CP/M FORTRAN -80程式語言

中册

第三章
FORTRAN-80
的表示式

FORTRAN的**表示式**（ expression ）是由單一的運算元（ operand ）或是一串以運算子（ operator ）連接的運算元所組成。表示式有兩種型態，它們分別是**算術表示式**（ Arithmetic expression)和**邏輯表示式**（ Logical expression ）。在以下幾節中將詳細地介紹此兩種表示式的運算元、運算子及使用規則。

§3.1　算術表示式

算術表示式是在兩個或兩個以上的運算元（可能全部爲常數、全部爲變數、或常數與變數之結合）之間指定一種計算（ Computation ）。當表示式中使用到兩個或兩個以上的運算元時，每一運算元必須以**算術運算子**分開。一個**算術運算子**是一種符號，它代表著須執行的算術運算。五種算術運算子已在§2.3.4節中詳述，讀者若還不清楚請自行複習。

以下將介紹算術表示式的各種形式和使用規則。

規則1：

第三章 FORTRAN-80 的表示式

FORTRAN的**表示式**（ expression ）是由單一的運算元（ operand ）或是一串以運算子（ operator ）連接的運算元所組成。表示式有兩種型態，它們分別是**算術表示式**（ Arithmetic expression)和**邏輯表示式**（ Logical expression ）。在以下幾節中將詳細地介紹此兩種表示式的運算元、運算子及使用規則。

§3.1 算術表示式

算術表示式是在兩個或兩個以上的運算元（ 可能全部爲常數、全部爲變數、或常數與變數之結合 ）之間指定一種計算（ Computation ）。當表示式中使用到兩個或兩個以上的運算元時，每一運算元必須以**算術運算子**分開。一個**算術運算子**是一種符號，它代表著須執行的算術運算。五種算術運算子已在§2.3.4節中詳述，讀者若還不清楚請自行複習。

以下將介紹算術表示式的各種形式和使用規則。

規則1：

常數、變數名稱、陣列元素或單獨的函數（ Function ）都是一種表示式。例如：

```
S(I)    JOBNO    217    17.26    SQRT(A+B)
```

規則 2 ：

若E爲一個表示式，且它的第一字元不是運算子，則+E和－E被稱作正、負號的表示式。例如：

```
-S              +JOBNO    -217    +17.26
-SQRT(A+B)
```

規則 3 ：

若E爲一個表示式，則（ E ）代表E被計算後所得的數值。例如：

```
(-A)    -(JOBNO)    -(X+1)    (A-SQRT(A+B))
```

規則 4 ：

若E是一個無正、負號的表示式而F爲任意的表示式，則F+E，F－E，F＊E，F/E 和F＊＊E等全部都是表示式。例如：

```
-(B(I,J)+SQRT(A+B(K,L)))
1.7E-2**(X+5.0)
-(B(I+3,3*J+5)+A)
```

規則 5 ：

一個被計算的表示式可能是整數、擴充的整數、實數、雙精確度或邏輯的型態，此型態是由表示式中元素的資料型態來決定。若是表示式中的元素之資料型態並不全部相同，

則表示式的型態是由其中具有最高型態的元素決定之。資料型態的層次（由最高到最低）依序如下：　DOUBLE PRECISION, REAL, INTEGER*4, INTEGER, LOGICAL。

規則6：

在表示式中，不允許兩個算術運算子相鄰。例如：

A/−B　必須寫成A/(−B)
2*−3　必須寫成2*(−3)

規則7：

在算術表示式中，全部的計算必須以算術運算子來指定。亦即，若有兩個以上的運算元必須以運算子來分開全部的運算元。例如：

(X)(Y)　必須寫成X*Y
2(K+3)　必須寫成2*(K+3)

規則8：

在表示式中，計算的先後順序是依據運算子的優先序（Precedence）決定。若爲相同的優先序，則依左至右的順序作計算。又括號是超越運算子，必須最先計算。以下是運算子的優先序：

優先序	運算子
第一	** （指數）
第二	− （負號）
第三	*, /（乘、除）
第四	+, −（加、減）

考慮以下FORTRAN的表示式：

```
-A*B  須當作 (-A)*B
-A+B  須當作 (-A)+B
-A**2+1 須當作 (-(A**2))+1
```

一個計算表示式的數值之實例如下：

已知一表示式　6 + 3 * 5 − 8/2 * * 2

計算數值的過程：

①求 2 的平方得 4

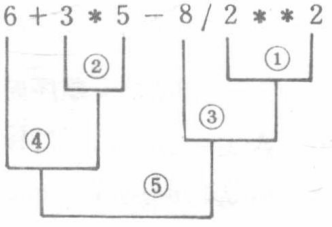
② 3 乘以 5 得 15

③ 8 除以①式的結果得 2

④ 6 加上②式的結果得 21

⑤最後，④式的結果減去③式的結果得到答案 19

以上計算的過程若以下式來表示將更清楚：



規則 9 ：

在表示式中使用到的任意變數，必須有前面已經給予定義（此變數須有一個數值）。考慮以下的設定敍述：

```
N = N + 1
```

如果變數N在此敍述之前，已經給予一個數值，則執行此一敍述後，N將得到一個適當的新數值。反之，將會有錯誤發生。

規則10：

在表示式中可以有多層次的括號存在，例如：

```
A*(Z-((Y+X)/T))**J+VAL
```

其中，Y＋X爲最內層的元素，(Y+X)/T 是次內層，Z－((Y+X)/T 爲再次內層。在此例子中，請注意左括號的數目須等於右括號的數目。又求此表示式的數值時，首先從最內層的元素開始計算，然後計算次內層，直到此表示式的數值被求得爲止。此例子的計算過程如下：

① e1＝Y＋X
② e2＝(e1)/T
③ e3＝2－e2
④ e4＝e3**J
⑤ e5＝A*e4
⑥ e6＝e5＋VAL

在步驟⑥中所得的 e6 ，即爲此表示式的數值。

§3.2　邏輯表示式

一個**邏輯表示式**是包含邏輯常數、變數或陣列元素之式子，它提供此式一個邏輯值 .TRUE. 或 .FALSE.。以下是一些邏輯表示式的例子：

```
.TRUE.
.FALSE.
X.AND.Y
X.OR.Y
.NOT.X
```

在上面的例子中，名稱X，Y爲邏輯變數。若X和Y全部爲.TRUE.，則X.AND.Y得到.TRUE.值。若X或Y有一個爲.TRUE.，則X.OR.Y得到.TRUE.值。而.NOT.X則產生和X相反的邏輯值。

一個邏輯表示式可以下面任何一種形式出現：

1. 邏輯表示式可以是單一的邏輯常數（亦即，.TRUE.或.FALSE.）、單一的邏輯變數、邏輯陣列元素或邏輯函數。
2. 由兩個算術表示式以**關係運算子**（Relational operator）結合一起形成邏輯表示式。
   ＜註＞ 關係運算子將在下節中詳述。
3. 邏輯運算子可以在邏輯常數、邏輯變數、邏輯陣列、邏輯函數、關係表示式或其它邏輯表示式之間作運算。

§3.3 關係表示式

關係表示式主要用來測試兩個數值間的關係，它的一般形式是：

```
e1 γ e2
```

其中，e1和e2 爲算術表示式，γ稱爲**關係運算子**。六種FORTRAN 的關係運算子與它們的數學上等效符號及運算如下:

關係運算子	數學等效符號	運　　算
.LT.	<	小於
.LE.	≤	小於或等於
.GT.	>	大於
.GE.	≥	大於或等於
.EQ.	=	等於
.NE.	≠	不等於

如果由關係運算子所定義的條件滿足的話，此關係表示式將得到 .TRUE. 值。反之，關係表示式將得到 .FALSE. 值。以下是一些關係表示式的例子：

```
A.EQ.B
(A**J).GT.(ZAP*(RHO*TAU-ALPH))
```

§3.4　邏輯運算子

在下表中，將列出全部的邏輯運算，其中，U和V代表邏輯表示式。

表 3-1　邏輯運算一覽表：

邏輯運算	說　　明
.NOT.U	此表示式的數值將是U的邏輯補數（亦即，1變成0；0變成1）。

U.AND.V　此表示式的數值將是U和V的邏輯乘積（亦即，若乘積的結果爲1，必須U和V在相對應的位元均爲1）。

U.OR.V　此表示式的數值將是U和V的邏輯和（亦即，若邏輯和爲1，必須U或V在相對應的位元上爲1，或U和V的相對應位元均爲1。

U.XOR.V　此表示式的數值是U和V作互斥OR（exclusive OR)的結果（亦即，若U和V在相對應的位元分別爲0和1或1和0，將得到數值1）。

實例：

已知U = 01101100，V = 11001001

則得到

```
.NOT.U = 10010011
U.AND.V = 01001000
U.OR .V = 11101101
U.XOR.V = 10100101
```

以下將對邏輯表示式的結構，作更進一步地討論：

1. 任何邏輯表示式，可以包含在括號內。注意，在邏輯表示式中，若使用.NOT.運算子且包含兩個或兩個以上的元素，則必須包含在括號內。
2. 在運算的先後順序上，括號用來代表最優先的計算順序。其它邏輯運算的先後順序如下：

a. 函數參考
b. 指數（**）
c. 乘和除（*和/）
d. 加和減（+和-）
e. 關係運算子（.LT.,.LE.,.EQ.,.NE.,.GT.,.GE.）
f. .NOT.
g. .AND.
h. .OR.,.XOR.

實例：

已知表示式

```
X.AND.Y.OR.B(3,2).GT.Z
```

其計算過程是

```
①  e1 = B(3,2).GT.Z
②  e2 = X.AND.Y
③  e3 = e2.OR.e1
```

實例：

已知表示式

```
X.AND.(Y.OR.B(3,2).GT.Z)
```

其計算過程是

```
①  e1 = B(3,2).GT.Z
②  e2 = Y.OR.e1
③  e3 = X.AND.e2
```

3. 連續兩個邏輯運算子在一起是不適當的邏輯表示式，除非其中第二個邏輯運算子爲 .NOT. 則例外。例如：

.AND..NOT.
或 .OR..NOT.

將是允許的邏輯表示式。

例如：

A.AND..NOT.B　　是允許的
A.AND..OR.B　　是不允許的

§3.5 表示式中的荷氏碼、文字串、和十六進制常數

荷氏碼、文字串，和十六進制常數被允許存在於表示式中，以取代整數常數。這些特殊的常數通常可以計算得到一個整數值，同時其資料長度被限制爲兩個位元組。但有一些例外是：

1. 長的荷氏碼或文字串，可以被當作副程式的參數。
2. 當結合實變數時，在DATA敍述中的荷氏碼、文字串、或十六進制常數可以有最多爲四個位元組的資料長度。或者在結合雙精確度變數時，其資料長度可以到達八個位元組。

第四章

替代敘述與規格敘述

從本章開始，我們將陸續地介紹FORTRAN-80所使用的各種敘述。誠如第二章所說的，敘述是構成FORTRAN程式的主體，因此各種敘述的寫法和用法對讀者而言是相當重要的，期望讀者能在以後幾章中多下工夫，好好地學習。本章主要是介紹替代敘述與規格敘述的各種型態和用法，並列舉實例說明以幫助讀者易於瞭解。

§4.1　替代敘述（Replacement Statements）

替代敘述是用來定義計算的式子，它和一般數學上的方程式之作用非常相似。替代敘述的通式爲：

V = e

其中，V爲任意的變數名稱或陣列元素，e爲一個表示式。在FORTRAN 的語義上，等號（ = ）被定義爲替代（ Replaced ）之含義，並非一般的等效於（ is equivalent to ）。因此，在FORTRAN 程式的執行中，首先將替代敘述等號右邊的表示式

求得一數值，並將所求得的數值存入等號左邊的變數名稱或陣列元素所指定的儲存位置（亦即，以表示式的數值取代等號左邊的變數或陣列元素）。

以下是應用在替代敍述的一些規則：

1. V和等號（＝）必須出現在同一行，此一規則即使是替代敍述爲邏輯的 IF 敍述中的一部分，亦將適用。
   例如：

```
C IN A REPLACEMENT STATEMENT THE '='
C      MUST BE IN THE INITIAL LINE.
       A(5,3) =
      1           B(7,2) + SIN(C)
```

   表示繼續行

   在上例中，A(5，3)＝ 必須在敍述的開始行，除非此敍述是邏輯的 IF 敍述的一部分。若A(5，3)＝ 使用在邏輯的 IF 敍述中，則必須在 IF 敍述的第一行出現。

2. 如果變數V和表示式e的資料型態不一致的話，則由表示式求得的數值將被轉換，使得此數值和變數的型態相一致。表 4-1 將示出那一種型態的表示式和那一種變數

表 4-1　資料型態的替代一覽表

變數型態	表示式（e）型態				
	整數	實數	邏輯	雙精確度	擴充整數
整數	Y	Y_a	Y_b	Y_a	Y_g
實數	Y_c	Y	Y_c	Y_e	Y_c
邏輯	Y_d	Y_a	Y	Y_a	Y_d
雙精確度	Y_c	Y	Y_c	Y	Y_c
擴充整數	Y_f	Y_e	$Y_{b,f}$	Y_e	Y

的型態相同。在表 4-1 中，Y 代表適當的替代，N 代表不適當的替代。在 Y 之下加足標表示須考慮轉換，其轉換之方法如下述。

轉換方法：

a. 實數表示式轉換成整數，有可能發生截尾現象，以和整數資料的範圍相一致。
b. 正、負號被延伸至第二位元組。
c. 將表示式的整數值以近似實數值，設定給變數。
d. 將整數表示式（使用低階位元組，忽略正、負號）的截尾數值，設定給變數。
e. 將實數表示式的概略值，設定給變數。
f. 正、負號被延伸至第三、第四位元組。
g. 將擴充整數表示式的截尾數值，設定給變數。

替代敍述中的表示式 e，可以是算術表示式或邏輯表示式，下面列舉一些替代敍述的正確寫法：

	替　代　敍　述	數學上意義或說明
①	D=B**2-4.*A*C	$d = b^2 - 4ac$
②	B = 10	b = 10
③	I = I + 1	i = i + 1
④	J = 4.*C-6*K1*K2	$j = 4C - 6K_1K_2$
⑤	A = .TRUE.	變數A的值爲眞
⑥	B = I.GT.J	變數B的值決定於邏輯表示式 I.GT.J 的邏輯值（眞或假）

⑦ B=.I.EQ.6.OR.X.LT.Y+1.6　若I的值等於6或X的值大於Y+1.6之值，則B得到.TRUE.值。反之，B的值爲.FALSE.。

實例：請利用替代敍述，將一實數型態的HOURS，轉換成整數型態的幾時、幾分、幾秒。假設已知HOURS=2.16。

＜分析＞利用實數型表示式轉換成整數型變數時，產生截尾現象（去掉小數部分），而得到整數型變數。

程式：

```
A>TYPE HOURTR.FOR
       INTEGER  HOUR,MINUTE,SECOND
       REAL  HOURS,MIN,SEC
       HOURS=2.16
       HOUR=HOURS
       MIN=(HOURS-HOUR)*60.
       MINUTE=MIN
       SEC=MIN-MINUTE
       SECOND=60.*SEC
       WRITE (3,20) HOUR,MINUTE,SECOND
20     FORMAT(1X,'HOUR= ',I2//1X,'MINUTE= ',I2//1X,'SECOND= ',I2)
       STOP
       END
```

執行過程及結果：

```
A>B:F80 =HOURTR
$MAIN

B>L6
B>L80 A:HOURTR,A:HOURTR/N/G

Link-80  Vers. 3.41
Copyright (C) 1980 by Microsoft
Created: 28-Dec-80
```

```
Data     0103     1D5D     < 7258>

33314 Bytes Free
[014F    1D5D         29]
[Begin execution]

HOUR=  2

MINUTE=  9

SECOND= 36 STOP
```

由上面執行結果，2.16 小時轉換成 2 小時 9 分 36 秒。

實例：試求 1 到 10 之總和及平均值，並從螢幕上印出。

＜分析＞先假設總和 SUM 的起始值爲 0 數值 I = 1，然後利用新的總和 SUM = SUM + I，逐一加至 10。又平均值等於 SUM / 10。

程式：

```
A>TYPE SAVG.FOR
C         THIS PROGRAM IS USED TO FIND
C           THE AVG OF NUMBER 1-10
          INTEGER  SUM
          I=1
          SUM=0
10        SUM=SUM+I
          I=I+1
          IF (I.LE.10) GO TO 10
          AVG=SUM/10.
          WRITE (3,40)SUM,AVG
40        FORMAT(1X,'SUM= ',I3,5X,'AVG= ',F4.2)
          STOP
          END
```

執行過程及結果：