18.2 Sorting Data of Arbitrary Structure in Memory

A. Purpose
Sort data having an organization or structure not supported by one of the subprograms in Chapter 18.1, for example, data having more than one key to determine the sorted order. The subprogram INSORT in Chapter 18.3 has similar functionality to GSORTP and is more efficient if the data are initially partly ordered, or the ordering criterion is expensive.

B. Usage

B.1 Program Prototype

INTEGER N, IP(\geq |N|), COMPAR
EXTERNAL COMPAR

Assign values to N and data elements indexed by 1 through N. Require N \geq 1.

CALL GSORTP (COMPAR, N, IP)

Following the call to GSORTP the contents of IP(1) through IP(N) are such that the J\textsuperscript{th} element of the sorted sequence is the IP(J)\textsuperscript{th} element of the original sequence.

B.2 Argument Definitions

COMPAR [in] An INTEGER FUNCTION subprogram that defines the relative order of elements of the data. COMPAR is invoked as COMPAR(I, J), and is expected to return \(-1\) (or any negative integer) if the I\textsuperscript{th} element of the original data is to precede the J\textsuperscript{th} element in the sorted sequence, \(+1\) (or any positive integer) if the I\textsuperscript{th} element is to follow the J\textsuperscript{th} element, and zero if the order is immaterial. GSORTP does not have access to the data. It is the caller’s responsibility to make the data known to COMPAR. Since COMPAR is a dummy procedure, it may have any name. Its name must appear in an EXTERNAL statement in the calling program unit.

N [in] |N| is the number of elements to sort, and the upper bound of subscripts to use to access IP. If N > 0 then IP(I) is initialized to I, for 1 \leq I \leq N. Actual arguments for COMPAR are always elements of IP.

IP() [out] An array to contain the definition of the sorted sequence. IP(1:N) are set so the J\textsuperscript{th} element of the sorted sequence is the IP(J)\textsuperscript{th} element of the original sequence.

C. Examples and Remarks
Program DRGSORTP illustrates the use of GSORTP to sort 1000 randomly generated real numbers. The output should consist of the single line

GSORTP succeeded

Stability
A sorting method is said to be stable if the original relative order of equal elements is preserved. This subroutine uses the quicksort algorithm, which is not inherently stable. To impose stability, return COMPAR = I – J if the I\textsuperscript{th} and J\textsuperscript{th} elements are equal.

D. Functional Description
See Section 18.1.D.

E. Error Procedures and Restrictions
See Section 18.1.E.

F. Supporting Information
The source language for these subroutines is ANSI Fortran 77.

Entry Required Files
GSORTP GSORTP

Designed and coded by W. V. Snyder, JPL 1988.
logical OK
integer I, COMPAR, P(1:1000)
external COMPAR
real R(1:1000)
common /R.COM/ R

Generate 1000 random numbers
call sranua(r, 1000)

Sort them using GSORTP.
ok=TRUE
call gsortp(compar, 1000, p)

Check the order.
do 10 i = 2, 1000
   if (r(p(i)).lt.r(p(i-1))) ok=FALSE.
10 continue
Print the results.
if(ok)then
   print *, 'GSORTP succeeded'
else
   print *, 'GSORTP failed'
end if

integer function COMPAR(I,J)

Determine the relative order of R(I) and R(J), where R is in
the common block /R.COM/. Return -1 if R(I) should precede R(J)
in the sorted order, +1 if R(I) should follow R(J), and 0
otherwise.

integer I,J
real R(1:1000)
common /R.COM/ R
if (R(I)-R(J)) 10,20,30
10 compar=-1
return
20 compar=0
return
30 compar=+1
return

end