NAG Library Routine Document

M01ZBF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

M01ZBF checks the validity of a permutation.

2 Specification

```
SUBROUTINE M01ZBF (IPERM, M1, M2, IFAIL)
INTEGER IPERM(M2), M1, M2, IFAIL
```

3 Description

M01ZBF can be used to check the validity of user-supplied ranks or indices, without the ranks or indices being corrupted.

4 References

None.

5 Arguments

1: IPERM(M2) – INTEGER array *Input/Output*

On entry: elements M1 to M2 of IPERM must be set to values which are supposed to be a permutation of the integers M1 to M2. If they are a valid permutation, the routine exits with IFAIL $= 0$.

On exit: used as internal workpsace prior to being restored and hence is unchanged.

2: M1 – INTEGER *Input*

3: M2 – INTEGER *Input*

On entry: the range of elements used in the array IPERM and the range of values in the permutation, as specified under IPERM.

Constraint: $0 < \text{M1} \leq \text{M2}$.

4: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL $= 1$

On entry, M2 $<$ 1,
or M1 $<$ 1,
or M1 $>$ M2.

IFAIL $= 2$

Elements M1 to M2 of IPERM contain a value outside the range M1 to M2.

IFAIL $= 3$

Elements M1 to M2 of IPERM contain a repeated value.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

If IFAIL $= 2$ or 3, elements M1 to M2 of IPERM do not contain a permutation of the integers M1 to M2.

7 Accuracy

Not applicable.

8 Parallelism and Performance

M01ZBF is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example reads in a vector of real numbers, and a vector of ranks; it calls M01ZBF to check the validity of the ranks before calling M01EAF to rearrange the real numbers into the specified order.

10.1 Program Text

```
    Program m01zbfe

!     M01ZBF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: m01eaf, m01zbf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, m1, m2
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: rv(:)
      Integer, Allocatable             :: iperm(:)
!     .. Executable Statements ..
      Write (nout,*) 'M01ZBF Example Program Results'

!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) m2
      Allocate (rv(m2),iperm(m2))

      m1 = 1

      Read (nin,*)(rv(i),i=m1,m2)
      Read (nin,*)(iperm(i),i=m1,m2)

      ifail = 0
      Call m01zbf(iperm,m1,m2,ifail)

      ifail = 0
      Call m01eaf(rv,m1,m2,iperm,ifail)

      Write (nout,*)
      Write (nout,*) 'Numbers in rank order'
      Write (nout,*)
      Write (nout,99999)(rv(i),i=m1,m2)

99999 Format (1X,10F7.1)
    End Program m01zbfe
```

10.2 Program Data

```
M01ZBF Example Program Data
12
5.3 4.6 7.8 1.7 5.3 9.9 3.2 4.3 7.8 4.5 1.2 7.6
  7   6  10   2   8  12   3   4  11   5   1   9
```

10.3 Program Results

```
 M01ZBF Example Program Results

 Numbers in rank order

     1.2    1.7    3.2    4.3    4.5    4.6    5.3    5.3    7.6    7.8
     7.8    9.9
```