19.2 Error Message Processor

A. Purpose

This package of subroutines processes diagnostic messages, providing options to write or not write the message, and to return or stop after message processing. These subroutines are intended primarily for use by other library subprograms although they are not restricted to that usage.

By passing error messages through these subroutines it is possible to (a) avoid having Fortran I/O statements in other library subprograms that do not otherwise execute any I/O functions, and (b) provide a means, via the subroutine ERMSET, for a user program to alter the nominal action of error message processing.

B. Usage

To process an error message the program detecting the error must make a sequence of one or more calls to subroutines of this package. Such a sequence of calls is distinguished by the condition that the argument, FLAG, has the value $'$,$'$ in all but the last call, and has the value $'.'$ in the last call of the sequence.

The first, and possibly only, call of a sequence must be to ERMSG, SERM1, DERM1, or IERM1. These subroutines all have the argument, LEVEL, that specifies the nominal action level for the entire sequence of calls.

If there are additional calls in the sequence, the subsequent calls must be to SERV1, DERV1, IERV1, or ERMOR, each of which may be called any number of times. These calls provide additional data values or character strings to be included in the printed error message.

The package contains two additional subroutines: ERMSET, which can be called by a user to alter the nominal action of the package, and ERFIN which is called by other subroutines of the package when FLAG $= '.'$ to handle the common final steps of processing an error message.

B.1 Type Statements for Arguments

INTEGER IERR, LEVEL, IDATA, IDELTA, NVAL

CHARACTER SUBNAM*n_1, MESS*n_2, FLAG, LABEL*n_3

CHARACTER*n_4 LABELS($\geq$NVAL)

REAL SDATA, SVALS($\geq$NVAL)

DOUBLE PRECISION DDATA, DVALS($\geq$NVAL)

B.2 Call Statements

In each of these calls, all arguments must be assigned values before the call and none of the argument values will be changed by the subroutine. It will usually be most convenient to supply most of the arguments, particularly those of type character, as literals in the call statement.

To initiate an error message:

CALL ERMSG (SUBNAM, IERR, LEVEL, MESS, FLAG)

To initiate a message, including one REAL datum:

CALL SERM1 (SUBNAM, IERR, LEVEL, MESS, LABEL, SDATA, FLAG)

To initiate a message, including one DOUBLE PRECISION datum:

CALL DERM1 (SUBNAM, IERR, LEVEL, MESS, LABEL, DDATA, FLAG)

To initiate a message, including one INTEGER datum:

CALL IERM1 (SUBNAM, IERR, LEVEL, MESS, LABEL, IDATA, FLAG)

To add a REAL datum to the current message:

CALL SERV1 (LABEL, SDATA, FLAG)

To add a DOUBLE PRECISION datum to the current message:

CALL DERV1 (LABEL, DDATA, FLAG)

To add an INTEGER datum to the current message:

CALL IERV1 (LABEL, IDATA, FLAG)

To add an additional character string, MESS, to the current message:

CALL ERMOR (MESS, FLAG)

To alter the nominal message processing action:

CALL ERMSET (IDELTA)

Subroutine called by other subroutines of the package when FLAG = $'.'$ to print the closing line of dollar signs and take the final action of returning or stopping:

CALL ERFIN

B.3 Argument Definitions

SUBNAM [in] Name of subprogram in which error has been detected. Suggest length of name ≤ 12.

IERR [in] Identification number for the error.

LEVEL [in] Should be set to 2, 0, or -2 to specify the nominal action desired.

2 = print and stop
0 = print and return
-2 = return with no printing

This specification applies over a complete sequence of calls to the package as described above. In particular if a stop is to occur, it will not happen until the last call of a sequence, *i.e.* a call with FLAG = $'.'$.

The actual action taken is governed by:

NALPHA = NDELTA + LEVEL,

where NDELTA is a saved local variable in the package. The initial value of NDELTA is zero but it can be changed by use of subroutine ERMSET.

MESS [in] Error message of length ≤ 72.

FLAG [in] A single character, either a comma or a period. A comma means further data or character strings to be included in the error message will be provided by subsequent calls. A period means this is the last call relating to the current error message.

LABEL [in] An identifying label to be printed with the data value given in SDATA, DDATA, or IDATA. Suggest length of label ≤ 35.

SDATA [in] Data item of type REAL to be printed with the error message.

DDATA [in] Data item of type DOUBLE PRECISION to be printed with the error message.

IDATA [in] Data item of type INTEGER to be printed with the error message.

NVAL [in] Number of array elements to be printed from LABELS() and from SVALS() or DVALS().

LABELS() [in] An array of NVAL labels to be printed with the data values given in SVALS() or DVALS(). The array elements should have CHARACTER length ≤ 35.

SVALS() [in] An array of NVAL REAL values to be printed.

DVALS() [in] An array of NVAL DOUBLE PRECISION values to be printed.

IDELTA [in] New value to be assigned to the saved local variable NDELTA. Alters the action of the error processing.

C. Examples and Remarks

Usage is illustrated by the program DRERMSG and the output ODERMSG.

D. Functional Description

D.1 Levels of action

The actual action level, NALPHA, computed as LEVEL + NDELTA, determines the action as follows:

NALPHA ≥ 2	print message and STOP.
NALPHA $= -1$, 0 or 1	print message and RETURN.
NALPHA ≤ -2	RETURN, doing no printing.

D.2 Effects of resetting NDELTA

The saved local variable, NDELTA, initially has the value zero. If it is changed by a call to ERMSET the effect can be interpreted as follows:

NDELTA	Effect
2	Print and stop on all diagnostics that would otherwise be printed.
0	Take the standard action.
-1	Do not stop on any diagnostics. Print as usual.
-2	Never stop. Print only those diagnostics that nominally result in a stop.
-4	Do not print or stop on any diagnostic.

D.3 Form of the error message

The message will begin with a line of 72 dollar signs. The next two lines will be:

Subprogram SUBNAM reports Error No. IERR

The initial message, MESS.

Following may be lines of the following forms:

(1) LABEL = *value*

where *value* is SDATA, DDATA, or IDATA,

(2) *label*1 = *val*1, *label*2 = *val*2, ...

where the *labels* are from LABELS() and the *values* are from SVALS() or DVALS(), or

(3) MESS

transmitted by subroutine ERMOR.

Finally the message will be terminated with another line of 72 dollar signs.

E. Error Procedures and Restrictions

This package shares data via a COMMON block named M77ERR. The user must not use this name for any other COMMON block.

If any of the character string arguments are longer than the suggested maximum lengths, the corresponding printed line of the error message will exceed a length of 72 characters.

The subroutine ERMSET should not be called between the beginning and end of a sequence of calls to the error processing subroutines.

F. Supporting Information

The source language is ANSI Fortran 77. Uses common block M77ERR.

Entry	Required Files
DERM1	DERM1, DERV1, ERFIN, ERMSG
DERV1	DERV1, ERFIN
ERFIN	ERFIN
ERMOR	ERFIN, ERMOR
ERMSET	ERFIN, ERMSG
ERMSG	ERFIN, ERMSG
IERM1	ERFIN, ERMSG, IERM1, IERV1
IERV1	ERFIN, IERV1
SERM1	ERFIN, ERMSG, SERM1, SERV1
SERV1	ERFIN, SERV1

This package is based on the subroutine EMESS, designed by F. T. Krogh, JPL, 1974, and programmed by S. A. Singletary, JPL, 1974. Present version designed and programmed by C. L. Lawson and S. Chiu, JPL, April 1983.

DRERMSG

```
c      program DRERMSG
c>> 1988-11-16 DRERMSG CLL
c
      integer IDELTA
      real SX
      double precision DX
c
      data SX,DX / 1.0E0,2.0D0 /
c
      call ERMSG('AAAA',1,0,'Description of error.',',')
      call SERV1('SX',SX,'.')
      call ERMSG('BBBB',2,0,'Description of 2nd error.','.')
      call SERM1('CCCC',3,0,'Description of 3rd error.','SX',SX,'.')
      call DERM1('DDDD',4,0,'Description of 4th error.','DX',DX,'.')
      do 10 IDELTA = -2, 3
        call ERMSET(IDELTA)
        call IERM1('EEEE',5,0,'Testing ERMSET','IDELTA',IDELTA,'.')
   10 continue
      stop
      end
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
SUBPROGRAM AAAA REPORTS ERROR NO.     1
Description of error.
  SX =     1.0000000
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
SUBPROGRAM BBBB REPORTS ERROR NO.     2
Description of 2nd error.
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
SUBPROGRAM CCCC REPORTS ERROR NO.     3
Description of 3rd error.
  SX =     1.0000000
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
SUBPROGRAM DDDD REPORTS ERROR NO.     4
Description of 4th error.
  DX =     2.0000000000000000
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
SUBPROGRAM EEEE REPORTS ERROR NO.     5
Testing ERMSET
  IDELTA =    -1
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
SUBPROGRAM EEEE REPORTS ERROR NO.     5
Testing ERMSET
  IDELTA =     0
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
SUBPROGRAM EEEE REPORTS ERROR NO.     5
Testing ERMSET
  IDELTA =     1
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
SUBPROGRAM EEEE REPORTS ERROR NO.     5
Testing ERMSET
  IDELTA =     2
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```