NAG Library Function Document

nag_enum_name_to_value (x04nac)

1 Purpose

nag_enum_name_to_value (x04nac) returns a value corresponding to the name of a NAG enumeration member.

2 Specification

```
#include <nag.h>
#include <nagx04.h>
int nag_enum_name_to_value (const char *enum_string)
```

3 Description

nag_enum_name_to_value (x04nac) takes a string argument, which must be the name of a NAG enumeration member (e.g., `"Nag_ColMajor"`, `"Nag_LogNormal""`, etc.), and returns, as an `int`, the corresponding NAG enumeration member value. If the input string is not the name of a valid NAG enumeration member then the value -1 is returned.

The reverse process of converting from enumeration member value to enumeration member name is also available using nag_enum_value_to_name (x04nbc).

Converting enumeration members to and from name and value may be of use when saving a set of problem arguments to file or reading problem arguments from a file for use in an application. In the case of saving problem arguments, any enumeration members to be saved should be saved using their names to be subsequently read as strings. When these strings are read back by an application, nag_enum_name_to_value (x04nac) can be used to convert the strings to values suitable for inclusion in NAG C Library function calls.

nag_enum_name_to_value (x04nac) is also useful when calling NAG C Library functions from external application interfaces that do not facilitate enumeration types. In such cases, NAG function calls that expect an enumeration member name as one of its input arguments can pass instead its corresponding integer value following a call to nag_enum_name_to_value (x04nac).

4 References

None.

5 Arguments

1: **enum_string** – const char * *Input*

On entry: the name of a NAG enumeration member, e.g., `"Nag_GeneralMatrix"`.

6 Error Indicators and Warnings

If the value -1 is returned then the input string is not recognized as a valid NAG enumeration member name.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_enum_name_to_value (x04nac) is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example reads problem arguments from input, converts enumeration member names to their correponding value using nag_enum_name_to_value (x04nac), and passes these values to a function for solving a system of equations.

10.1 Program Text

```
/* nag_enum_name_to_value (x04nac) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf07.h>
#include <nagx04.h>

int main(void)
{
  /* Scalars */
  Integer i, j, n, nrhs, pda, pdb;
  Integer exit_status = 0;
  NagError fail;
  Nag_OrderType order;
  Nag_TransType trans;
  Nag_MatrixType matrix;
  Nag_DiagType unitdiag;

  /* Arrays */
  double *a = 0, *b = 0;
  Integer *ipiv = 0;
  char nag_enum_arg[20];
#ifdef NAG_LOAD_FP
  /* The following line is needed to force the Microsoft linker
     to load floating point support */
  float force_loading_of_ms_float_support = 0;
#endif /* NAG_LOAD_FP */

  INIT_FAIL(fail);

  printf("nag_enum_name_to_value (x04nac) Example Program Results\n\n");

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  /* Read the problem dimensions. */
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%" NAG_IFMT "%*[^\n] ", &n, &nrhs);
#else
```

```
  scanf("%" NAG_IFMT "%" NAG_IFMT "%*[^\n] ", &n, &nrhs);
#endif

  /* Read the storage order of the matrices and convert to value. */
#ifdef _WIN32
  scanf_s("%19s%*[^\n] ", nag_enum_arg, (unsigned)_countof(nag_enum_arg));
#else
  scanf("%19s%*[^\n] ", nag_enum_arg);
#endif

  /* nag_enum_name_to_value (x04nac).
   * Converts NAG enum member name to value
   */
  order = (Nag_OrderType) nag_enum_name_to_value(nag_enum_arg);

  /* Read whether matrix A is to be transposed and convert. */
#ifdef _WIN32
  scanf_s("%19s%*[^\n] ", nag_enum_arg, (unsigned)_countof(nag_enum_arg));
#else
  scanf("%19s%*[^\n] ", nag_enum_arg);
#endif
  trans = (Nag_TransType) nag_enum_name_to_value(nag_enum_arg);

  /* Read and convert parameters for writing solution using
   * nag_gen_real_mat_print (x04cac). */
#ifdef _WIN32
  scanf_s("%19s%*[^\n] ", nag_enum_arg, (unsigned)_countof(nag_enum_arg));
#else
  scanf("%19s%*[^\n] ", nag_enum_arg);
#endif
  matrix = (Nag_MatrixType) nag_enum_name_to_value(nag_enum_arg);

#ifdef _WIN32
  scanf_s("%19s%*[^\n] ", nag_enum_arg, (unsigned)_countof(nag_enum_arg));
#else
  scanf("%19s%*[^\n] ", nag_enum_arg);
#endif
  unitdiag = (Nag_DiagType) nag_enum_name_to_value(nag_enum_arg);

  /* Set up defines for reading matrices using given order. */
#define A(I, J) a[(J-1)*pda + I - 1]
#define B(I, J) b[(J-1)*pdb + I - 1]
  if (order == (int) Nag_ColMajor) {
    pda = n;
    pdb = n;
  }
  else {
    pda = n;
    pdb = nrhs;
  }

  /* Allocate memory */
  if (!(a = NAG_ALLOC(n * n, double)) ||
      !(b = NAG_ALLOC(n * nrhs, double)) || !(ipiv = NAG_ALLOC(n, Integer)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

  /* Read A and B from data file */
  if (order == Nag_ColMajor) {
    for (i = 1; i <= n; ++i) {
      for (j = 1; j <= n; ++j)
#ifdef _WIN32
        scanf_s("%lf", &A(i, j));
#else
        scanf("%lf", &A(i, j));
#endif
    }
#ifdef _WIN32
```

```
    scanf_s("%*[^\n] ");
#else
    scanf("%*[^\n] ");
#endif
    for (i = 1; i <= n; ++i) {
      for (j = 1; j <= nrhs; ++j)
#ifdef _WIN32
        scanf_s("%lf", &B(i, j));
#else
        scanf("%lf", &B(i, j));
#endif
    }
#ifdef _WIN32
    scanf_s("%*[^\n] ");
#else
    scanf("%*[^\n] ");
#endif
  }
  else {
    for (i = 1; i <= n; ++i) {
      for (j = 1; j <= n; ++j)
#ifdef _WIN32
        scanf_s("%lf", &A(j, i));
#else
        scanf("%lf", &A(j, i));
#endif
    }
#ifdef _WIN32
    scanf_s("%*[^\n] ");
#else
    scanf("%*[^\n] ");
#endif
    for (i = 1; i <= n; ++i) {
      for (j = 1; j <= nrhs; ++j)
#ifdef _WIN32
        scanf_s("%lf", &B(j, i));
#else
        scanf("%lf", &B(j, i));
#endif
    }
#ifdef _WIN32
    scanf_s("%*[^\n] ");
#else
    scanf("%*[^\n] ");
#endif
  }

  /* Factorize A */
  /* nag_dgetrf (f07adc).
   * LU factorization of real m by n matrix
   */
  nag_dgetrf(order, n, n, a, pda, ipiv, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_dgetrf (f07adc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  /* Compute solution */
  /* nag_dgetrs (f07aec).
   * Solution of real system of linear equations, multiple
   * right-hand sides, matrix already factorized by nag_dgetrf
   * (f07adc)
   */
  nag_dgetrs(order, trans, n, nrhs, a, pda, ipiv, b, pdb, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_dgetrs (f07aec).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }
```

```
  /* nag_enum_value_to_name (x04nbc).
   * Converts NAG enum member value to its name
   */
  printf("Array storage scheme used in nag_dgetrs (f07aec) is %s\n",
         nag_enum_value_to_name(order));

  /* nag_error_name_to_code (x04ncc).
   * Converts NAG error name to its code value
   */
  printf("nag_dgetrs (f07aec) returns with the error code "
         "(fail.code) set to %d\n\n", nag_error_name_to_code("NE_NOERROR"));

  /* Print solution */
  /* nag_gen_real_mat_print (x04cac).
   * Print real general matrix (easy-to-use)
   */
  fflush(stdout);
  nag_gen_real_mat_print(order, matrix, unitdiag, n, nrhs, b, pdb,
                         "Solution(s)", 0, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_gen_real_mat_print (x04cac).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }
END:
  NAG_FREE(a);
  NAG_FREE(b);
  NAG_FREE(ipiv);
  return exit_status;
}
```

10.2 Program Data

```
nag_enum_name_to_value (x04nac) Example Program Data
  4  2                         :Values of N and NRHS
Nag_ColMajor
Nag_NoTrans
Nag_GeneralMatrix
Nag_NonUnitDiag
  1.80   2.88   2.05  -0.89
  5.25  -2.95  -0.95  -3.80
  1.58  -2.69  -2.90  -1.04
 -1.11  -0.66  -0.59   0.80   :End of matrix A
  9.52  18.47
 24.35   2.25
  0.77 -13.28
 -6.22  -6.21                 :End of matrix B
```

10.3 Program Results

```
nag_enum_name_to_value (x04nac) Example Program Results

Array storage scheme used in nag_dgetrs (f07aec) is Nag_ColMajor
nag_dgetrs (f07aec) returns with the error code (fail.code) set to 0

 Solution(s)
            1          2
 1      1.0000     3.0000
 2     -1.0000     2.0000
 3      3.0000     4.0000
 4     -5.0000     1.0000
```