NAG Library Function Document

nag_error_name_to_code (x04ncc)

1 Purpose

nag_error_name_to_code (x04ncc) takes, as a string argument, the name of a NAG error (e.g., **NE_BAD_PARAM**) and returns, as an `int`, the corresponding **fail** code for the given NAG error.

2 Specification

```
#include <nag.h>
#include <nagx04.h>
int nag_error_name_to_code (const char *error_name)
```

3 Description

nag_error_name_to_code (x04ncc) takes, as a string argument, the name of a NAG error (e.g., **NE_BAD_PARAM**) and returns, as an `int`, the corresponding **fail** code for the given NAG error. If the input string does not correspond to a NAG error name then the function returns -1.

Converting the error name to `int` may be useful when the NAG Header Files are not available, but the `int` value of an error name is required, say, for checking **fail.code** following a call to a NAG function.

4 References

None.

5 Arguments

1: **error_name** – const char * *Input*

On entry: the value of a NAG enumeration member.

6 Error Indicators and Warnings

If the value -1 is returned then the input value is not recognized as a valid NAG fail code.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_error_name_to_code (x04ncc) is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example produces integer error codes corresponding to a number of error code names.

10.1 Program Text

```
/* nag_error_name_to_code (x04ncc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

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

int main(void)
{
  /* Scalars */
  Integer exit_status = 0;
  int ierror;
  /* Pointers */
  const char *str_error;

  printf("nag_error_name_to_code (x04ncc) Example Program Results\n\n");

  /* Print header. */
  printf("   Error name     Value Returned Value\n");
  printf("----------------- ----- --------------\n");

  /* Convert the error strings to values and print. */

  str_error = "NE_BAD_PARAM";
  /* nag_error_name_to_code (x04ncc).
   * Converts NAG error name to its code value
   */
  ierror = nag_error_name_to_code(str_error);
  printf("%-17s %5d %12d\n", str_error, NE_BAD_PARAM, ierror);

  str_error = "NE_INCOMPAT_PARAM";
  ierror = nag_error_name_to_code(str_error);
  printf("%-17s %5d %12d\n", str_error, NE_INCOMPAT_PARAM, ierror);

  str_error = "NE_COMPLEX_ZERO";
  ierror = nag_error_name_to_code(str_error);
  printf("%-17s %5d %12d\n", str_error, NE_COMPLEX_ZERO, ierror);

  str_error = "NE_NOT_MONOTONIC";
  ierror = nag_error_name_to_code(str_error);
  printf("%-17s %5d %12d\n", str_error, NE_NOT_MONOTONIC, ierror);

  str_error = "NE_TOO_MANY_ITER";
  ierror = nag_error_name_to_code(str_error);
  printf("%-17s %5d %12d\n", str_error, NE_TOO_MANY_ITER, ierror);

  str_error = "NE_SINGULAR";
  ierror = nag_error_name_to_code(str_error);
  printf("%-17s %5d %12d\n", str_error, NE_SINGULAR, ierror);

  str_error = "NE_INITIALIZATION";
  ierror = nag_error_name_to_code(str_error);
  printf("%-17s %5d %12d\n", str_error, NE_INITIALIZATION, ierror);

  return exit_status;
}
```

10.2 Program Data

None.

10.3 Program Results

```
nag_error_name_to_code (x04ncc) Example Program Results

   Error name     Value Returned Value
----------------- ----- --------------
NE_BAD_PARAM         70             70
NE_INCOMPAT_PARAM    88             88
NE_COMPLEX_ZERO     133            133
NE_NOT_MONOTONIC    245            245
NE_TOO_MANY_ITER    357            357
NE_SINGULAR         366            366
NE_INITIALIZATION  2049           2049
```