NAG Toolbox

nag_sort_charvec_search (m01nc)

1 Purpose

nag_sort_charvec_search (m01nc) examines an ordered vector of null terminated strings and returns the index of the first value equal to the sought-after item. Character items are compared according to the ASCII collating sequence.

2 Syntax

```
[result, ifail] = nag_sort_charvec_search(valid, ch, item, 'm1', m1, 'm2', m2)
[result, ifail] = m01nc(valid, ch, item, 'm1', m1, 'm2', m2)
```

3 Description

nag_sort_charvec_search (m01nc) is based on Professor Niklaus Wirth's implementation of the Binary Search algorithm (see Wirth (2004)), but with two modifications. First, if the sought-after item is less than the value of the first element of the array to be searched, 0 is returned. Second, if a value equal to the sought-after item is not found, the index of the immediate lower value is returned.

4 References

Wirth N (2004) *Algorithms and Data Structures* 35–36 Prentice Hall

5 Parameters

5.1 Compulsory Input Parameters

1: **valid** – LOGICAL

If **valid** is set to *true* argument checking will be performed. If **valid** is set to *false* nag_sort_charvec_search (m01nc) will be called without argument checking, which includes checking that array **ch** is sorted in ascending order and the function will return with **ifail** $= 0$. See Section 9 for further details.

2: **ch**(**m2**) – CHARACTER(*) array

Elements **m1** to **m2** contain character data to be searched.

Constraint: elements **m1** to **m2** of **ch** must be sorted in ascending order. The length of each element of **ch** must not exceed 255. Trailing space characters are ignored.

3: **item** – CHARACTER(*)

The sought-after item. Trailing space characters are ignored.

5.2 Optional Input Parameters

1: **m1** – INTEGER

Default: 1

The index of the first element of **ch** to be searched.

Constraint: **m1** ≥ 1.

2: **m2** – INTEGER

Default: the dimension of the array **ch**.

The index of the last element of **ch** to be searched.

Constraint: $\mathbf{m2} \geq \mathbf{m1}$.

5.3 Output Parameters

1: **result**

The result of the function.

2: **ifail** – INTEGER

$\mathbf{ifail} = 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

(**Note:** these errors will only be returned if $\mathbf{valid} = \mathit{true}$.)

$\mathbf{ifail} = 2$

On entry, **ch** must be sorted in ascending order.

$\mathbf{ifail} = 3$

Constraint: $\mathbf{m1} \geq 1$.

$\mathbf{ifail} = 4$

Constraint: $\mathbf{m2} \geq \mathbf{m1}$.

$\mathbf{ifail} = 5$

On entry, the length of each element of **ch** must be at most 255.

$\mathbf{ifail} = -99$

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

$\mathbf{ifail} = -399$

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

$\mathbf{ifail} = -999$

Dynamic memory allocation failed.

7 Accuracy

Not applicable.

8 Further Comments

The argument **valid** should be used with caution. Set it to *false* only if you are confident that the other arguments are correct, in particular that array **ch** is in fact arranged in ascending order. If you wish to search the same array **ch** many times, you are recommended to set **valid** to *true* on first call of nag_sort_charvec_search (m01nc) and to *false* on subsequent calls, in order to minimize the amount of time spent checking **ch**, which may be significant if **ch** is large.

The time taken by nag_sort_charvec_search (m01nc) is $O(\log(n))$, where $n = \mathbf{m2} - \mathbf{m1} + 1$, when **valid** $= false$.

9 Example

This example reads a list of character data and sought-after items and performs the search for these items.

9.1 Program Text

```
function m01nc_example

fprintf('m01nc example results\n\n');

ch    = {'black  '; 'blue   '; 'crimson'; 'cyan   '; 'green  ';
         'orange '; 'pink   '; 'purple '; 'red    '; 'white  '};
item  = {'blond  '; 'amber  '; 'plum   '; 'yellow '};

m1 = nag_int([ 1    1    1    5]);
m2 = nag_int([ 7   10   10   10]);

validate = true;
for j = 1:4
  [result, ifail] = m01nc( ...
                           validate, ch, item{j}, 'm1', m1(j), 'm2', m2(j));
  validate = false;

  fprintf('Search for %s in index range [%2d:%2d]: index = %2d\n', ...
          item{j}, m1(j), m2(j), result);
end
```

9.2 Program Results

```
m01nc example results

Search for blond   in index range [ 1: 7]: index =  1
Search for amber   in index range [ 1:10]: index =  0
Search for plum    in index range [ 1:10]: index =  7
Search for yellow  in index range [ 5:10]: index = 10
```