Author(s) : H. Renshall **Library:** KERNLIB
Submitter : **Submitted:** 27.11.1984
Language : Fortran **Revised:**

Sort One-Dimensional Character Array into Itself

SORCHA does a slow linear sort of a type CHARACTER array into itself in either ascending or descending order. The sort is done on any user specified substring of the elements in a CHARACTER array.

Structure:

SUBROUTINE subprogram
User Entry Names: SORCHA

Usage:

```
CALL SORCHA(A,ICH1,ICH2,NPOINT,ITYPE)
```

A (CHARACTER) One-dimensional array of dimension NPOINT to be sorted into itself. The maximum length of the elements in A is 256 characters.

ICH1 (INTEGER) Variable or constant giving the first character position in each element of A of the substring upon which the array shall be sorted. ICH1 should be 1 if the whole length of the elements of A is to be used.

ICH2 (INTEGER) Variable or constant giving the last character position analogously to ICH1 above. ICH2 should be equal to the length of the elements of A if the sort should be on the entire length of the elements of A.

NPOINT (INTEGER) Variable or constant. The first NPOINT elements of A will be sorted.

ITYPE (INTEGER) Variable or constant controlling the type of the sort. It is possible to sort in ascending or descending order; in addition it is possible to use either the Fortran collation sequence ordering via the LLE and LGE functions, or the machine internal relational sequence ordering via the LE and GE relations (see **Notes**).
= 1 : Ascending sort, i.e. A(1) will be lower than A(2), using collation sequence.
= 2 : Descending sort, i.e. A(2) will be lower than A(1), using collation sequence.
= 3 : Ascending sort, i.e. A(1) will be lower than A(2), using relational sequence.
= 4 : Descending sort, i.e. A(2) will be lower than A(1), using relational sequence.

Notes:

On the machines and compilers tested (CDC with FTN5, VAX VMS with Fortran, ND500 with FORT-5, IBM with VS-Fortran and Siemens compilers) the collating sequence orders are the same and give blank less than numbers and numbers less than letters (this matches the ASCII internal representations).

On IBM with both compilers the relational sorts give blank less than letters and letters less than numbers (the EBCDIC sequence).

On CDC, VAX and ND500 collation and relational orders are the same.

On all machines the relational sort is faster than the collation sequence sort.

•