Utility: nagdmc_index

Purpose

nagdmc_index converts an array of ranks to the corresponding index array.

Declaration

```
#include <nagdmc.h>
void nagdmc_index(long n, long rank[], int *info);
```

Parameters

1: **n** – `long` *Input*

On entry: the number of rank values.

Constraint: $\mathbf{n} > 0$.

2: **rank**[**n**] – `long` *Input/Output*

On entry: an array of **n** ranks.

Constraints: $0 \leq \mathbf{rank}[i] < \mathbf{n}$, for $i = 0, 1, \ldots, \mathbf{n} - 1$.

On exit: the indexes corresponding to the ranks.

3: **info** – `int *` *Output*

On exit: **info** gives information on the success of the function call:

0: the function successfully completed its task.

i; $i = 1, 2$: the specification of the ith formal parameter was incorrect.

100: an internal error occurred during the execution of the function.

Notation

None.

Description

There are two common ways of describing a permutation using an integer vector. The first uses ranks in which case the ith element holds the position to which the ith data element should be moved in order to sort the data; in other words its rank in the sorted order. The second uses indices in which case the ith element holds the current position of the data element which would occur in the ith position in sorted order. For example, given the values:

3.5 5.9 2.9 0.5,

to be sorted in ascending order, the ranks would be:

3 4 2 1,

and the indices would be:

4 3 1 2.

NAG DMC routines generate ranks and these can be used to sort data values in ascending order. However, if it is desired simply to refer to the data in sorted order without actually re-ordering them, indices are more convenient than ranks.

nagdmc_index can be used to convert ranks to indices, or indices to ranks, as the two permutations are inverses of each another.

References and Further Reading

None.

See Also

nagdmc_rank_long	computes ranks for integer-valued data.
nagdmc_rank_real	computes ranks for real-valued data.
