASE/LOWERCASE SECTION ----------

ENTRY #	U/L ENTRY	CHAR CODE	CHAR	UPPER CASE	LOWER CASE	ENTRY #	U/L ENTRY	CHAR CODE	CHAR	UPPER CASE	LOWER CASE
259	1	160				307	49	208	Å	Å	å
260	2	161	À	À	à	308	50	209	î	I	î
261	3	162	Î	Î	î	309	51	210	Ø	Ø	ø
262	4	163	Ô	Ô	ó	310	52	211	Æ	Æ	æ
263	5	164	Ú	Ú	ú	311	53	212	å	Å	å
264	6	165	À	À	à	312	54	213	í	Í	í
265	7	166	È	È	è	313	55	214	ø	Ø	ø
266	8	167	Ô	Ô	o	314	56	215	æ	Æ	æ
267	9	168	´	´	´	315	57	216	Ä	Ä	ä
268	10	169	`	`	`	316	58	217	ì	I	ì
269	11	170	^	^	^	317	59	218	Ö	Ö	ö
270	12	171	¨	¨	¨	318	60	219	Ü	Ü	ü
271	13	172	~	~	~	319	61	220	É	É	é
272	14	173	Ê	Ê	ê	320	62	221	ï	I	ï
273	15	174	Ô	Ô	ô	321	63	222	ß	ß	ß
274	16	175	£	£	£	322	64	223			
275	17	176	¯	¯	¯	323	65	224			
276	18	177	Ā	Ā	ā	324	66	225			
277	19	178	ā	Ā	ā	325	67	226			
278	20	179	°	°	°	326	68	227			
279	21	180	Ç	Ç	ç	327	69	228			
280	22	181	ç	C	ç	328	70	229			
281	23	182	Ñ	Ñ	ñ	329	71	230			
282	24	183	ñ	Ñ	ñ	330	72	231			
283	25	184	¡	¡	¡	331	73	232			
284	26	185	¿	¿	¿	332	74	233			
285	27	186	¤	¤	¤	333	75	234			
286	28	187	£	£	£	334	76	235			
287	29	188	º	º	º	335	77	236			
288	30	189	§	§	§	336	78	237			
289	31	190	²	²	²	337	79	238			
290	32	191	·	·	·	338	80	239			
291	33	192	à	A	à	339	81	240			
292	34	193	ê	Ê	ê	340	82	241			
293	35	194	ô	Ô	ô	341	83	242			
294	36	195	û	U	û	342	84	243			
295	37	196	á	Á	á	343	85	244			
296	38	197	é	É	é	344	86	245			
297	39	198	ó	Ó	ó	345	87	246			
298	40	199	ú	Ú	ú	346	88	247			
299	41	200	à	À	à	347	89	248			
300	42	201	è	È	è	348	90	249			
301	43	202	ò	Ò	ò	349	91	250			
302	44	203	ù	U	ù	350	92	251			
303	45	204	ä	Ä	ä	351	93	252			
304	46	205	ë	E	ë	352	94	253			
305	47	206	ö	Ö	ö	353	95	254			
306	48	207	ü	Ü	ü	354	96	255			

Computer Museum

------------- MODE SECTION -------------

ENTRY #	MODE ENTRY	TYPE DESCRIPTION

|||||||||| TYPE DETAILS ||||||||||

-- DON'T CARES --

0 ASCII - 255 = Don't care

-- ACCENT PRIORITIES --

-- TWO FOR ONE REPLACEMENTS --

-- ONE FOR TWO REPLACEMENTS --

FRENCH TABLE

ENTRY #1 -- LENGTH OF COLLATING TABLE = 374
ENTRY #2 -- LENGTH OF MODE TABLE = 20

------------- COLLATING SECTION -------------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
3	1	0	NUL	0	0
4	2	1	SOH	1	0
5	3	2	STX	2	0
6	4	3	ETX	3	0
7	5	4	EOT	4	0
8	6	5	ENQ	5	0
9	7	6	ACK	6	0
10	8	7	BEL	7	0
11	9	8	BS	8	0
12	10	9	HT	9	0
13	11	10	LF	10	0
14	12	11	VT	11	0
15	13	12	FF	12	0
16	14	13	CR	13	0
17	15	14	SO	14	0
18	16	15	SI	15	0
19	17	16	DLE	16	0
20	18	17	DC1	17	0
21	19	18	DC2	18	0
22	20	19	DC3	19	0
23	21	20	DC4	20	0
24	22	21	NAK	21	0
25	23	22	SYN	22	0
26	24	23	ETB	23	0
27	25	24	CAN	24	0
28	26	25	EM	25	0
29	27	26	SUB	26	0
30	28	27	ESC	27	0
31	29	28	FS	28	0
32	30	29	GS	29	0
33	31	30	RS	30	0
34	32	31	US	31	0
35	33	32	SP	32	0
36	34	33	!	33	0
37	35	34	"	34	0
38	36	35	#	35	0
39	37	36	$	36	0
40	38	37	%	37	0
41	39	38	&	38	0
42	40	39	'	39	0
43	41	40	(	40	0
44	42	41	)	41	0
45	43	42	*	42	0
46	44	43	+	43	0
47	45	44	,	44	0
48	46	45	-	255	0
49	47	46	.	46	0
50	48	47	/	47	0
51	49	48	0	48	0
52	50	49	1	49	0
53	51	50	2	50	0
54	52	51	3	51	0
55	53	52	4	52	0
56	54	53	5	53	0
57	55	54	6	54	0
58	56	55	7	55	0
59	57	56	8	56	0
60	58	57	9	57	0
61	59	58	:	58	0
62	60	59	;	59	0
63	61	60	<	60	0
64	62	61	=	61	0
65	63	62	>	62	0
66	64	63	?	63	0
67	65	64	@	64	0
68	66	65	A	65	0
69	67	66	B	68	0
70	68	67	C	69	0
71	69	68	D	70	0
72	70	69	E	71	0
73	71	70	F	72	0
74	72	71	G	73	0
75	73	72	H	74	0
76	74	73	I	75	0
77	75	74	J	76	0
78	76	75	K	77	0
79	77	76	L	78	0
80	78	77	M	79	0
81	79	78	N	80	0
82	80	79	O	82	0
83	81	80	P	84	0
84	82	81	Q	85	0
85	83	82	R	86	0
86	84	83	S	87	0
87	85	84	T	88	0
88	86	85	U	89	0
89	87	86	V	91	0
90	88	87	W	92	0
91	89	88	X	93	0
92	90	89	Y	94	0
93	91	90	Z	95	0
94	92	91	[	96	0
95	93	92	\	97	0
96	94	93	]	98	0
97	95	94	^	99	0
98	96	95	_	100	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
99	97	96	`	101	0
100	98	97	a	102	0
101	99	98	b	103	0
102	100	99	c	104	0
103	101	100	d	106	0
104	102	101	e	107	0
105	103	102	f	108	0
106	104	103	g	109	0
107	105	104	h	110	0
108	106	105	i	111	0
109	107	106	j	112	0
110	108	107	k	113	0
111	109	108	l	114	0
112	110	109	m	115	0
113	111	110	n	116	0
114	112	111	o	118	0
115	113	112	p	119	0
116	114	113	q	120	0
117	115	114	r	121	0
118	116	115	s	122	0
119	117	116	t	123	0
120	118	117	u	124	0
121	119	118	v	125	0
122	120	119	w	126	0
123	121	120	x	127	0
124	122	121	y	128	0
125	123	122	z	129	0
126	124	123	{	133	0
127	125	124	|	134	0
128	126	125	}	135	0
129	127	126	~	136	0
130	128	127	DEL	0	0
131	129	128	CLR	0	0
132	130	129	IV	0	0
133	131	130	BL	0	0
134	132	131	IV-B	0	0
135	133	132	UL	0	0
136	134	133	IV-U	0	0
137	135	134	BL-U	0	0
138	136	135	I-B-U	0	0
139	137	136		0	0
140	138	137		0	0
141	139	138		0	0
142	140	139		0	0
143	141	140		0	0
144	142	141		0	0
145	143	142		0	0
146	144	143		0	0

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
147	145	144		0	0
148	146	145		0	0
149	147	146		0	0
150	148	147		0	0
151	149	148		0	0
152	150	149		0	0
153	151	150		0	0
154	152	151		0	0
155	153	152		0	0
156	154	153		0	0
157	155	154		0	0
158	156	155		0	0
159	157	156		0	0
160	158	157		0	0
161	159	158		0	0
162	160	159		0	0
163	161	160		0	0
164	162	161	Á	65	0
165	163	162	Í	75	0
166	164	163	Ó	82	0
167	165	164	Ú	89	0
168	166	165	Á	65	0
169	167	166	É	71	0
170	168	167	Ó	82	0
171	169	168	´	137	0
172	170	169	`	138	0
173	171	170	^	139	0
174	172	171	¨	140	0
175	173	172	˜	141	0
176	174	173	É	71	0
177	175	174	Ó	82	0
178	176	175	£	142	0
179	177	176	¯	143	0
180	178	177	Ā	65	0
181	179	178	ā	102	0
182	180	179	°	144	0
183	181	180	Ç	69	0
184	182	181	ç	105	0
185	183	182	Ñ	81	0
186	184	183	ñ	117	0
187	185	184	¡	145	0
188	186	185	¿	146	0
189	187	186	¤	147	0
190	188	187	£	148	0
191	189	188	º	149	0
192	190	189	§	150	0
193	191	190	²	151	0
194	192	191	·	152	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR	ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
195	193	192	â	102	0	243	241	240		0	0
196	194	193	ê	107	0	244	242	241		0	0
197	195	194	ô	118	0	245	243	242		0	0
198	196	195	û	124	0	246	244	243		0	0
199	197	196	á	102	0	247	245	244		0	0
200	198	197	é	107	0	248	246	245		0	0
201	199	198	ó	118	0	249	247	246		0	0
202	200	199	ú	124	0	250	248	247		0	0
203	201	200	à	102	0	251	249	248		0	0
204	202	201	è	107	0	252	250	249		0	0
205	203	202	ò	118	0	253	251	250		0	0
206	204	203	ù	124	0	254	252	251		0	0
207	205	204	ä	102	0	255	253	252		0	0
208	206	205	ë	107	0	256	254	253		0	0
209	207	206	ö	118	0	257	255	254		0	0
210	208	207	ü	124	0	258	256	255		0	0
211	209	208	Å	65	0						
212	210	209	î	111	0						
213	211	210	Ø	82	0						
214	212	211	Æ	65	0						
215	213	212	å	102	0						
216	214	213	í	111	0						
217	215	214	ø	118	0						
218	216	215	æ	102	0						
219	217	216	Ä	65	0						
220	218	217	ì	111	0						
221	219	218	Ö	82	0						
222	220	219	Ü	89	0						
223	221	220	É	71	0						
224	222	221	ï	111	0						
225	223	222	ß	122	1						
226	224	223		0	0						
227	225	224		0	0						
228	226	225		0	0						
229	227	226		0	0						
230	228	227		0	0						
231	229	228		0	0						
232	230	229		0	0						
233	231	230		0	0						
234	232	231		0	0						
235	233	232		0	0						
236	234	233		0	0						
237	235	234		0	0						
238	236	235		0	0						
239	237	236		0	0						
240	238	237		0	0						
241	239	238		0	0						
242	240	239		0	0						

---------- UPPERCASE/LOWERCASE SECTION ----------

ENTRY #	U/L ENTRY	CHAR CODE	CHAR	UPPER CASE	LOWER CASE
259	1	160			
260	2	161	Á	Á	á
261	3	162	Í	Í	í
262	4	163	Ó	Ó	ó
263	5	164	Ú	Ú	ú
264	6	165	À	À	à
265	7	166	É	É	é
266	8	167	Ò	Ò	o
267	9	168	´	´	´
268	10	169	`	`	`
269	11	170	^	^	^
270	12	171	¨	¨	¨
271	13	172	~	~	~
272	14	173	Ê	Ê	ê
273	15	174	Ô	Ô	ô
274	16	175	£	£	£
275	17	176			
276	18	177	Ā	Ā	ā
277	19	178	ā	Ā	ā
278	20	179	°	°	°
279	21	180	Ç	Ç	ç
280	22	181	ç	C	ç
281	23	182	Ñ	Ñ	ñ
282	24	183	ñ	Ñ	ñ
283	25	184	¡	¡	¡
284	26	185	¿	¿	¿
285	27	186	¤	¤	¤
286	28	187	£	£	£
287	29	188	º	º	º
288	30	189	§	§	§
289	31	190	²	²	²
290	32	191	•	•	•
291	33	192	â	A	â
292	34	193	ê	E	ê
293	35	194	ô	O	ô
294	36	195	û	U	û
295	37	196	á	A	á
296	38	197	é	E	é
297	39	198	ó	O	ó
298	40	199	ú	U	ú
299	41	200	à	A	à
300	42	201	è	E	è
301	43	202	ò	O	ò
302	44	203	ù	U	ù
303	45	204	ä	A	ä
304	46	205	ë	E	ë
305	47	206	ö	O	ö
306	48	207	ü	U	ü
307	49	208	Å	Å	å
308	50	209	î	I	î
309	51	210	Ø	Ø	ø
310	52	211	Æ	Æ	æ
311	53	212	å	A	å
312	54	213	í	I	í
313	55	214	ø	Ø	ø
314	56	215	æ	Æ	æ
315	57	216	Ä	Ä	ä
316	58	217	ì	I	ì
317	59	218	Ö	Ö	ö
318	60	219	Ü	Ü	ü
319	61	220	É	É	é
320	62	221	ï	I	ï
321	63	222	ß	ß	ß
322	64	223			
323	65	224			
324	66	225			
325	67	226			
326	68	227			
327	69	228			
328	70	229			
329	71	230			
330	72	231			
331	73	232			
332	74	233			
333	75	234			
334	76	235			
335	77	236			
336	78	237			
337	79	238			
338	80	239			
339	81	240			
340	82	241			
341	83	242			
342	84	243			
343	85	244			
344	86	245			
345	87	246			
346	88	247			
347	89	248			
348	90	249			
349	91	250			
350	92	251			
351	93	252			
352	94	253			
353	95	254			
354	96	255			

------------- MODE SECTION -------------

ENTRY #	MODE ENTRY	TYPE DESCRIPTION
355	1	TWO FOR ONE CHARACTER REPLACEMENT
356	2	TWO FOR ONE CHARACTER REPLACEMENT

|||||||||| TYPE DETAILS ||||||||||

-- DON'T CARES --

0 - ASCII - 45 = Don't care

-- ACCENT PRIORITIES --

-- TWO FOR ONE REPLACEMENTS --

ENTRY #	REPLACEMENT	ASCII
1	ß = ss = ss	222

-- ONE FOR TWO REPLACEMENTS --

GERMAN TABLE

ENTRY #1 -- LENGTH OF COLLATING TABLE = 374
ENTRY #2 -- LENGTH OF MODE TABLE = 20

------------- COLLATING SECTION -------------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR	ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
3	1	0	NUL	0	0	51	49	48	0	48	0
4	2	1	SOH	1	0	52	50	49	1	49	0
5	3	2	STX	2	0	53	51	50	2	50	0
6	4	3	ETX	3	0	54	52	51	3	51	0
7	5	4	EOT	4	0	55	53	52	4	52	0
8	6	5	ENQ	5	0	56	54	53	5	53	0
9	7	6	ACK	6	0	57	55	54	6	54	0
10	8	7	BEL	7	0	58	56	55	7	55	0
11	9	8	BS	8	0	59	57	56	8	56	0
12	10	9	HT	9	0	60	58	57	9	57	0
13	11	10	LF	10	0	61	59	58	:	58	0
14	12	11	VT	11	0	62	60	59	;	59	0
15	13	12	FF	12	0	63	61	60	<	60	0
16	14	13	CR	13	0	64	62	61	=	61	0
17	15	14	SO	14	0	65	63	62	>	62	0
18	16	15	SI	15	0	66	64	63	?	63	0
19	17	16	DLE	16	0	67	65	64	@	64	0
20	18	17	DC1	17	0	68	66	65	A	65	0
21	19	18	DC2	18	0	69	67	66	B	71	0
22	20	19	DC3	19	0	70	68	67	C	72	0
23	21	20	DC4	20	0	71	69	68	D	74	0
24	22	21	NAK	21	0	72	70	69	E	75	0
25	23	22	SYN	22	0	73	71	70	F	79	0
26	24	23	ETB	23	0	74	72	71	G	80	0
27	25	24	CAN	24	0	75	73	72	H	81	0
28	26	25	EM	25	0	76	74	73	I	82	0
29	27	26	SUB	26	0	77	75	74	J	84	0
30	28	27	ESC	27	0	78	76	75	K	85	0
31	29	28	FS	28	0	79	77	76	L	86	0
32	30	29	GS	29	0	80	78	77	M	87	0
33	31	30	RS	30	0	81	79	78	N	88	0
34	32	31	US	31	0	82	80	79	O	90	0
35	33	32	SP	32	0	83	81	80	P	95	0
36	34	33	!	33	0	84	82	81	Q	96	0
37	35	34	"	34	0	85	83	82	R	97	0
38	36	35	#	35	0	86	84	83	S	98	0
39	37	36	$	36	0	87	85	84	T	99	0
40	38	37	%	37	0	88	86	85	U	100	0
41	39	38	&	38	0	89	87	86	V	102	0
42	40	39	'	39	0	90	88	87	W	103	0
43	41	40	(	40	0	91	89	88	X	104	0
44	42	41	)	41	0	92	90	89	Y	105	0
45	43	42	*	42	0	93	91	90	Z	106	0
46	44	43	+	43	0	94	92	91	[	107	0
47	45	44	,	44	0	95	93	92	\	108	0
48	46	45	-	45	0	96	94	93	]	109	0
49	47	46	.	46	0	97	95	94	^	110	0
50	48	47	/	47	0	98	96	95	_	111	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR	ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
99	97	96	`	112	0	147	145	144		0	0
100	98	97	a	113	0	148	146	145		0	0
101	99	98	b	120	0	149	147	146		0	0
102	100	99	c	121	0	150	148	147		0	0
103	101	100	d	123	0	151	149	148		0	0
104	102	101	e	124	0	152	150	149		0	0
105	103	102	f	129	0	153	151	150		0	0
106	104	103	g	130	0	154	152	151		0	0
107	105	104	h	131	0	155	153	152		0	0
108	106	105	i	132	0	156	154	153		0	0
109	107	106	j	137	0	157	155	154		0	0
110	108	107	k	138	0	158	156	155		0	0
111	109	108	l	139	0	159	157	156		0	0
112	110	109	m	140	0	160	158	157		0	0
113	111	110	n	141	0	161	159	158		0	0
114	112	111	o	143	0	162	160	159		0	0
115	113	112	p	148	0	163	161	160		0	0
116	114	113	q	149	0	164	162	161	Á	68	0
117	115	114	r	150	0	165	163	162	Í	83	0
118	116	115	s	151	0	166	164	163	Ó	91	0
119	117	116	t	152	0	167	165	164	Ú	101	0
120	118	117	u	153	0	168	166	165	Á	69	0
121	119	118	v	157	0	169	167	166	É	77	0
122	120	119	w	158	0	170	168	167	Ó	92	0
123	121	120	x	159	0	171	169	168	´	166	0
124	122	121	y	160	0	172	170	169	`	167	0
125	123	122	z	161	0	173	171	170	^	168	0
126	124	123	{	162	0	174	172	171	¨	169	0
127	125	124	|	163	0	175	173	172	˜	170	0
128	126	125	}	164	0	176	174	173	Ê	78	0
129	127	126	~	165	0	177	175	174	Ô	93	0
130	128	127	DEL	0	0	178	176	175	₤	171	0
131	129	128	CLR	0	0	179	177	176	¯	172	0
132	130	129	IV	0	0	180	178	177	Ā	70	0
133	131	130	BL	0	0	181	179	178	ā	119	0
134	132	131	IV-B	0	0	182	180	179	°	173	0
135	133	132	UL	0	0	183	181	180	Ç	73	0
136	134	133	IV-U	0	0	184	182	181	ç	122	0
137	135	134	BL-U	0	0	185	183	182	Ñ	89	0
138	136	135	I-B-U	0	0	186	184	183	ñ	142	0
139	137	136		0	0	187	185	184	¡	174	0
140	138	137		0	0	188	186	185	¿	175	0
141	139	138		0	0	189	187	186	¤	176	0
142	140	139		0	0	190	188	187	£	177	0
143	141	140		0	0	191	189	188	º	178	0
144	142	141		0	0	192	190	189	§	179	0
145	143	142		0	0	193	191	190	²	180	0
146	144	143		0	0	194	192	191	·	181	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
195	193	192	â	118	0
196	194	193	ê	127	0
197	195	194	ô	146	0
198	196	195	û	156	0
199	197	196	á	116	0
200	198	197	é	125	0
201	199	198	ó	144	0
202	200	199	ú	154	0
203	201	200	à	117	0
204	202	201	è	126	0
205	203	202	ò	145	0
206	204	203	ù	155	0
207	205	204	ä	113	1
208	206	205	ë	128	0
209	207	206	ö	143	3
210	208	207	ü	153	5
211	209	208	Å	67	0
212	210	209	î	135	0
213	211	210	Ø	94	0
214	212	211	Æ	66	0
215	213	212	å	115	0
216	214	213	í	133	0
217	215	214	ø	147	0
218	216	215	æ	114	0
219	217	216	Ä	65	7
220	218	217	ì	134	0
221	219	218	Ö	90	9
222	220	219	Ü	100	11
223	221	220	É	76	0
224	222	221	ï	136	0
225	223	222	ß	151	13
226	224	223		0	0
227	225	224		0	0
228	226	225		0	0
229	227	226		0	0
230	228	227		0	0
231	229	228		0	0
232	230	229		0	0
233	231	230		0	0
234	232	231		0	0
235	233	232		0	0
236	234	233		0	0
237	235	234		0	0
238	236	235		0	0
239	237	236		0	0
240	238	237		0	0
241	239	238		0	0
242	240	239		0	0

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
243	241	240		0	0
244	242	241		0	0
245	243	242		0	0
246	244	243		0	0
247	245	244		0	0
248	246	245		0	0
249	247	246		0	0
250	248	247		0	0
251	249	248		0	0
252	250	249		0	0
253	251	250		0	0
254	252	251		0	0
255	253	252		0	0
256	254	253		0	0
257	255	254		0	0
258	256	255		0	0

---------- UPPERCASE/LOWERCASE SECTION ----------

ENTRY #	U/L ENTRY	CHAR CODE	CHAR	UPPER CASE	LOWER CASE
259	1	160			
260	2	161	Á	Á	à
261	3	162	Í	Í	í
262	4	163	Ó	Ó	ó
263	5	164	Ú	Ú	ú
264	6	165	À	À	à
265	7	166	È	È	è
266	8	167	Ò	Ò	ò
267	9	168	´	´	´
268	10	169	`	`	`
269	11	170	^	^	^
270	12	171	¨	¨	¨
271	13	172	~	~	~
272	14	173	Ê	Ê	ê
273	15	174	Ô	Ô	ô
274	16	175	£	£	£
275	17	176	¯	¯	¯
276	18	177	Ā	Ā	ā
277	19	178	ā	Ā	ā
278	20	179	°	°	°
279	21	180	Ç	Ç	ç
280	22	181	ç	Ç	ç
281	23	182	Ñ	Ñ	ñ
282	24	183	ñ	Ñ	ñ
283	25	184	¡	¡	¡
284	26	185	¿	¿	¿
285	27	186	¤	¤	¤
286	28	187	£	£	£
287	29	188	º	º	º
288	30	189	§	§	§
289	31	190	²	²	²
290	32	191	+	+	+
291	33	192	â	A	â
292	34	193	ê	É	ê
293	35	194	ô	Ô	ô
294	36	195	û	U	û
295	37	196	á	Á	á
296	38	197	é	É	é
297	39	198	ó	Ó	ó
298	40	199	ú	Ú	ú
299	41	200	à	À	à
300	42	201	è	È	è
301	43	202	ò	Ò	ò
302	44	203	ù	U	ù
303	45	204	ä	Ä	ä
304	46	205	ë	E	ë
305	47	206	ö	Ö	ö
306	48	207	ü	Ü	ü
307	49	208	Å	Å	å
308	50	209	î	I	î
309	51	210	Ø	Ø	ø
310	52	211	Æ	Æ	æ
311	53	212	å	Å	å
312	54	213	í	Í	í
313	55	214	ø	Ø	ø
314	56	215	æ	Æ	æ
315	57	216	Ä	Ä	ä
316	58	217	ì	I	ì
317	59	218	Ö	Ö	ö
318	60	219	Ü	Ü	ü
319	61	220	É	É	é
320	62	221	ï	I	ï
321	63	222	ß	ß	ß
322	64	223			
323	65	224			
324	66	225			
325	67	226			
326	68	227			
327	69	228			
328	70	229			
329	71	230			
330	72	231			
331	73	232			
332	74	233			
333	75	234			
334	76	235			
335	77	236			
336	78	237			
337	79	238			
338	80	239			
339	81	240			
340	82	241			
341	83	242			
342	84	243			
343	85	244			
344	86	245			
345	87	246			
346	88	247			
347	89	248			
348	90	249			
349	91	250			
350	92	251			
351	93	252			
352	94	253			
353	95	254			
354	96	255			

------------- MODE SECTION -------------

ENTRY #	MODE ENTRY	TYPE DESCRIPTION
355	1	TWO FOR ONE CHARACTER REPLACEMENT
356	2	TWO FOR ONE CHARACTER REPLACEMENT
357	3	TWO FOR ONE CHARACTER REPLACEMENT
358	4	TWO FOR ONE CHARACTER REPLACEMENT
359	5	TWO FOR ONE CHARACTER REPLACEMENT
360	6	TWO FOR ONE CHARACTER REPLACEMENT
361	7	TWO FOR ONE CHARACTER REPLACEMENT
362	8	TWO FOR ONE CHARACTER REPLACEMENT
363	9	TWO FOR ONE CHARACTER REPLACEMENT
364	10	TWO FOR ONE CHARACTER REPLACEMENT
365	11	TWO FOR ONE CHARACTER REPLACEMENT
366	12	TWO FOR ONE CHARACTER REPLACEMENT
367	13	TWO FOR ONE CHARACTER REPLACEMENT
368	14	TWO FOR ONE CHARACTER REPLACEMENT

|||||||||| TYPE DETAILS ||||||||||

-- DON'T CARES --

-- ACCENT PRIORITIES --

-- TWO FOR ONE REPLACEMENTS --

ENTRY #	REPLACEMENT	ASCII
1	ä = ae = ae	204
3	ö = oe = oe	206
5	ü = ue = ue	207
7	Ä = AE = Ae	216
9	Ö = OE = Oe	218
11	Ü = UE = Ue	219
13	ß = ss = ss	222

-- ONE FOR TWO REPLACEMENTS --

SPANISH TABLE

ENTRY #1 -- LENGTH OF COLLATING TABLE = 374
ENTRY #2 -- LENGTH OF MODE TABLE = 20

------------- COLLATING SECTION -------------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
3	1	0	NUL	0	0
4	2	1	SOH	1	0
5	3	2	STX	2	0
6	4	3	ETX	3	0
7	5	4	EOT	4	0
8	6	5	ENQ	5	0
9	7	6	ACK	6	0
10	8	7	BEL	7	0
11	9	8	BS	8	0
12	10	9	HT	9	0
13	11	10	LF	10	0
14	12	11	VT	11	0
15	13	12	FF	12	0
16	14	13	CR	13	0
17	15	14	SO	14	0
18	16	15	SI	15	0
19	17	16	DLE	16	0
20	18	17	DC1	17	0
21	19	18	DC2	18	0
22	20	19	DC3	19	0
23	21	20	DC4	20	0
24	22	21	NAK	21	0
25	23	22	SYN	22	0
26	24	23	ETB	23	0
27	25	24	CAN	24	0
28	26	25	EM	25	0
29	27	26	SUB	26	0
30	28	27	ESC	27	0
31	29	28	FS	28	0
32	30	29	GS	29	0
33	31	30	RS	30	0
34	32	31	US	31	0
35	33	32	SP	32	0
36	34	33	!	33	0
37	35	34	"	34	0
38	36	35	#	35	0
39	37	36	$	36	0
40	38	37	%	37	0
41	39	38	&	38	0
42	40	39	'	39	0
43	41	40	(	40	0
44	42	41	)	41	0
45	43	42	*	42	0
46	44	43	+	43	0
47	45	44	,	44	0
48	46	45	-	45	0
49	47	46	.	46	0
50	48	47	/	47	0

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
51	49	48	0	48	0
52	50	49	1	49	0
53	51	50	2	50	0
54	52	51	3	51	0
55	53	52	4	52	0
56	54	53	5	53	0
57	55	54	6	54	0
58	56	55	7	55	0
59	57	56	8	56	0
60	58	57	9	57	0
61	59	58	:	58	0
62	60	59	;	59	0
63	61	60	<	60	0
64	62	61	=	61	0
65	63	62	>	62	0
66	64	63	?	63	0
67	65	64	@	64	0
68	66	65	A	65	0
69	67	66	B	66	0
70	68	67	C	67	1
71	69	68	D	69	0
72	70	69	E	70	0
73	71	70	F	71	0
74	72	71	G	72	0
75	73	72	H	73	0
76	74	73	I	74	0
77	75	74	J	75	0
78	76	75	K	76	0
79	77	76	L	77	4
80	78	77	M	79	0
81	79	78	N	80	0
82	80	79	O	82	0
83	81	80	P	83	0
84	82	81	Q	84	0
85	83	82	R	85	0
86	84	83	S	86	0
87	85	84	T	87	0
88	86	85	U	88	0
89	87	86	V	89	0
90	88	87	W	90	0
91	89	88	X	91	0
92	90	89	Y	92	0
93	91	90	Z	93	0
94	92	91	[	98	0
95	93	92	\	99	0
96	94	93	]	100	0
97	95	94	^	101	0
98	96	95	_	102	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR	ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
99	97	96	`	103	0	147	145	144		0	0
100	98	97	a	104	0	148	146	145		0	0
101	99	98	b	105	0	149	147	146		0	0
102	100	99	c	106	7	150	148	147		0	0
103	101	100	d	108	0	151	149	148		0	0
104	102	101	e	109	0	152	150	149		0	0
105	103	102	f	110	0	153	151	150		0	0
106	104	103	g	111	0	154	152	151		0	0
107	105	104	h	112	0	155	153	152		0	0
108	106	105	i	113	0	156	154	153		0	0
109	107	106	j	114	0	157	155	154		0	0
110	108	107	k	115	0	158	156	155		0	0
111	109	108	l	116	10	159	157	156		0	0
112	110	109	m	118	0	160	158	157		0	0
113	111	110	n	119	0	161	159	158		0	0
114	112	111	o	121	0	162	160	159		0	0
115	113	112	p	122	0	163	161	160		0	0
116	114	113	q	123	0	164	162	161	Á	65	0
117	115	114	r	124	0	165	163	162	Í	74	0
118	116	115	s	125	0	166	164	163	Ó	82	0
119	117	116	t	126	0	167	165	164	Ú	88	0
120	118	117	u	127	0	168	166	165	Á	65	0
121	119	118	v	128	0	169	167	166	É	70	0
122	120	119	w	129	0	170	168	167	Ó	82	0
123	121	120	x	130	0	171	169	168	´	137	0
124	122	121	y	131	0	172	170	169	`	138	0
125	123	122	z	132	0	173	171	170	^	139	0
126	124	123	{	133	0	174	172	171	¨	140	0
127	125	124	|	134	0	175	173	172	˜	141	0
128	126	125	}	135	0	176	174	173	É	70	0
129	127	126	~	136	0	177	175	174	Ó	82	0
130	128	127	DEL	0	0	178	176	175	₤	142	0
131	129	128	CLR	0	0	179	177	176	¯	143	0
132	130	129	IV	0	0	180	178	177	Ā	65	0
133	131	130	BL	0	0	181	179	178	ª	104	0
134	132	131	IV-B	0	0	182	180	179	°	144	0
135	133	132	UL	0	0	183	181	180	Ç	67	0
136	134	133	IV-U	0	0	184	182	181	ç	106	0
137	135	134	BL-U	0	0	185	183	182	Ñ	81	0
138	136	135	I-B-U	0	0	186	184	183	ñ	120	0
139	137	136		0	0	187	185	184	¡	145	0
140	138	137		0	0	188	186	185	¿	146	0
141	139	138		0	0	189	187	186	¤	147	0
142	140	139		0	0	190	188	187	£	148	0
143	141	140		0	0	191	189	188	º	149	0
144	142	141		0	0	192	190	189	§	150	0
145	143	142		0	0	193	191	190	²	151	0
146	144	143		0	0	194	192	191	·	152	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
195	193	192	â	104	0
196	194	193	ê	109	0
197	195	194	ô	121	0
198	196	195	û	127	0
199	197	196	á	104	0
200	198	197	é	109	0
201	199	198	ó	121	0
202	200	199	ú	127	0
203	201	200	à	104	0
204	202	201	è	109	0
205	203	202	ò	121	0
206	204	203	ù	127	0
207	205	204	ä	104	0
208	206	205	ë	109	0
209	207	206	ö	121	0
210	208	207	ü	127	0
211	209	208	Å	65	0
212	210	209	î	113	0
213	211	210	Ø	82	0
214	212	211	Æ	65	0
215	213	212	å	104	0
216	214	213	í	113	0
217	215	214	ø	121	0
218	216	215	æ	104	0
219	217	216	Ä	65	0
220	218	217	ì	113	0
221	219	218	Ö	82	0
222	220	219	Ü	88	0
223	221	220	É	70	0
224	222	221	ï	113	0
225	223	222	ß	125	13
226	224	223		0	0
227	225	224		0	0
228	226	225		0	0
229	227	226		0	0
230	228	227		0	0
231	229	228		0	0
232	230	229		0	0
233	231	230		0	0
234	232	231		0	0
235	233	232		0	0
236	234	233		0	0
237	235	234		0	0
238	236	235		0	0
239	237	236		0	0
240	238	237		0	0
241	239	238		0	0
242	240	239		0	0

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
243	241	240		0	0
244	242	241		0	0
245	243	242		0	0
246	244	243		0	0
247	245	244		0	0
248	246	245		0	0
249	247	246		0	0
250	248	247		0	0
251	249	248		0	0
252	250	249		0	0
253	251	250		0	0
254	252	251		0	0
255	253	252		0	0
256	254	253		0	0
257	255	254		0	0
258	256	255		0	0

---------- UPPERCAS[illegible]ERCASE SECTION ----------

ENTRY #	U/L ENTRY	CHAR CODE	CHAR	UPPER CASE	LOWER CASE
259	1	160			
260	2	161	Á	Á	á
261	3	162	Í	Í	í
262	4	163	Ó	Ó	ó
263	5	164	Ú	Ú	ú
264	6	165	À	À	à
265	7	166	È	È	è
266	8	167	Ò	Ò	o
267	9	168	´	´	´
268	10	169	`	`	`
269	11	170	^	^	^
270	12	171	¨	¨	¨
271	13	172	˜	˜	˜
272	14	173	Ê	Ê	ê
273	15	174	Ô	Ô	ô
274	16	175	£	£	£
275	17	176	_	_	_
276	18	177	Ā	Ā	ā
277	19	178	ā	Ā	ā
278	20	179	°	°	°
279	21	180	Ç	Ç	ç
280	22	181	ç	Ç	ç
281	23	182	Ñ	Ñ	ñ
282	24	183	ñ	Ñ	ñ
283	25	184	¡	¡	¡
284	26	185	¿	¿	¿
285	27	186	¤	¤	¤
286	28	187	£	£	£
287	29	188	º	º	º
288	30	189	§	§	§
289	31	190	²	²	²
290	32	191	•	•	•
291	33	192	â	A	â
292	34	193	ê	E	ê
293	35	194	ô	O	ô
294	36	195	û	U	û
295	37	196	á	A	á
296	38	197	é	E	é
297	39	198	ó	O	ó
298	40	199	ú	U	ú
299	41	200	à	A	à
300	42	201	è	E	è
301	43	202	ò	O	ò
302	44	203	ù	U	ù
303	45	204	ä	Ä	ä
304	46	205	ë	E	ë
305	47	206	ö	Ö	ö
306	48	207	ü	Ü	ü
307	49	208	Å	Å	å
308	50	209	î	I	î
309	51	210	Ø	Ø	ø
310	52	211	Æ	Æ	æ
311	53	212	å	Å	å
312	54	213	í	I	í
313	55	214	ø	Ø	ø
314	56	215	æ	Æ	æ
315	57	216	Ä	Ä	ä
316	58	217	ì	I	ì
317	59	218	Ö	Ö	ö
318	60	219	Ü	Ü	ü
319	61	220	É	É	é
320	62	221	ï	I	ï
321	63	222	ß	ß	ß
322	64	223			
323	65	224			
324	66	225			
325	67	226			
326	68	227			
327	69	228			
328	70	229			
329	71	230			
330	72	231			
331	73	232			
332	74	233			
333	75	234			
334	76	235			
335	77	236			
336	78	237			
337	79	238			
338	80	239			
339	81	240			
340	82	241			
341	83	242			
342	84	243			
343	85	244			
344	86	245			
345	87	246			
346	88	247			
347	89	248			
348	90	249			
349	91	250			
350	92	251			
351	93	252			
352	94	253			
353	95	254			
354	96	255			

------------- MODE SECTION -------------

ENTRY #	MODE ENTRY	TYPE DESCRIPTION
355	1	ONE FOR TWO CHARACTER REPLACEMENT
356	2	ONE FOR TWO CHARACTER REPLACEMENT
357	3	* TERMINATOR WORD *
358	4	ONE FOR TWO CHARACTER REPLACEMENT
359	5	ONE FOR TWO CHARACTER REPLACEMENT
360	6	* TERMINATOR WORD *
361	7	ONE FOR TWO CHARACTER REPLACEMENT
362	8	ONE FOR TWO CHARACTER REPLACEMENT
363	9	* TERMINATOR WORD *
364	10	ONE FOR TWO CHARACTER REPLACEMENT
365	11	ONE FOR TWO CHARACTER REPLACEMENT
366	12	* TERMINATOR WORD *
367	13	TWO FOR ONE CHARACTER REPLACEMENT
368	14	TWO FOR ONE CHARACTER REPLACEMENT

|||||||||| TYPE DETAILS ||||||||||

-- DON'T CARES --

-- ACCENT PRIORITIES --

-- TWO FOR ONE REPLACEMENTS --

ENTRY #	REPLACEMENT	ASCII
13	ß = ss = ss	222

-- ONE FOR TWO REPLACEMENTS --

ENTRY #	REPLACEMENT AND SEQUENCE NUMBER
1	CH = 68
2	Ch = 68
4	LL = 78
5	Ll = 78
7	cH = 107
8	ch = 107
10	lL = 117
11	ll = 117

SWEDISH TABLE

ENTRY #1 -- LENGTH OF COLLATING TABLE = 374
ENTRY #2 -- LENGTH OF MODE TABLE = 20

------------- COLLATING SECTION -------------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
3	1	0	NUL	0	0
4	2	1	SOH	1	0
5	3	2	STX	2	0
6	4	3	ETX	3	0
7	5	4	EOT	4	0
8	6	5	ENQ	5	0
9	7	6	ACK	6	0
10	8	7	BEL	7	0
11	9	8	BS	8	0
12	10	9	HT	9	0
13	11	10	LF	10	0
14	12	11	VT	11	0
15	13	12	FF	12	0
16	14	13	CR	13	0
17	15	14	SO	14	0
18	16	15	SI	15	0
19	17	16	DLE	16	0
20	18	17	DC1	17	0
21	19	18	DC2	18	0
22	20	19	DC3	19	0
23	21	20	DC4	20	0
24	22	21	NAK	21	0
25	23	22	SYN	22	0
26	24	23	ETB	23	0
27	25	24	CAN	24	0
28	26	25	EM	25	0
29	27	26	SUB	26	0
30	28	27	ESC	27	0
31	29	28	FS	28	0
32	30	29	GS	29	0
33	31	30	RS	30	0
34	32	31	US	31	0
35	33	32	SP	32	0
36	34	33	!	33	0
37	35	34	"	34	0
38	36	35	#	35	0
39	37	36	$	36	0
40	38	37	%	37	0
41	39	38	&	38	0
42	40	39	'	39	0
43	41	40	(	40	0
44	42	41	)	41	0
45	43	42	*	42	0
46	44	43	+	43	0
47	45	44	,	44	0
48	46	45	-	45	0
49	47	46	.	46	0
50	48	47	/	47	0
51	49	48	0	48	0
52	50	49	1	49	0
53	51	50	2	50	0
54	52	51	3	51	0
55	53	52	4	52	0
56	54	53	5	53	0
57	55	54	6	54	0
58	56	55	7	55	0
59	57	56	8	56	0
60	58	57	9	57	0
61	59	58	:	58	0
62	60	59	;	59	0
63	61	60	<	60	0
64	62	61	=	61	0
65	63	62	>	62	0
66	64	63	?	63	0
67	65	64	@	64	0
68	66	65	A	65	0
69	67	66	B	66	0
70	68	67	C	67	0
71	69	68	D	68	0
72	70	69	E	69	0
73	71	70	F	70	0
74	72	71	G	71	0
75	73	72	H	72	0
76	74	73	I	73	0
77	75	74	J	74	0
78	76	75	K	75	0
79	77	76	L	76	0
80	78	77	M	77	0
81	79	78	N	78	0
82	80	79	O	80	0
83	81	80	P	81	0
84	82	81	Q	82	0
85	83	82	R	83	0
86	84	83	S	84	0
87	85	84	T	85	0
88	86	85	U	86	0
89	87	86	V	87	0
90	88	87	W	88	0
91	89	88	X	89	0
92	90	89	Y	90	0
93	91	90	Z	91	0
94	92	91	[	111	0
95	93	92	\	112	0
96	94	93	]	113	0
97	95	94	^	114	0
98	96	95	_	115	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
99	97	96		116	0
100	98	97	a	117	0
101	99	98	b	118	0
102	100	99	c	119	0
103	101	100	d	120	0
104	102	101	e	122	0
105	103	102	f	123	0
106	104	103	g	124	0
107	105	104	h	125	0
108	106	105	i	126	0
109	107	106	j	127	0
110	108	107	k	128	0
111	109	108	l	129	0
112	110	109	m	130	0
113	111	110	n	131	0
114	112	111	o	132	0
115	113	112	p	133	0
116	114	113	q	134	0
117	115	114	r	135	0
118	116	115	s	136	0
119	117	116	t	137	0
120	118	117	u	138	0
121	119	118	v	139	0
122	120	119	w	140	0
123	121	120	x	141	0
124	122	121	y	142	0
125	123	122	z	143	0
126	124	123	{	169	0
127	125	124	|	170	0
128	126	125	}	171	0
129	127	126	~	172	0
130	128	127	DEL	0	0
131	129	128	CLR	0	0
132	130	129	IV	0	0
133	131	130	BL	0	0
134	132	131	IV-B	0	0
135	133	132	UL	0	0
136	134	133	IV-U	0	0
137	135	134	BL-U	0	0
138	136	135	I-B-U	0	0
139	137	136		0	0
140	138	137		0	0
141	139	138		0	0
142	140	139		0	0
143	141	140		0	0
144	142	141		0	0
145	143	142		0	0
146	144	143		0	0

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
147	145	144		0	0
148	146	145		0	0
149	147	146		0	0
150	148	147		0	0
151	149	148		0	0
152	150	149		0	0
153	151	150		0	0
154	152	151		0	0
155	153	152		0	0
156	154	153		0	0
157	155	154		0	0
158	156	155		0	0
159	157	156		0	0
160	158	157		0	0
161	159	158		0	0
162	160	159		0	0
163	161	160		0	0
164	162	161	Á	94	0
165	163	162	Í	102	0
166	164	163	Ó	103	0
167	165	164	Ú	109	0
168	166	165	Á	95	0
169	167	166	É	100	0
170	168	167	Ó	104	0
171	169	168	´	173	0
172	170	169	`	174	0
173	171	170	^	175	0
174	172	171	¨	176	0
175	173	172	˜	177	0
176	174	173	Ê	101	0
177	175	174	Ô	105	0
178	176	175	₤	178	0
179	177	176	¯	179	0
180	178	177	Ā	97	0
181	179	178	ā	150	0
182	180	179	°	180	0
183	181	180	Ç	98	0
184	182	181	ç	151	0
185	183	182	Ñ	108	0
186	184	183	ñ	164	0
187	185	184	¡	181	0
188	186	185	¿	182	0
189	187	186	¤	183	0
190	188	187	£	184	0
191	189	188	º	185	0
192	190	189	§	186	0
193	191	190	²	187	0
194	192	191	·	188	0

---------- COLLATING SECTION - CONT ----------

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
195	193	192	â	148	0
196	194	193	ê	153	0
197	195	194	ô	161	0
198	196	195	û	167	0
199	197	196	á	146	0
200	198	197	é	122	0
201	199	198	ó	159	0
202	200	199	ú	165	0
203	201	200	à	147	0
204	202	201	è	152	0
205	203	202	ò	160	0
206	204	203	ù	166	0
207	205	204	ä	149	0
208	206	205	ë	154	0
209	207	206	ö	162	0
210	208	207	ü	168	0
211	209	208	Å	93	0
212	210	209	î	157	0
213	211	210	Ø	107	0
214	212	211	Æ	92	0
215	213	212	å	145	0
216	214	213	í	155	0
217	215	214	ø	163	0
218	216	215	æ	144	0
219	217	216	Ä	96	0
220	218	217	ì	156	0
221	219	218	Ö	106	0
222	220	219	Ü	110	0
223	221	220	É	99	0
224	222	221	ï	158	0
225	223	222	ß	136	1
226	224	223		0	0
227	225	224		0	0
228	226	225		0	0
229	227	226		0	0
230	228	227		0	0
231	229	228		0	0
232	230	229		0	0
233	231	230		0	0
234	232	231		0	0
235	233	232		0	0
236	234	233		0	0
237	235	234		0	0
238	236	235		0	0
239	237	236		0	0
240	238	237		0	0
241	239	238		0	0
242	240	239		0	0

ENTRY #	COLL. ENTRY	CHAR CODE	CHAR	SEQ #	MODE PTR
243	241	240		0	0
244	242	241		0	0
245	243	242		0	0
246	244	243		0	0
247	245	244		0	0
248	246	245		0	0
249	247	246		0	0
250	248	247		0	0
251	249	248		0	0
252	250	249		0	0
253	251	250		0	0
254	252	251		0	0
255	253	252		0	0
256	254	253		0	0
257	255	254		0	0
258	256	255		0	0

---------- UPPERCASE/LOWERCASE SECTION ----------

ENTRY #	U/L ENTRY	CHAR CODE	CHAR	UPPER CASE	LOWER CASE	ENTRY #	U/L ENTRY	CHAR CODE	CHAR	UPPER CASE	LOWER CASE
259	1	160				307	49	208	Å	Å	å
260	2	161	Á	Á	á	308	50	209	î	I	î
261	3	162	Í	Í	í	309	51	210	Ø	Ø	ø
262	4	163	Ó	Ó	ó	310	52	211	Æ	Æ	æ
263	5	164	Ú	Ú	ú	311	53	212	å	Å	å
264	6	165	À	À	à	312	54	213	í	Í	í
265	7	166	È	È	è	313	55	214	ø	Ø	ø
266	8	167	Ò	Ò	ò	314	56	215	æ	Æ	æ
267	9	168	´	´	´	315	57	216	Ä	Ä	ä
268	10	169	`	`	`	316	58	217	ì	I	ì
269	11	170	^	^	^	317	59	218	Ö	Ö	ö
270	12	171	¨	¨	¨	318	60	219	Ü	Ü	ü
271	13	172	~	~	~	319	61	220	É	É	é
272	14	173	Ê	Ê	ê	320	62	221	ï	I	ï
273	15	174	Ô	Ô	ô	321	63	222	ß	ß	ß
274	16	175	£	£	£	322	64	223			
275	17	176	¯	¯	¯	323	65	224			
276	18	177	Ā	Ā	ā	324	66	225			
277	19	178	ā	Ā	ā	325	67	226			
278	20	179	°	°	°	326	68	227			
279	21	180	Ç	Ç	ç	327	69	228			
280	22	181	ç	C	ç	328	70	229			
281	23	182	Ñ	Ñ	ñ	329	71	230			
282	24	183	ñ	Ñ	ñ	330	72	231			
283	25	184	¡	¡	¡	331	73	232			
284	26	185	¿	¿	¿	332	74	233			
285	27	186	¤	¤	¤	333	75	234			
286	28	187	£	£	£	334	76	235			
287	29	188	º	º	º	335	77	236			
288	30	189	§	§	§	336	78	237			
289	31	190	Ω	Ω	Ω	337	79	238			
290	32	191	₧	₧	₧	338	80	239			
291	33	192	â	A	â	339	81	240			
292	34	193	ê	Ê	ê	340	82	241			
293	35	194	ô	Ô	ô	341	83	242			
294	36	195	û	U	û	342	84	243			
295	37	196	á	Á	á	343	85	244			
296	38	197	é	É	é	344	86	245			
297	39	198	ó	Ó	ó	345	87	246			
298	40	199	ú	Ú	ú	346	88	247			
299	41	200	à	À	à	347	89	248			
300	42	201	è	È	è	348	90	249			
301	43	202	ò	Ò	ò	349	91	250			
302	44	203	ù	U	ù	350	92	251			
303	45	204	ä	Ä	ä	351	93	252			
304	46	205	ë	E	ë	352	94	253			
305	47	206	ö	Ö	ö	353	95	254			
306	48	207	ü	Ü	ü	354	96	255			

------------- MODE SECTION -------------

ENTRY #	MODE ENTRY	TYPE DESCRIPTION
355	1	TWO FOR ONE CHARACTER REPLACEMENT
356	2	TWO FOR ONE CHARACTER REPLACEMENT

|||||||||| TYPE DETAILS ||||||||||

-- DON'T CARES --

-- ACCENT PRIORITIES --

-- TWO FOR ONE REPLACEMENTS --

ENTRY #	REPLACEMENT	ASCII
1	ß = ss = ss	222

-- ONE FOR TWO REPLACEMENTS --

Error Messages

Mainframe Errors

1	Missing ROM or configuration error. Also, check to see if all option ROMs are installed properly.
2	Memory overflow; subprogram larger than block of memory. Also check to see if your arrays are too large to fit in memory.
3	Line not found or not in current program segment. Check the spelling of line labels and line identifiers.
4	Improper return. Branched into the middle of a subroutine.
5	Abnormal program termination; no END or STOP statement.
6	Improper FOR/NEXT matching.
7	Undefined function or subroutine. Check spellings.
8	Improper parameter matching. Check the parameter lists in SUB and CALL, and DEF FN and FN statements to see if they match in number and type.
9	Improper number of parameters. Check the number of arguments used in an FN or CALL reference.
10	String value required.
11	Numeric value required.
12	Attempt to redeclare variable. Once a variable name has been declared in a DIM, COM, REAL, SHORT or INTEGER statement, it can't be redeclared in that program segment.
13	Array dimensions not specified. You must dimension the array, either explicitly or implicitly.
14	Multiple OPTION BASE statements or OPTION BASE statement preceded by variable declarative statements.
15	Invalid bounds on array dimension or string length in DIM, COM, REAL, SHORT or INTEGER statement. Strings can't be longer than 32 767 characters. The range of array subscripts is −32 767 through 32 767.
16	Dimensions are improper or inconsistent; more than 32 767 elements in an array. Check for wrong number of subscripts in an array reference. Check any matrix multiplication for proper sizes.
17	Subscript out of range.
18	Substring out of range or string too long. Check substring specifiers against length of string.
19	Improper value. Check numbers being entered, especially their exponents.
20	Integer precision overflow. The range is −32 768 through 32 767.

Appendix C

21	Short precision overflow. Short-precision numbers have six significant digits and an exponent in the range −63 through 63.
22	Real precision overflow. Full-precision numbers have twelve significant digits and an exponent in the range −99 through 99.
23	Intermediate result overflow.
24	TAN (n*π/2), when n is odd.
25	Magnitude of argument of ASN or ACS is greater than 1.
26	Zero to negative power.
27	Negative base to non-integer power.
28	LOG or LGT of negative number.
29	LOG or LGT of zero.
30	SQR of negative number.
31	Division by zero; or X MOD Y with Y = 0.
32	String does not represent valid number or string response when numeric data required. Check any use of VAL function and its argument. Check for correct spelling of variable name.
33	Improper argument for NUM, CHR$, or RPT$ function.
34	Referenced line is not IMAGE statement. Check the line identifier in the PRINT USING statement.
35	Improper format string.
36	Out of DATA. Make sure READ and DATA statements correspond. Use RESTORE if appropriate.
37	EDIT string longer than 160 characters. Try using a substring.
38	I/O function not allowed. TYP and other I/O functions aren't allowed in any I/O statement like DISP or PRINT. Place the value into a variable.
39	Function subprogram not allowed. An FN reference isn't allowed in any I/O statement, or in redim subscripts. Place the value into a variable.
40	Improper replace, delete or REN command. SUB and DEF FN can only be replaced by another SUB or DEF FN. They can only be deleted if the rest of the corresponding subprogram is deleted. A renumbering may cause out-of-range line numbers if completed, so an error occurs; check increment value.
41	First line number greater than second.
42	Attempt to replace or delete a busy line or subprogram. Typically, this is caused by trying to delete an input statement that is still requesting values.
43	Matrix not square. The dimensions of an identity matrix or of one used to find an inverse or determinant must be the same size.
44	Illegal operand in matrix transpose or matrix multiply. The result matrix can't be one of the operands.
45	Nested keyboard entry statements.
46	No binary in memory for STORE BIN or no program in memory for SAVE. Check line numbers in SAVE against program in memory.
47	Subprogram COM declaration is not consistent with main program. Check number, type and dimensions of variables.

48	Recursion in single-line DEF FN function. Only subprograms can be called recursively.
49	Line specified in ON declaration not found.
50	File number less than 1 or greater than 10.
51	File not currently assigned. Execute an ASSIGN statement for the file, or check the accuracy of the file number used.
52	Improper mass storage unit specifier. Check the values of the select code, unit code and controller address.
53	Improper file name. A file name can have 1-6 characters and can't contain a colon, quote mark, NULL or CHR$(255).
54	Duplicate file name. Choose another name or PURGE the old one.
55	Directory overflow. There is a maximum number of files that a mass storage medium can hold. A file will have to be removed to add another.
56	File name is undefined. Check the spelling.
57	Mass Storage ROM is missing. Check to see that the ROM is installed properly.
58	Improper file type. Use LOAD for PROG files, ASSIGN and GET on DATA files and LOADKEY for KEYS files.
59	Physical or logical end-of-file found. Attempting to READ# or PRINT# past the end of the file. Compare the data list to the file size.
60	Physical or logical end-of-record found in random mode. Compare the data list to the record size.
61	Defined record size is too small for data item. You can either PURGE and RE-CREATE the file with longer records or regroup the data being recorded.
62	File is protected or wrong protect code specified. Check to see that the protect code is included and spelled properly.
63	The number of physical records is greater than 32 767. That's the limit; use something smaller.
64	Medium overflow (out of user storage space). A file can't be set up because there isn't enough space. Use another medium or purge unwanted files.
65	Incorrect data type. You can't use GET on a DATA file that doesn't contain a program. Use TYP to find out what kind of data the computer is trying to be read.
66	Excessive rejected tracks during a mass storage initialization. The medium can't be initialized. If the medium is a flexible disk, use a different one. If the medium is a hard disc, call your HP Sales and Service Office for assistance, to determine whether there has been a hardware failure.
67	Mass storage parameter less than or equal to 0. Check values of variables. Record numbers, record lengths and number of defined records must be positive numbers.
68	Invalid line number in GET or LINK operation. Check line numbers. May be trying to LINK to file that doesn't contain a program.
69 – 79	See Mass Storage ROM errors.
80	Cartridge out or door open. Also check to see if interface is connected properly.

81	Mass storage device failure. Possible power failure.
82	Mass storage device not present. Check mass storage unit specifier.
83	Write protected. Check the write-protection device on the medium or drive.
84	Record not found. There is a bad spot on the medium.
85	Mass storage medium is not initialized.
86	Not a compatible tape cartridge.
87	Record address error; information can't be read. Hardware failure. Check for a dirty read head.
88	Read data error. Hardware failure. Check for a dirty read head.
89	Check read error.
90	Mass storage system error.
91–99	See Mass Storage ROM errors.
100	Item in print using list is string but image specifier is numeric.
101	Item in print using list is numeric but image specifier is string.
102	Numeric field specifier wider than printer width.
103	Item in print using list has no corresponding image specifier.
104	ON KBD statement not allowed in a subprogram.
105–109	Unused
110–113	See Plotter ROM errors.

System Error octal number; octal number

This error indicates a malfunction in the machine's firmware system. Contact your Sales and Service Office.

I/O Device Errors

Two error messages can occur when attempting to direct an operation to an I/O device that is not ready for use. A printer which is out of paper or no device at a specifed select code are examples. The first message that appears is –

I/O ERROR ON SELECT CODE select code

If the condition is not corrected, the machine beeps intermittently and the following message replaces the first –

I/O TIMEOUT ON SELECT CODE select code

The I/O device can be made usable by correcting the error (loading paper, or changing the select code, for example), then executing the READY# command –

READY# select code

This command readies the I/O device and the operation which was attempted is attempted again. The select code must be specified by an integer.

If you get an I/O error on select code 0 and the printer is not out of paper, call your Sales and Service Office.

In some cases, such as an interface which is not connected, READY# for that select code may not solve the I/O error. In this case, (STOP) should be pressed to regain control of the computer. Be sure to turn the power off before inserting an interface. After the problem is remedied, the operation or program can be tried again.

If you get an I/O error and you have an ON KBD statement in effect, you must press (STOP) to gain control of the computer. Otherwise, the READY# command will be trapped by ON KBD.

Mass Storage ROM Errors

69	Format switch on the disc off. Turn it on.
70	Not a disc interface. Check mass storage unit specifier.
71	Disc interface power off. Turn it on.
72	Incorrect controller address, controller power off, or disc time out. Check mass storage unit specifier; make sure controller is on.
73	Incorrect device type in mass storage unit specifer.
74	Drive missing or power off.
75	Disc system error, type I[1].
76	Incorrect unit code in mass storage unit specifier.
77	Disc system error, type II[1].
78-79	Unused
91-99	Unused

Graphics ROM Errors

110	Plotter type specification not recognized. Check spelling of "GRAPHICS", "9872A" or "INCREMENTAL".
111	Plotter has not been specified. Check select codes.
112	No graphics hardware installed in the System 45B.
113	LIMIT specifications out of range.

1 See the Mass Storage Techniques Manual.

I/O ROM Errors

114	98036 card improperly configured.
115	TDISP not allowed unless peripheral keyboard active.
116	TOPEN is active on another select code.
150	Improper select code.
151	A negative select code was specified that does not match present bus addressing.
152	Parity error.
153	Either insufficient input data to satisfy enter list or attempt to ENTER from source into source, or enter count exhausted without linefeed.
154	Integer overflow, or ENTER count greater than 32 767 bytes or 16 383 words.
155	Invalid interface register number. (Can only specify 4-7.)
156	Improper expression type in READIO, WRITEIO, or STATUS list.
157	No linefeed was found to satify "/" ENTER image specifier, or no linefeed record delimiter was found in 512 characters of input.
158	Improper image specifier or nesting image specifiers more than 4 levels deep.
159	Numeric data was not received for numeric enter list item.
160	Repetition of input character more than 32 768 times.
161	Attempted to create CONVERT table or EOL sequence for source or destination variable which is locally defined in a subprogram.
162	Attempted to delete a nonexistent CONVERT table or EOL sequence.
163	I/O error, such as interface card not present, device timeout, interface or peripheral failure, or stop key hit. (Interface FLAG line=0.)
164	Transfer type specified is incorrect type for interface card.
165	A FHS or DMA transfer with no format specifies a count which exceeds size of variable, or image specifier indicates more characters than will fit in the specified variable.
166	A NOFORMAT FHS or DMA type transfer does not start on an odd numbered character position, sush as A$[3].
167	Interface status error, TRL Character or an EOI was received on an HP-IB Interface before ENTER list or image specification was satisfied.
184	Improper argument for OCTAL or DECIMAL Function.

Advanced Programming ROM Error Messages

330	Lexical table size exceeds array size.
331	Improper pointer array*.
332	Non-existent dimension specified in MAT REORDER.
333	Pointer array contains out-of-range subscript value.
334	Pointer array length does not equal number of records.
335	Pointer array is not one-dimensional
336	Number of records (plus twice the number of secondary keys plus twice the number of substrings) exceeds 16 383.
337	Subscript extends beyond dimensioned maximum length.
338	Subscript out-of-range in key specifier.
339	Starting location is an out-of-range subscript value.
340	Lexical table is too small to include all characters.
341	Main lexical table length plus mode section length does not equal specified table length.
342	Array is not one-dimensioned or is not integer.
343	Lexical mode section pointer out-of-range.
344	Lexical table length exceeds 16 383.

* This error occurs when data is lost in the process of reordering the array. If this error does not occur, it does not necessarily imply that the pointer array contains a permutation.

Subject Index

U

V

HEWLETT hp PACKARD

SALES & SERVICE OFFICES

AFRICA, ASIA, AUSTRALIA

ANGOLA
Telectra
Empresa Técnica de
Equipamentos
Eléctricos, S.A.R.L.
R. Barbosa Rodrigues, 42-I°DT.°
Caixa Postal, 6487
Luanda
Tel: 35515/6
Cable: TELECTRA Luanda

AUSTRALIA
Hewlett-Packard Australia
Pty. Ltd.
31-41 Joseph Street
Blackburn, Victoria 3130
P.O. Box 36
Doncaster East, Victoria 3108
Tel: 896351
Telex: 31-024
Cable: HEWPARD Melbourne

Hewlett-Packard Australia
Pty. Ltd.
31 Bridge Street
Pymble
New South Wales, 2073
Tel: 4496566
Telex: 21561
Cable: HEWPARD Sydney

Hewlett-Packard Australia
Pty. Ltd.
153 Greenhill Road
Parkside, S.A., 5063
Tel: 2725911
Telex: 82536
Cable: HEWPARD Adelaide

Hewlett-Packard Australia
Pty. Ltd.
141 Stirling Highway
Nedlands, W.A. 6009
Tel: 3865455
Telex: 93859
Cable: HEWPARD Perth

Hewlett-Packard Australia
Pty. Ltd.
121 Wollongong Street
Fyshwick, A.C.T. 2609
Tel: 804244
Telex: 62650
Cable: HEWPARD Canberra

Hewlett Packard Australia
Pty. Ltd.
5th Floor
Teachers Union Building
495-499 Boundary Street
Spring Hill, Queensland 4000
Tel: 2291544
Cable: HEWPARD Brisbane

BANGLADESH
The General Electric Co.
of Bangladesh Ltd.
Magnet House 72
Dilkusha Commercial Area
Motijheel, Dacca 2
Tel: 252415, 252419
Telex: 734
Cable: GECDAC Dacca

ETHIOPIA
Abdella Abdulmalik
P.O. Box 2635
Addis Ababa
Tel: 11 93 40

GUAM
Medical Only
Guam Medical Supply, Inc.
Suite C, Airport Plaza
P.O. Box 8947
Tamuning 96911
Tel: 646-4513
Cable: EARMED Guam

HONG KONG
Schmidt & Co. (Hong Kong) Ltd.
Wing On Centre, 28th Floor
Connaught Road, C.
Hong Kong
Tel: 5-455644
Telex: 74766 SCHMC HX

INDIA
Blue Star Ltd.
Kasturi Buildings
Jamshedji Tata Rd
Bombay 400 020
Tel: 29 50 21
Telex: 011-2156
Cable: BLUEFROST

Blue Star Ltd.
Sahas
414/2 Vir Savarkar Marg
Prabhadevi
Bombay 400 025
Tel: 45 78 87
Telex: 011-4093
Cable: FROSTBLUE

Blue Star Ltd.
Band Box House
Prabhadevi
Bombay 400 025
Tel: 45 73 01
Telex: 011-3751
Cable: BLUESTAR

Blue Star Ltd.
Bhavdeep
Stadium Road
Ahmedabad 380 014
Tel: 42880
Telex: 234
Cable: BLUEFROST

Blue Star Ltd.
7 Hare Street
P.O. Box 506
Calcutta 700 001
Tel: 23-0131
Telex: 021-7655
Cable: BLUESTAR

Blue Star Ltd.
Bhandari House
7th & 8th Floor
91 Nehru Place
New Delhi 110 024
Tel: 634770 & 635166
Telex: 031-2463
Cable: BLUESTAR

Blue Star Ltd.
Blue Star House
11/11A Magarath Road
Bangalore 560 025
Tel: 55668
Telex: 043-430
Cable: BLUESTAR

Blue Star Ltd.
Meeakshi Mandiram
xxx/1678 Mahatma Gandhi Rd.
Cochin 682 016
Tel: 32069,32161,32282
Telex: 0885-514
Cable: BLUESTAR

Blue Star Ltd.
1-1-117/1
Sarojini Devi Road
Secunderabad 500 003
Tel: 70126, 70127
Telex: 015-459
Cable: BLUEFROST

Blue Star Ltd.
2/34 Kodambakkam High Road
Madras 600 034
Tel: 82056
Telex: 041-379
Cable: BLUESTAR

INDONESIA
BERCA Indonesia P.T.
P.O. Box 496/Jkt.
Jln.Abdul Muis 62
Jakarta
Tel: 349255, 349886
Telex: 46748 BERSIL IA
Cable: BERSAL

BERCA Indonesia P.T.
P.O. Box 174/Sby.
23 Jln. Jimerto
Surabaya
Tel: 42027
Cable: BErcacon

ISRAEL
Electronics Engineering Div.
of Motorola Israel Ltd.
16, Kremenetski Street
P.O.Box 25016
Tel-Aviv
Tel: 38973
Telex: 33569, 34164
Cable: BASTEL Tel-Aviv

JAPAN
Yokogawa-Hewlett-Packard Ltd.
Chuo Bldg., 4th Floor
4-20, Nishinakajima 5-chome
Yodogawa-ku, Osaka-shi
Osaka,532
Tel: 06-304-6021
Telex: 523-3624

Yokogawa-Hewlett-Packard Ltd.
29-21, Takaido-Higashi 3-chome
Suginami-ku, **Tokyo** 168
Tel: 03-331-6111
Telex: 232-2024 YHP-Tokyo
Cable: YHPMARKET TOK 23 724

Yokogawa-Hewlett-Packard Ltd.
Nakamo Building
24 Kami Sasajima-cho
Nakamura-ku, **Nagoya**, 450
Tel: 052 571-5171

Yokogawa-Hewlett-Packard Ltd.
Tanigawa Building
2-24-1 Tsuruya-cho
Kanagawa-ku
Yokohama, 221
Tel: 045-312-1252
Telex: 382-3204 YHP YOK

Yokogawa-Hewlett-Packard Ltd.
Mito Mitsui Building
105, 1-chome, San-no-maru
Mito, Ibaragi 310
Tel: 0292-25-7470

Yokogawa-Hewlett-Packard Ltd.
Inoue Building
1348-3, Asahi-cho, 1-chome
Atsugi, Kanagawa 243
Tel: 0462-24-0452

Yokogawa-Hewlett-Packard Ltd.
Kumagaya Asahi
Hachijuni Building
4th Floor
3-4, Tsukuba
Kumagaya, Saitama 360
Tel: 0485-24-6563

KENYA
Advanced Communications Ltd.
P.O. Box 30070
Nairobi
Tel: 331955
Telex: 22639

Medical Only
International Aeradio (E.A.) Ltd.
P.O. Box 19012
Nairobi Airport
Nairobi
Tel: 336055/56
Telex: 22201/22301
Cable: INTAERIO Nairobi

Medical Only
International Aeradio (E.A.) Ltd.
P.O. Box 95221
Mombasa

KOREA
Samsung Electronics Co., Ltd.
15th Floor, Daeyongak Bldg.,
25-5, 1-KA
Choong Moo-Ro, Chung-Ku,
Seoul
Tel: (23) 6811, 778-3401/2/3/4
Telex: 2257S

MALAYSIA
Hewlett-Packard Sales SDN BHD
Suite 2.21/2.22
Bangunan Angkasa Raya
Jalan Ampang
Kuala Lumpur
Tel: 23320/27491

Protel Engineering
P.O. Box 1917
Lot 259, Satok Road
Kuching, **Sarawak**
Tel: 53544
Cable: PROTELENG

MOZAMBIQUE
A.N. Goncalves, Ltd.
162, 1° Apt. 14 Av. D. Luis
Caixa Postal 107
Maputo
Tel: 27091, 27114
Telex: 6-203 NEGON Mo
Cable: NEGON

NEW GUINEA
Hewlett-Packard Australia
Pty. Ltd.
Development Bank Building
Ground Floor
Ward Strip
Port Moresby, Paupua
Tel: 258933

NEW ZEALAND
Hewlett-Packard (N.Z.) Ltd.
4-12 Cruickshank Street
Kilbirnie, Wellington 3
P.O. Box 9443
Courtney Place
Wellington
Tel: 877-199
Cable: HEWPACK Wellington

Hewlett-Packard (N.Z.) Ltd.
Pakuranga Professional Centre
267 Pakuranga Highway
Box 51092
Pakuranga
Tel: 569-651
Cable: HEWPACK Auckland

Analytical/Medical Only
Medical Supplies N.Z. Ltd.
Scientific Division
79 Carlton Gore Road, Newmarket
P.O. Box 1234
Auckland
Tel: 75-289
Cable: DENTAL Auckland

Analytical/Medical Only
Medical Supplies N.Z. Ltd.
Norrie and Parumoana Streets
Porirua
Tel: 75-098
Telex: 3858

Analytical/Medical Only
Medical Supplies N.Z. Ltd.
P.O. Box 309
239 Stanmore Road
Christchurch
Tel: 892-019
Cable: DENTAL Christchurch

Analytical/Medical Only
Medical Supplies N.Z. Ltd.
303 Great King Street
P.O. Box 233
Dunedin
Tel: 88-817
Cable: DENTAL Dunedin

NIGERIA
The Electronics
Instrumentations Ltd.
N6B/770 Oyo Road
Oluseun House
P.M.B. 5402
Ibadan
Tel: 461577
Telex: 31231 TEIL NG
Cable: THETIEL Ibadan

The Electronics Instrumentations Ltd.
144 Agege Motor Road, Mushin
P.O. Box 6645
Lagos
Cable: THETEIL Lagos

PAKISTAN
Mushko & Company Ltd.
Oosman Chambers
Abdullah Haroon Road
Karachi-3
Tel: 511027, 512927
Telex: 2894
Cable: COOPERATOR Karachi

Mushko & Company, Ltd.
38B, Satellite Town
Rawalpindi
Tel: 41924
Cable: FEMUS Rawalpindi

PHILIPPINES
The Online Advanced
Systems Corporation
Rico House
Amorsolo cor. Herrera Str.
Legaspi Village, **Makati**
P.O. Box 1510
Metro Manila
Tel: 85-35-81, 85-34-91,85-32-21
Telex: 3274 ONLINE

RHODESIA
Field Technical Sales
45 Kelvin Road North
P.O. Box 3458
Salisbury
Tel: 705231 (5 lines)
Telex: RH 4122

SINGAPORE
Hewlett-Packard Singapore
(Pte.) Ltd.
1150 Depot Road
Alexandra P.O. Box 58
Singapore 4
Tel: 270-2355
Telex: HPSG RS 21486
Cable: HEWPACK, Singapore

SOUTH AFRICA
Hewlett-Packard South Africa
(Pty.) Ltd.
Private Bag Wendywood,
Sandton, Transvaal, 2144
Hewlett-Packard Centre
Daphne Street, Wendywood,
Sandton, 2144
Tel: 802-1040/6
Telex: 8-4782
Cable: HEWPACK Johannesburg

Hewlett-Packard South Africa
(Pty.), Ltd.
P.O. Box 120
Howard Place, Cape Province, 7450
Pine Park Centre, Forest Drive,
Pinelands, Cape Province, 7405
Tel: 53-7955 thu 9
Telex: 57-0006

SRI LANKA
Metropolitan Agencies Ltd.
209/9 Union Place
Colombo 2
Tel: 35947
Telex: 1377METROLTO CE
Cable: METROLTO

SUDAN
Radison Trade
P.O. Box 921
Khartoum
Tel: 44048
Telex: 375

TAIWAN
Hewlett-Packard Far East Ltd.
Taiwan Branch
39 Chung Hsiao West Road
Section 1, 7th Floor
Taipei
Tel: 3819160-9,3141010
Cable: HEWPACK TAIPEI

Hewlett-Packard Far East Ltd.
Taiwan Branch
68-2, Chung Cheng 3rd. Road
Kaohsiung
tel: (07) 242318-Kaohsiung

Analytical Only
San Kwang Instruments Co., Ltd.
20 Yung Sui Road
Taipei
Tel: 3615446-9 (4 lines)
Telex: 22894 SANKWANG
Cable: SANKWANG Taipei

TANZANIA
Medical Only
International Aeradio (E.A.), Ltd.
P.O. Box 861
Dar es Salaam
Tel: 21251 Ext. 265
Telex: 41030

THAILAND
UNIMESA Co. Ltd.
Elcom Research Building
2538 Sukumvit Ave.
Bangchak,Bangkok
Tel: 3932387, 3930338
Cable: UNIMESA Bangkok

UGANDA
Medical Only
International Aeradio (E.A.), Ltd.
P.O. Box 2577
Kampala
Tel: 54388
Cable: INTAERIO Kampala

ZAMBIA
R.J. Tilbury (Zambia) Ltd.
P.O. Box 2792
Lusaka
Tel: 73793
Cable: ARJAYTEE, Lusaka

OTHER AREAS NOT LISTED, CONTACT:
Hewlett-Packard Intercontinental
3200 Hillview Ave.
Palo Alto, California 94304
Tel: (415) 856-1501
TWX: 910-373-1267
Cable: HEWPACK Palo Alto
Telex: 034-8300, 034-8493

CANADA

ALBERTA
Hewlett-Packard (Canada) Ltd.
11620A - 168th Street
Edmonton T5M 3T9
Tel: (403) 452-3670
TWX: 610-831-2431

Hewlett-Packard (Canada) Ltd.
210,7220 Fisher St. S.E.
Calgary T2H 2H6
Tel: (403) 253-2713
Twx: 610-821-6141

BRITISH COLUMBIA
Hewlett-Packard (Canada) Ltd.
10691 Shellbridge Way
Richmond V6X 2W7
Tel: (604) 270-2277
TWX: 610-925-5059

MANITOBA
Hewlett-Packard (Canada) Ltd.
380-550 Century St.
Winnipeg R3H 0Y1
Tel: (204) 786-6701
TWX: 610-671-3531

NOVA SCOTIA
Hewlett-Packard (Canada) Ltd.
800 Windmill Road
Dartmouth B3B 1L1
Tel: (902) 469-7820
TWX: 610-271-4482

ONTARIO
Hewlett-Packard (Canada) Ltd.
1020 Morrison Dr.
Ottawa K2H 8K7
Tel: (613) 820-6483
TWX: 610-563-1636

Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
Mississauga L4V 1M8
Tel: (416) 678-9430
TWX: 610-492-4246

Hewlett-Packard (Canada) Ltd.
552 Newbold Street
London N6E 2S5
Tel: (519) 686-9181

QUEBEC
Hewlett-Packard (Canada) Ltd.
275 Hymus Blvd.
Pointe Claire H9R 1G7
Tel: (514) 697-4232
TWX: 610-422-3022
TLX: 05-821-521 HPCL

FOR CANADIAN AREAS NOT LISTED:
Contact Hewlett-Packard (Canada)
Ltd. in Mississauga.

2/79

CENTRAL AND SOUTH AMERICA

ARGENTINA
Hewlett-Packard Argentina
S.A.
Av. Leandro N. Alem 822 - 12°
1001 **Buenos Aires**
Tel: 31-6063,4,5,6
Telex: 122443 AR CIGY
Cable: HEWPACKARG

Biotron S.A.C.I.y M.
Bolivar 177
1066 **Buenos Aires**
Tel: 30-4846, 34-9356, 34-0460,
33-2863
Telex: 011-7595
Cable: Biotron Baries

BOLIVIA
Casa Kavlin S.A.
Calle Potosi' 1130
P.O. Box 500
La Paz
Tel: 41530,53221
Telex: CWC BX 5298,ITT 3560082
Cable: KAVLIN

BRAZIL
Hewlett-Packard do Brasil
I.e.C. Ltda.
Alameda Rio Negro, 750
Alphaville
06400 **Barueri** SP
Tel: 429-3222
Cable: HEWPACK Sao Paulo

Hewlett-Packard do Brasil
I.e.C. Ltda.
Rua Padre Chagas, 32
90000-**Pôrto Alegre**-RS
Tel: (0512) 22-2998, 22-5621
Cable: HEWPACK Porto Alegre

Hewlett-Packard do Brasil
I.e.C. Ltda.
Av. Epitacio Pessoa, 4664
Lagoa
20000-**Rio de Janeiro**-RJ
Tel:
Telex: 021-21905 HPBR-BR
Cable: HEWPACK
Rio de Janeiro

CHILE
Jorge Calcagni y Cia. Ltda
Vicuna MacKenna 3, Ofic. 1204
Casilla 16475
Correo 9, **Santiago**
Tel: 34152
Telex: JCALCAGNI

COLOMBIA
Instrumentación
Henrik A. Langebaek & Kier S.A.
Carrera 7 No. 48-75
Apartado Aéreo 6287
Bogotá, I.D.E.
Tel: 269-8877
Telex: 44400
Cable: AARIS Bogotá

Instrumentacion
H.A. Langebaek & Kier S.A.
Carrera 63 No. 49-A-31
Apartado 54098
Medellin
Tel: 304475

COSTA RICA
Científica Costarricense S.A.
Avenida 2, Calle 5
San Pedro de Montes de Oca
Apartado 10159
San José
Tel: 24-38-20, 24-08-19
Telex: 2367 GALGUR CR
Cable: GALGUR

ECUADOR
Computadoras y Equipos
Electrónicos del Ecuador
P.O. Box 6423 CCI
Eloy Alfaro No. 1824,3°Piso
Quito
Tel: 453 482
Telex: 2548 CYEDE ED
Cable: CYEDE-Quito

Medical Only
Hospitalar S.A.
Casilla 3590
Robles 625
Quito
Tel: 545-250
Cable: HOSPITALAR-Quito

EL SALVADOR
IPESA
Bulevar de los Heroes 11-48
San Salvador
Tel: 252787

GUATEMALA
IPESA
Avenida Reforma 3-48,
Zona 9
Guatemala City
Tel: 316627,314786,66471-5,ext.9
Telex: 4192 Teletro Gu

MEXICO
Hewlett-Packard Mexicana,
S.A. de C.V.
Av. Periférico Sur No. 6501
Tepepan, Xochimilco
Mexico 23, D.F.
Tel: 905-676-4600
Telex: 017-74-507

Hewlett-Packard Mexicana,
S.A. de C.V.
Ave. Constitución No. 2184
Monterrey, N.L.
Tel: 48-71-32, 48-71-84
Telex: 038-410

NICARAGUA
Roberto Terán G.
Apartado Postal 689
Edificio Terán
Managua
Tel: 25114, 23412,23454,22400
Cable: ROTERAN Managua

PANAMA
Electrónico Balboa, S.A.
Aparatado 4929
Panama 5
Calle Samuel Lewis
Edificio "Alfa", No.2
Cuidad de Panama
Tel: 64-2700
Telex: 3483103 Curundu,
Canal Zone
Cable: ELECTRON Panama

PERU
Compañía Electro Médica S.A.
Los Flamencos 145
San Isidro Casilla 1030
Lima 1
Tel: 41-4325
Telex: Pub. Booth 25424 SISIDRO
Cable: ELMED Lima

SURINAME
Surtel Radio Holland N.V.
Grote Hofstr. 3-5
P.O. Box 155
Paramaribo
Tel: 72118, 77880
Cable: Surtel

TRINIDAD & TOBAGO
CARTEL
Caribbean Telecoms Ltd.
P.O. Box 732
69 Frederick Street
Port-of-Spain
Tel: 62-53068

URUGUAY
Pablo Ferrando S.A.C.el.
Avenida Italia 2877
Casilla de Correo 370
Montevideo
Tel: 40-3102
Telex: 702 Public Booth Para
Pablo Ferrando
Cable: RADIUM Montevideo

VENEZUELA
Hewlett-Packard de Venezuela
C.A.
P.O. Box 50933
Caracas 105
Los Ruices Norte
3a Transversal
Edificio Segre
Caracas 107
Tel: 239-4133 (20 lines)
Telex: 25146 HEWPACK
Cable: HEWPACK Caracas

FOR AREAS NOT LISTED, CONTACT:
Hewlett-Packard
Inter-Americas
3200 Hillview Ave.
Palo Alto, California 94304
Tel: (415) 856-1501
TWX: 910-373-1260
Cable: HEWPACK Palo Alto
Telex: 034-8300, 034-8493

EUROPE, NORTH AFRICA AND MIDDLE EAST

AUSTRIA
Hewlett Packard Ges m b H
Handelskai 52
P O Box 7
A-1205 Vienna
Tel 351621-27
Cable HEWPAK Vienna
Telex 75923 hewpak a

BAHRAIN
Medical Only
[illegible]
[illegible]
Bahrain
Tel 54886, 56123
Telex 8550 WAEL GJ
Cable WAEL-PHARM

Analytical Only
Al Hamidiya Trading and Contracting
P O Box 20074
Manama
Tel 259978, 259958
Telex 8895 KALDIA GJ

BELGIUM
Hewlett-Packard Benelux S A /N.V
Avenue du Col-Vert, 1, (Groenkraaglaan)
B-1170 Brussels
Tel (02) 660 50 50
Cable PALOBEN Brussels
Telex 23-494 paloben bru

CYPRUS
Kypronics
19 Gregorios Xenopoulos Street
P O Box 1152
Nicosia
Tel 45628/29
Cable Kypronics Pandehis
Telex 3018

CZECHOSLOVAKIA
Vyvojova a Provozni Zakladna Vyzkumnych Ustavu v Bechovicich
CSSR-25097 Bechovice u Prahy
Tel 89 93 41
Telex 12133

Institute of Medical Bionics
Vyskumny Ustav Lekarskej Bioniky
Jedlova 6
CS-88346 Bratislava-Kramare
Tel 4251
Telex 93229

DDR
Entwicklungslabor der TU Dresden
Forschungsinstitut Meinsberg
DDR-7305 Waldheim/Meinsberg
Tel. 37 667
Telex: 518741

Export Contact AG Zuerich
Guenther Forgber
Schlegelstrasse 15
1040 Berlin
Tel. 42-74-12
Telex: 111889

DENMARK
Hewlett-Packard A/S
Datavej 52
DK-3460 Birkerod
Tel (02) 81 66 40
Cable HEWPACK AS
Telex 37409 hpas dk

Hewlett-Packard A/S
Navervej 1
DK-8600 Silkeborg
Tel (06) 82 71 66
Telex 37409 hpas dk
Cable HEWPACK AS

EGYPT
I.E.A.
International Engineering Associates
24 Hussein Hegazi Street
Kasr-el-Aini
Cairo
Tel 23 829
Telex: 93830
Cable INTENGASSO

SAMITRO
Sami Amin Trading Office
18 Abdel Aziz Gawish
Abdine-Cairo
Tel 24932
Cable SAMITRO CAIRO

FINLAND
Hewlett-Packard OY
Nahkahousuntie 5
P O Box 6
SF-00211 Helsinki 21
Tel (90) 6923031

FRANCE
Hewlett-Packard France
Avenue des Tropiques
Les Ulis
Boite Postale No. 6
91401 Orsay-Cedex
Tel (1) 907 78 25
TWX 600048F

Hewlett-Packard France
Chemin des Mouilles
B.P. 162
69130 Ecully
Tel (78) 33 81 25,
TWX 310617F

Hewlett-Packard France
Péricentre de la Cépière
31081 Toulouse-Le Mirail
Tel:(61) 40 11 12
TWX: 510957F

Hewlett-Packard France
Le Ligoures
Bureau de vente de Marseilles
Place Rouée de Villeneuve
13100 Aix-en-Provence
Tel. (42) 59 41 02

Hewlett-Packard France
2, Allee de la Bourgnette
35100 Rennes
Tel: (99) 51 42 44
TWX 740912F

Hewlett-Packard France
18, rue du Canal de la Marne
67300 Schiltigheim
Tel: (88) 83 08 10
TWX: 890141F

Hewlett-Packard France
Immeuble péricentre
Rue van Gogh
59650 Villeneuve D Ascq
Tel: (20) 91 41 25
TWX: 160124F

Hewlett-Packard France
Bureau de Vente
Centre d' affaires Paris-Nord
Bâtiment Ampère
Rue de la Commune de Paris
B.P. 300
93153 Le Blanc Mesnil Cédex
Tel: (01) 931 88 50

Hewlett-Packard France
Av. du Pdt. Kennedy
33700 Merignac
Tel: (56) 97 22 69

Hewlett-Packard France
"France-Evry" immeuble Lorraine
Boilevard de France
91035 Evry-Cedex
Tel: 077 96 60

Hewlett-Packard France
60, Rue de Metz
57130 Jouy aux Arches
Tel: (87) 89 45 32

GERMAN FEDERAL REPUBLIC
Hewlett-Packard GmbH
Vertriebszentrale Frankfurt
Berner Strasse 117
Postfach 560 140
D-6000 Frankfurt 56
Tel: (0611) 50-04-1
Cable: HEWPACKSA Frankfurt
Telex: 04 13249 hpffm d

Hewlett-Packard GmbH
Technisches Büro Böblingen
Herrenberger Strasse 110
D-7030 Böblingen, Württemberg
Tel: (0703) 667-1
Cable: HEWPACK Böblingen
Telex: 07265739 bbn

Hewlett-Packard GmbH
Technisches Büro Düsseldorf
Emanuel-Leutze-Str.1 (Seestern)
D-4000 Düsseldorf
Tel: (0211) 59711
Telex: 085/86 533 hpdd d

Hewlett-Packard GmbH
Technisches Büro Hamburg
Wendenstrasse 23
D-2000 Hamburg 1
Tel: (040) 24 13 93
Cable: HEWPACKSA Hamburg
Telex: 21 63 032 hphh d

Hewlett-Packard GmbH
Technisches Büro Hannover
Am Grossmarkt 6
D-3000 Hannover 91
Tel: (0511) 46 60 01
Telex 092 3259

Hewlett-Packard GmbH
Technisches Büro Nürnberg
Neumeyerstrasse 90
D-8500 Nürnberg
Tel: (0911) 56 30 83
Telex 0623 860

Hewlett-Packard GmbH
Technisches Büro München
Eschenstrasse 5
D-8021 Taufkirchen
Tel. (089) 6117-1

Hewlett-Packard GmbH
Technisches Büro Berlin
Kaithstrasse 2-4
D-1000 Berlin 30
Tel: (030) 24 90 86
Telex 018 3405 hpbln d

GREECE
Kostas Karayannis
8 Omirou Street
Athens 133
Tel: 32 30 303/32/37 731

Analytical Only
INTECO
G. Papathanassiou & Co.
17 Marni Street
Athens 103
Tel: 5522 915/5221 989
Telex: 21 5329 INTE GR
Cable: INTEKNIKA

Medical Only
Technomed Hellas Ltd.
52 Skoufa Street
Athens 135
Tel: 3626 972

HUNGARY
MTA
Müszerügy és Méréstechnikai Szolgalata
Hewlett-Packard Service
Lenin Krt. 67, P.O.Box 241
1391Budapest VI
Tel. 42 03 38
Telex: 22 51 14

ICELAND
Medical Only
Elding Trading Company Inc.
Hafnarnvoli - Tryggvagötu
P.O.Box 895
IS-Reykjavik
Tel: 1 58 20/1 63 03
Cable: ELDING Reykjavik

IRAN
Hewlett-Packard Iran Ltd.
No. 13, Fourteenth St.
Mir Emad Avenue
P O. Box 41/2419
Tehran
Tel: 851082-5
Telex: 213405 hewp ir

IRELAND
Hewlett-Packard Ltd.
King Street Lane
Winnersh, Wokingham
Berks, RG11 5AR
GB-England
Tel: (0734) 78 47 74
Telex: 847178
Cable: Hewpie London

Hewlett-Packard Ltd.
2C Avonbeg Industrial Estate
Long Mile Road
Dublin 12, Eire
Tel: (01) 514322
Telex: 30439

Medical Only
Cardiac Services (Ireland) Ltd.
Kilmore Road
Artane
Dublin 5, Eire
Tel: (01) 315820

Medical Only
Cardiac Services Co.
95A Finaghy Rd. South
Belfast BT10 0BY
GB-Northern Ireland

ITALY
Hewlett-Packard Italiana S p A
Via G Di Vittorio, 9
20063 Cernusco Sul Naviglio (MI)
Tel (2) 903691
Telex 311046 HEWPACKIT

Hewlett-Packard Italiana S p A
Via Turazza 14
35100 Padova
Tel (49) 664888
Telex 41612 HEWPACKIT

Hewlett-Packard Italiana S p A
Via G. Armellini 10
I-00143 Roma
Tel (06) 54 69 61
Telex 61514
Cable HEWPACKIT Roma

Hewlett-Packard Italiana S.p.A.
Corso Giovanni Lanza 94
I-10133 Torino
Tel:(011) 682245/659308

Medical/Calculators Only
Hewlett-Packard Italiana S.p.A.
Via Principe Nicola 43 G/C
I-95126 Catania
Tel:(095) 37 05 04

Hewlett-Packard Italiana S.p.A.
Via Nuova San Rocco A. Capodimonte, 62A
I-80131 Napoli
Tel: (081) 7913544

Hewlett-Packard Italiana S.p.A.
Via E. Masi, 9/8
I-40137 Bologna
Tel: (051) 307887/300040

JORDAN
Mouasher Cousins Co.
P.O. Box 1387
Amman
Tel. 24907/39907
Telex: SABCO JO 1456
Cable: MOUASHERCO

KUWAIT
Al-Khaldiya Trading & Contracting
P.O. Box 830-Safat
Kuwait
Tel:42 4910/41 1726

LUXEMBURG
Hewlett-Packard Benelux S.A./N.V.
Avenue du Col-Vert, 1 (Groenkraaglaan)
B-1170 Brussels
Tel: (02) 672 22 40
Cable: PALOBEN Brussels
Telex: 23 494

MOROCCO
Dolbeau
81 rue Karatchi
Casablanca
Tel: 3041 82
Telex: 23051/22822
Cable: MATERIO

Gerep
3, rue d'Agadir
Casablanca
Tel: 272093/5
Telex: 23739
Cable: GEREP-CASA

Cogedir
31 rue Omar Slaoui
Casablanca
Tel: 27 65 40
Telex: 21737/23003
Cable: COGEDIR

NETHERLANDS
Hewlett-Packard Benelux N.V.
Van Heuven Goedhartlaan 121
P.O. Box 667
NL-Amstelveen 1134
Tel: (020) 47 20 21

NORWAY
Hewlett-Packard Norge A/S
Osterdalen 18
P.O. Box 34
1345 Osteraas
Tel: (02) 1711 80
Telex: 16621 hpnas n

Hewlett-Packard Norge A/S
Nygaardsgaten 114
5000 Bergen

POLAND
Biuro Informacji Technicznej
Hewlett-Packard
Ul Stawki 2, 6P
00-950 Warszawa
Tel 33 25 88 39 67 43
Telex 81 24 53 hepa pl

UNIPAN
Biuro Obslugi Technicznej
01-447 Warszawa
ul Newelska 6
Poland

Zaklady Naprawcze Sprzetu Medycznego
Plac Komuny Paryskiej 6
90-007 Lodz
Tel 334-41, 337-83
Telex 886981

PORTUGAL
Telectra-Empresa Técnica de Equipamentos Eléctricos S.a.r.l.
Rua Rodrigo da Fonseca 103
P.O. Box 2531
P-Lisbon 1
Tel: (19) 68 60 72
Cable: TELECTRA Lisbon
Telex: 12598

Medical only
Mundinter
Intercambio Mundial de Comércio S.a.r.l.
P.O. Box 2761
Avenida Antonio Augusto de Aguiar 138
P - Lisbon
Tel: (19) 53 21 31/7
Telex: 16691 munter p
Cable: INTERCAMBIO Lisbon

QATAR
Nasser Trading & Contracting
P.O. Box 1563
Doha
Tel: 22170
Telex: 4439 NASSER
Cable: NASSER

RUMANIA
Hewlett-Packard Reprezentanta
Bd.n. Balcescu 16
Bucuresti
Tel: 15 80 23/13 88 85
Telex: 10440

I.I.R.U.C.
Intreprinderea Pentru Intretinerea
Si Repararea Utilajelor de Calcul
B-dul Prof. Dimitrie Pompei 6
Bucuresti-Sectorul 2
Tel: 88-20-70, 88-24-40, 88-67-95
Telex: 118

SAUDI ARABIA
Modern Electronic Establishment (Head Office)
P.O. Box 1228, Baghdadiah Street
Jeddah
Tel: 27 798
Telex: 40035
Cable: ELECTA JEDDAH

Modern Electronic Establishment (Branch)
P.O. Box 2728
Riyadh
Tel: 62596/66232
Cable: RAOUFCO

Modern Electronic Establishment (Branch)
P.O. Box 193
Al-Khobar
Tel: 44678-44813

SPAIN
Hewlett-Packard Española, S.A.
Calle Jerez 3
E-Madrid 16
Tel: (1) 458 26 00 (10 lines)
Telex: 23515 hpe

Hewlett-Packard Espadñola S.A.
Colonia Mirasierra
Edificio Juban
% Costa Brava, 13
Madrid 34

Hewlett-Packard Española, S.A.
Milanesado 21-23
E-Barcelona 17
Tel: (3) 203 6200 (5 lines)

Hewlett-Packard Española, S.A.
Av Ramón y Cajal, 1
Edificio Sevilla, planta 9°
E-Sevilla 5
Tel 64 44 54/58

Hewlett-Packard Española S.A.
Edificio Albia II 7° B
E-Bilbao 1
Tel 23 83 06/23 82 06

Hewlett-Packard Española S.A.
C/ Ramon Gordillo 1
Entlo.
E-Valencia-10
Tel 36-361 13.54/361 13 58

SWEDEN
Hewlett-Packard Sverige AB
Enighetsvägen 3, Fack
S-161 Bromma 20
Tel: (08) 730 05 50
Telex: 10721
Cable: MEASUREMENTS Stockholm

Hewlett-Packard Sverige AB
Frötallisgatan 30
S-421 32 Västra Frölunda
Tel: (031) 49 09 50
Telex: 10721 via Bromma office

SWITZERLAND
Hewlett-Packard (Schweiz) AG
Zürcherstrasse 20
P.O. Box 307
CH-8952 Schlieren-Zurich
Tel: (01) 7305240
Telex: 53933 hpag ch
Cable: HPAG CH

Hewlett-Packard (Schweiz) AG
Château Bloc 19
CH-1219 Le Lignon-Geneva
Tel: (022) 96 03 22
Telex: 27333 hpag ch
Cable: HEWPACKAG Geneva

SYRIA
General Electronic Inc.
Nuri Basha-Ahnaf Ebn Kays Street
P.O. Box 5781
Damascus
Tel: 33 24 87
Telex: 11215 ITIKAL
Cable: ELECTROBOR DAMASCUS

Medical/Personal Calculator only
Sawah & Co.
Place Azmé
B.P. 2308
Damascus
Tel: 16 367-19 697-14 268

Suleiman Hilal El Mlawi
P O Box 2528
Mamoun Bitar Street, 56-58
Damascus Tel: 11 46 63

TUNISIA
Tunisie Electronique
31 Avenue de la Liberte
Tunis
Tel: 280 144

Corema
1 ter. Av. de Carthage
Tunis
Tel: 253 821
Telex: 12319 CABAM TN

TURKEY
TEKNIM Company Ltd.
Riza Sah Pehlevi Caddesi No. 7
Kavaklidere, Ankara
Tel: 275800
Telex: 42155

Teknim Com., Ltd.
Barbaros Bulvari 55/12
Besikyas, Istanbul
Tel: 613 546
Telex: 23540

Medical only
E.M.A.
Muhendislik Kollektif Sirketi
Mediha Eldem Sokak 41/6
Yüksel Caddesi
Ankara
Tel: 17 56 22
Cable: EMATRADE/Ankara

Analytical only
Yilmaz Ozyurek
Milli Mudafaa Cad 18/6
Kizilay
Ankara
Tel 25 03 09 - 17 80 26
Telex 42576 OZEK TR
Cable OZYUREK ANKARA

UNITED ARAB EMIRATES
Emitac Ltd (Head Office)
P O Box 1641
Sharjah
Tel 354121 3
Telex 8136

Emitac Ltd. Branch Office
P O Box 2711
Abu Dhabi
Tel 331370 1

UNITED KINGDOM
Hewlett-Packard Ltd
King Street Lane
Winnersh, Wokingham
Berks. RG11 5AR
Tel: (0734) 784774
Telex 847178/9

Hewlett-Packard Ltd.
Trafalgar House
Navigation Road
Altrincham
Cheshire WA14 1NU
Tel: (061) 928 8422
Telex: 668068

Hewlett-Packard Ltd.
Lygon Court
Hereward Rise
Dudley Road
Halesowen,
West Midlands B62 8SD
Tel: (021) 550 9911
Telex: 339105

Hewlett-Packard Ltd.
Wedge House
799, London Road
Thornton Heath
Surrey CR4 8XL
Tel: (01) 6840103
Telex: 946825

Hewlett-Packard Ltd
10, Wesley St.
Castleford
Yorks WF10 1AE
Tel: (0977) 550016
Telex: 557355

Hewlett-Packard Ltd
1, Wallace Way
Hitchin
Hertfordshire, SG4 0SE
Tel: (0462) 31111
Telex: 82.59.81

USSR
Hewlett-Packard
Representative Office USSR
Pokrovsky Boulevard 4/17-kw 12
Moscow 101000
Tel: 294.20.24
Telex: 7825 hewpak su

YUGOSLAVIA
Iskra-Standard/Hewlett-Packard
Miklosiceva 38/VII
61000 Ljubljana
Tel: 31 58 79/32 16 74

SOCIALIST COUNTRIES NOT SHOWN PLEASE CONTACT:
Hewlett-Packard Ges.m.b.H
Handelskai 52
P.O. Box 7
A-1205 Vienna, Austria
Tel: (0222) 35 18 21 to 27

MEDITERRANEAN AND MIDDLE EAST COUNTRIES NOT SHOWN PLEASE CONTACT:
Hewlett-Packard S.A.
Mediterranean and Middle East Operations
35, Kolokotroni Street
Platia Kefallariou
GR-Kifisaia-Athens, Greece
Tel: 8080337/259/429

FOR OTHER AREAS NOT LISTED CONTACT
Hewlett-Packard S.A.
7, rue du Bois-du-Lan
P O. Box
CH-1217 Meyrin 2 - Geneva
Switzerland
Tel: (022) 82 70 00

UNITED STATES

ALABAMA
P O Box 4207
8290 Whitesburg Dr.
Huntsville 35802
Tel (205) 881-4591

8933 E Roebuck Blvd.
Birmingham 35206
Tel. (205) 836-2203/2

ARIZONA
2336 E. Magnolia St.
Phoenix 85034
Tel (602) 244-1361

2424 East Aragon Rd.
Tucson 85706
Tel. (602) 889-4661

***ARKANSAS**
Medical Service Only
P O Box 5646
Brady Station
Little Rock 72215
Tel. (501) 376-1844

CALIFORNIA
1579 W. Shaw Ave.
Fresno 93771
Tel (209) 224-0582

1430 East Orangethorpe Ave.
Fullerton 92631
Tel (714) 870-1000

3939 Lankershim Boulevard
North Hollywood 91604
Tel (213) 877-1282
TWX: 910-499-2671

5400 West Rosecrans Blvd.
P O Box 92105
World Way Postal Center
Los Angeles 90009
Tel (213) 776-7500
TWX 910-325-6608

*Los Angeles
Tel (213) 776-7500

3003 Scott Boulevard
Santa Clara 95050
Tel (408) 988-7000

*Ridgecrest
Tel (714) 446-6165

646 W North Market Blvd
Sacramento 95834
Tel (916) 929-7222

9606 Aero Drive
P.O. Box 23333
San Diego 92123
Tel: (714) 279-3200

*Tarzana
Tel: (213) 705-3344

COLORADO
5600 OTC Parkway
Englewood 80110
Tel: (303) 771-3455

CONNECTICUT
12 Lunar Drive
New Haven 06525
Tel. (203) 389-6551
TWX: 710-465-2029

FLORIDA
P.O. Box 24210
2727 N.W. 62nd Street
Ft. Lauderdale 33309
Tel: (305) 973-2600

4428 Emerson Street
Unit 103
Jacksonville 32207
Tel: (904) 725-6333

P.O. Box 13910
6177 Lake Ellenor Dr.
Orlando 32809
Tel: (305) 859-2900

P.O. Box 12826
Suite 5, Bldg 1
Office Park North
Pensacola 32575
Tel. (904) 476-8422

GEORGIA
P.O. Box 105005
450 Interstate North Parkway
Atlanta 30348
Tel: (404) 955-1500

Medical Service Only
*Augusta 30903
Tel: (404) 736-0592

P O Box 2103
1172 N. Davis Drive
Warner Robins 31098
Tel: (912) 922-0449

HAWAII
2875 So. King Street
Honolulu 96826
Tel. (808) 955-4455

ILLINOIS
5201 Tollview Dr.
Rolling Meadows 60008
Tel: (312) 255-9800
TWX: 910-687-2260

INDIANA
7301 North Shadeland Ave.
Indianapolis 48250
Tel: (317) 842-1000
TWX: 810-260-1797

IOWA
2415 Heinz Road
Iowa City 52240
Tel: (319) 338-9466

KENTUCKY
Medical Only
3901 Atkinson Dr.
Suite 407 Atkinson Square
Louisville 40218
Tel: (502) 456-1573

LOUISIANA
P.O. Box 1449
3229-39 Williams Boulevard
Kenner 70063
Tel: (504) 443-6201

MARYLAND
7121 Standard Drive
Parkway Industrial Center
Hanover 21076
Tel: (301) 796-7700
TWX: 710-862-1943

2 Choke Cherry Road
Rockville 20850
Tel: (301) 948-6370
TWX: 710-828-9684

MASSACHUSETTS
32 Hartwell Ave.
Lexington 02173
Tel: (617) 861-8960
TWX: 710-326-6904

MICHIGAN
23855 Research Drive
Farmington Hills 48024
Tel: (313) 476-6400

724 West Centre Ave.
Kalamazoo 49002
Tel: (606) 323-8362

MINNESOTA
2400 N. Prior Ave.
St. Paul 55113
Tel: (612) 636-0700

MISSISSIPPI
322 N. Mart Plaza
Jackson 39206
Tel: (601) 982-9363

MISSOURI
11131 Colorado Ave.
Kansas City 64137
Tel: (816) 763-8000
TWX: 910-771-2087

1024 Executive Parkway
St. Louis 63141
Tel: (314) 878-0200

NEBRASKA
Medical Only
7101 Mercy Road
Suite 101
Omaha 68106
Tel: (402) 392-0948

NEVADA
*Las Vegas
Tel: (702) 736-6610

NEW JERSEY
W. 120 Century Rd.
Paramus 07652
Tel: (201) 265-5000
TWX: 710-990-4951

Crystal Brook Professional Building, Route 35
Eatontown 07724
Tel:(201) 542-1384

NEW MEXICO
P.O. Box 11634
Station E
11300 Lomas Blvd., N.E.
Albuquerque 87123
Tel (505) 292-1330
TWX: 910-989-1185

156 Wyatt Drive
Las Cruces 88001
Tel (505) 526-2484
TWX: 910-9983-0550

NEW YORK
6 Automation Lane
Computer Park
Albany 12205
Tel: (518) 458-1550
TWX: 710-444-4961

650 Perinton Hill Office Park
Fairport 14450
Tel: (716) 223-9950
TWX: 510-253-0092

No.1 Pennsylvania Plaza
55th floor
34th street & 8th Avenue
New York 10001
Tel: (212) 971-0800

5858 East Molloy Road
Syracuse 13211
Tel: (315) 455-2468

1 Crossways Park West
Woodbury 11797
Tel. (516) 921-0300

NORTH CAROLINA
5805 Roanne Way
Greensboro 27405
Tel: (919) 852-1800

OHIO
Medical/Computer Only
Bldg. 300
1313 E. Kemper Rd.
Cincinnati 45426
Tel: (513) 871-7400

16500 Sprague Road
Cleveland 44130
Tel: (216) 243-7300
TWX: 810-423-9430

330 Progress Rd.
Dayton 45449
Tel: (513) 859-8202

1041 Kingsmill Parkway
Columbus 43229
Tel: (614) 436-1041

OKLAHOMA
P O. Box 32008
6301 N. Meridan Avenue
Oklahoma City 73112
Tel: (405) 721-0200

9920 E. 42nd Street
Suite 121
Tulsa 74145

OREGON
17890 SW Lower Boones Ferry road
Tualatin 97062
Tel: (503) 620-3350

PENNSYLVANIA
111 Zeta Drive
Pittsburgh 15238
Tel: (412) 782-0400

1021 8th Avenue
King of Prussia Industrial Park
King of Prussia 19406
Tel: (215) 265-7000
TWX: 510-660-2670

PUERTO RICO
Hewlett-Packard Inter-Americas
Puerto Rico Branch Office
Calle 272,
Edif. 203 Urg. Country Club
Carolina 00924
Tel: (809) 762-7255
Telex: 345 0514

SOUTH CAROLINA
P. O. Box 6442
6941-0 N. Trenholm Road
Columbia 29260
Tel: (803) 782-6493

TENNESSEE
8914 Kingston Pike
Knoxville 37922
Tel: (615) 523-0522

3027 Vanguard Dr.
Director's Plaza
Memphis 38131
Tel: (901) 346-8370

*Nashville
Medical Service only
Tel: (615) 244-5448

TEXAS
4171 North Mesa
Suite C110
El Paso 79902
Tel: (915) 533-3555

P.O. Box 1270
201 E. Arapaho Rd.
Richardson 75080
Tel: (214) 231-6101

P.O. Box 42816
10535 Harwin Dr.
Houston 77036
Tel: (713) 776-6400

*Lubbock
Medical Service only
Tel: (806) 799-4472

205 Billy Mitchell Road
San Antonio 78226
Tel: (512) 434-8241

UTAH
2160 South 3270 West Street
Salt Lake City 84119
Tel: (801) 972-4711

VIRGINIA
P.O. Box 12778
Norfolk 23502
Tel: (804) 460-2871

P.O. Box 9669
2914 Hungary Springs Road
Richmond 23228
Tel: (804) 285-3431

WASHINGTON*
Bellefield Office Pk.
1203-114th Ave. S.E.
Bellevue 98004
Tel: (206) 454-3971
TWX: 910-443-2446

P.O. Box 4010
Spokane 99202
Tel: (509) 535-0864

***WEST VIRGINIA**
Medical/Analytical Only
Charleston
Tel: (304) 345-1640

WISCONSIN
9004 West Lincoln Ave.
West Allis 53227
Tel: (414) 541-0550

FOR U.S. AREAS NOT LISTED:
Contact the regional office nearest you: Atlanta, Georgia ... North Hollywood, California ... Rockville, Maryland ... Rolling Meadows, Illinois. Their complete addresses are listed above.

*Service Only

[illegible] 09845-910[illegible]
Microfiche No. 09845-[illegible]

Printed in U.S.A.
May 1, 1979