

## nag\_ran\_permut\_vec (g05ehc)

### 1. Purpose

`nag_ran_permut_vec (g05ehc)` performs a pseudo-random permutation of a vector of integers.

### 2. Specification

```
#include <nag.h>
#include <nagg05.h>
```

```
void nag_ran_permut_vec(Integer index[], Integer n, NagError *fail)
```

### 3. Description

The function generates a single pseudo-random permutation of the elements of **index** without inspecting their values. Each of the  $n!$  possible permutations of the  $n$  values may be regarded as being equiprobable.

### 4. Parameters

**index[n]**

Input: the  $n$  integer values to be permuted.

Output: the  $n$  permuted integer values.

**n**

Input: the number of values to be permuted.

Constraint:  $n \geq 1$ .

**fail**

The NAG error parameter, see the Essential Introduction to the NAG C Library.

### 5. Error Indications and Warnings

**NE\_INT\_ARG\_LT**

On entry, **n** must not be less than 1: **n** = *value*.

### 6. Further Comments

It should be noted that if  $n$  is 20 or more it is theoretically impossible to generate all  $n!$  permutations as  $n!$  exceeds the cycle length of the internal random number generator.

The time taken by the function is of order  $n$ .

In order to permute other kinds of objects (i.e., vectors, or matrices of higher dimensions), the following technique may be used:

(a) Set **index**[ $i - 1$ ] =  $i$ , for  $i = 1, 2, \dots, n$  (where  $n$  is the number of objects)

(b) Use `nag_ran_permut_vec` to permute **index**

(c) Use the contents of **index** as a set of indices to access the relevant object.

In order to divide pseudo-randomly an array of  $n$  objects (`obj_array[n]`) into  $k$  subgroups of chosen sizes  $S_1, S_2 \dots S_k$  a similar procedure may be used. For the first  $S_1$ , elements of **index** set **index**[ $i$ ] = 1,  $i = 0 \dots S_1 - 1$ , for the next  $S_2$  elements of **index** set **index**[ $S_1 + i$ ] = 2,  $i = 0 \dots S_2 - 1$ , for size  $S_j$  set **index**[ $S_1 + S_2 + \dots + S_{j-1} + i$ ] =  $j$ ,  $i = 0 \dots S_j - 1$  etc. Permute **index** using `nag_ran_permut_vec` and then, if **index**[ $i$ ] =  $j$ , `obj_array[i]` is to be included in the  $j$ th subgroup.

#### 6.1. Accuracy

Not applicable.

#### 6.2. References

Kendall M G and Stuart A (1969) *The Advanced Theory of Statistics (Vol 2)*. (3rd Edn) Griffin, London.

Knuth D E (1981) *The Art of Computer Programming (Vol 2)*. (2nd Edn) Addison-Wesley.

**7. See Also**

nag\_ran\_sample\_vec (g05ejc)

**8. Example**

A vector containing 0 and the first 7 positive integers in ascending order is permuted and the permutation is printed. This is repeated a total of 10 times.

**8.1. Program Text**

```

/* nag_ran_permut_vec(g05ehc) Example Program
 *
 * Copyright 1994 Numerical Algorithms Group.
 *
 * Mark 3, 1994.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg05.h>

#define NMAX 8

main()
{
    Integer j, k, n, m;
    Integer index[NMAX];
    Integer seed = 0;

    Vprintf("g05ehc Example Program Results\n");
    g05cbc(seed);
    n = NMAX;
    m = 10;

    Vprintf("\n%ld Permutations of the first %ld integers \n\n", m, n);
    for (j = 0; j < m; ++j)
    {
        /* construct index vector to be permuted */
        for (k = 0; k < n; ++k)
            index[k] = k;
        g05ehc(index, n, NAGERR_DEFAULT);
        for (k = 0; k < n; ++k)
            Vprintf("%ld ", index[k]);
        Vprintf("\n");
    }
    exit(EXIT_SUCCESS);
}

```

**8.2. Program Data**

None.

**8.3. Program Results**

g05ehc Example Program Results

10 Permutations of the first 8 integers

```

6 7 0 1 3 5 2 4
2 0 3 5 6 7 4 1
6 5 4 0 2 3 7 1
5 1 6 2 7 4 0 3
0 5 1 3 6 7 4 2
3 0 4 7 6 5 2 1
7 2 0 5 3 1 4 6
0 1 5 6 7 3 2 4
1 4 6 5 2 0 3 7
1 7 5 6 2 4 0 3

```