포오트란 프로그래밍 語의 変更에 関하여

金栄沢 (서울大 計算統計学科 教授)

1. ANSI X 3J3 Committee에 依하여 1975. 8월에 새로운 표준화된 Fortran이 提議되고 認可됨에 따라 다음과 같은 主要変更이 発生하게 되었다.

2. 変更内容을 項目別로 보면

1. The main program may contain a PROGRAM statement.

   A BLOCK DATA subprogram may have a name.

2. Character data type:

   a. Constants: 'ABC'

   b. Declaration of character variables and arrays:

      CHARACTER*4 A, B*B, C(9, 9)*5, D

   c. Concatenation operator: B // 'Q'

   d. Substrings:

      B(2:5) = C(3, 5)(3:4) // D(2:3)

3. Expressions:

   a. Integer, real, or double precision expressions may appear in subscripts, DO parameters, computed GO TO, etc.

b. Expressions of all types may appear in output lists.

c. Character expressions may be used as format specification

```
PRINT '(13, A1, 13)' I, '=', J * K
READ (5, '(A80)') CHARS
```

d. Mixing of arithmetic data types is permitted

e. The prooessor may evaluate any equivalent expression, except that integer division and integrity of parentheses must be preserved.

4. Implicit types of names beginning with designated letters may be declared by means of IMPLICIT statement. Also character lengths

5. PARAMETER statement:

```
PARAMETER N = 1.7, C = '((('
```

a. The type of name depends upon the constant.

b. A parameter name may appear as a primary in an expression, in a data statement, and as the character length in an IMPLICIT statement.

6. Arrays:

a. Arrays may have as many as 7 dimensions.

b. Explicit lower bounds for arrays (with lower bound default = 1):

```
REAL A(-3:-1, 0:7)
```

c. Adjustable dimensions may be transmitted via COMMON as well as in the argument list.

7. Computed GO TO defaults to following statement if the value of the control expression is out of range.

8. DO loops:

   a. Parameters, which may be integer, real, or double precision, are evaluated and converted to the type of the DO variable.

   b. The increment value may be negative.

   c. A trip count is established initially, and is unaffected by changing entities in parameter expressions. The DO variable must not be changed during execution of the loop.

   d. The minimum trip count is zero.

   e. The DO variable remains defined at completion, with the value it would have had if the loop had been executed one more time.

   f. Examples:

```
           Q = .1
           DO 9 X = .9, Q / 2, -Q
                A = X
9               Q = 2 * Q
```

The loop is executed 9 times; at completion, A = 1, X = 0

```
    N = 1
    DO 9,I = 5, 3
9      N = N - 1
```

The loop is executed 0 times; at completion, N = 1, I = 5.

9. Keyword specification of unit and format in input-output statements; also error and end of file specifiers:

```
END = specifier (statement label)
ERR = specifier (statement lable)
UNIT = specifier (expression)
FMT = specifier (reference to statement label or
                 character entit )
READ (ERR = 99, UNIT = N - 2, END = 98, 1 FMT = F) A
```

10. Stream input-output: the property is specified for a file. The next character read or written is the one following the last one read or written, regardless of record boundaries.

11. Direct-access input-output: the property is specified for a file. An input-output statement on a direct-access file transmits one record:

```
WRITE (9, REC = K * 2) B
```

12. Internal files: a character entity name may be used in place of a file name for an input or output statement. (This implements the features of ENCODE and DECODE).

```
     CHARACTER*5 C
     WRITE (C, 9) I
9    FORMAT (I 5)
```

13. OPEN statement: OPEN (list)

The list must specify a unit number, and may also include any of the following:

```
ERR = label
NAME = file name (e.g., 'ABC/DEF')
STATUS = 'OLD'
         'NEW'
         'SCRATCH'
         'UNKNOWN'
ACCESS = 'SEQUENTIAL'
         'STREAM'
         'DIRECT'
FORM   = 'FORMATTED'
         'UNFORMATTED'
RECL = record length in a direct access file
MAXREC = largest record number in a direct access
         file
```

14. CLOSE Statement:

```
ERR = label
STATUS = 'KEEP'
         'DELETE'
```

15. INQUIRE statement:

UNIT = or FILE =

ERR = label

EXIST = logical

OPENED = logical

NUMBER = integer (number of unit connected to file)

NAMED = logical

NAME = character

ACCESS = character ('SEQUENTIAL' 'STREAM', or 'DIRECT')

FORM = character ('FORMATTED' or 'UNFORMATTED')

RECL = integer

MAXREC = integer

The INQUIRE statement permits programmatic determination of the properties of a file or unit.

16. A FORMAT statement label reference may be set by an ASSIGN statement

17. New edit descriptors:

I w.m	at least m digits (left zero fill)
E w.d Ee	e digits in exponent field
E w.d De	e digits in exponent field
A W	character data
A	character data (field width determined by character entity)
'XXXX'	character constant (output only)
T n	tab to position n

±n X	relative tab of n positions (right or left)
'	terminate input-output if list is exhausted
±S, S	control of optional plus sign
BZ, BN	input blanks converted to zeros or skipped

18. List-directed input-output: "*" in place of format identifier specifies a "default" format determined by the input-output list items, and on input by the form of the data.

```
      READ *, I. A(I)
```

19. Intrinsic functions now include those formerly called intrinsic and basic external functions.

    a. Generic functions. Most of them return a value having the same type as the argument(s). For example,

    ```
    SIN (2.3D2) is double
    MIN (1.7, X) is real
    ```

    b. Type conversion generic functions may have integer, real, double, or complex arguments:

    ```
    INT
    REAL
    DBLE
    CMPLX
    ```

    For example, REAL when applied to integer arguments performs FLOAT operation; for double precision arguments truncates to single precision; and delivers real part of complex arguments.

c. CMPLX and ATAN may have one or two arguments.

20. The INTRINSIC statement allows an intrinsic function to be passed as an actual argument.

21. The EXTERNAL statement identifies and external procedure (not an intrinsic) and allows it to be passed as an actual argument.

22. The ENTRY statement allows alternate entry into a subprogram.

23. The SAVE statement specifies local variables and common blocks (not references to dummy arguments) to be saved between executions of a subroutine.

24. Alternate return:

Calling program:

```
      CALL SUBRTN (A, *14, B, *9)
      ...
    9 CONTINUE
      ...
   14 CONTINUE
      ...
```

Subroutine:

```
    SUBROUTINE SUBRTN (X, *, Y, *)
    N = 3
    RETURN N - 1
    END
```

Control is returned to the statement with label 9 in the calling program. RETURN N would be a normal return if N is less than one or greater than two.

3. 結 論

Fortran에 關하여는 여러가지 意見이 있었으며 새로이 変更되는 Fortran에 対하여도 從前의 Fortran과 比較하여 많은 意見이 있다.