Массивы

Массивом называется упорядоченная последовательность величин обозначенных одним именем. Переменная с индексом является элементом массива A(i, j).

Имеет до 7 |измерений. В FORTRAN – 2008 15 измерений.

Пример (для Фортран 90).

REAL, DIMENSION(2, -10:5, -5:5,4, -3:2, 5, 2) : : S

Некоторые понятия: 1) Размер; 2) Ранг массива; 3) Экстент; 4) Форма массива.

В нашем случае (2, 16, 11, 4, 6, 5, 2)

Расположение в памяти.

Элементы массива расположены «по столбцам».

Сечения массивов. (для Фортран – 90 и выше)

Примеры. A (I : K), B(2, 1 : N – 1)

Нерегулярное сечение массива:

INTEGER, DIMENSION (1:4) : : INDEX = (/2, 7, 3, 19 /)

REAL, DIMENSION (1: 50) : : A

Выражение A (INDEX) – создает сечение из 4-х эл-тов с индексами 2, 7, 3, 19.

Для выбора подмножеств используется триплет вида: [индекс1] : [индекс2] : [индекс3]

1 – нижняя граница сечения; 2 – верхняя; 3 - шаг с которым выбираются элементы;

Пример. REAL A (1: 10)

A (3 : 7 : 2) = 3.0 Эл-ты A(3), A(5), A(7) каждый получит значение 3.0

Пример. B (I, 100 : 0 : -10) можно выбирать элементы в обратном порядке.

Обращение к элементам и подобъектам массивов

Простейший случай - имя_массива (список_индексов)

В общем случае - частный_указатель [% частный_указатель] ….. ,

частный_указатель – частное_имя [(список_индексов)]

Пример.

```
TYPE  LEVEL
CHARACTER * 10  : :  LABEL
INTEGER, DIMENSION ( 3 )   : :  ORTS
REAL ,  DIMENSION ( 3, 3 )   : :   PHASE
END TYPE LEVEL
```

Массив структур типа LEVEL :

```
TYPE ( LEVEL ) , DIMENSION ( 20, 30, 60 )   : :   LEV
```

Элемент этого массива - скаляр типа LEVEL LEV (I-1, J*2, K+L)

К отдельным элементам можно обратиться LEV (I-1, J*2, K+L)%ORTS (2)

LEV (I-1, J*2, K+L)%PHASE(2, 3)

Но ***невозможно*** сформировать обращение LEV%ORTS (2) и LEV%PHASE(2,3)

Встроенные операции и функции для работы с массивами

1) Конструкторы массивов (/ список_значений)
2) S= A + B ; S= A * B; где S, A, B матрицы
3) Используются логические массивы – маски
4) Оператор и конструкции WHERE. (аналог IF)
   a) WHERE (A > 0.0) A = LOG (A)
   b) WHERE (логическое_выражение_массив)
      присваивание_массивов
      ELSE WHERE
      присваивание_массивов
      END WHERE

5) FORALL (i = 2 : n, j = 4 : m , A (i , j) > 0) A (i , j) = 1 / A (i , j)

6) ***Динамические массивы***

Описание : REAL , ALLOCATABLE, DIMENSION (: , :) : : X

Выделение памяти : ALLOCATE (X (N, N))

Освобождение памяти : DEALLOCATE (X)

Пример. REAL, ALLOCATABLE, DIMENSION (: , : , :) : : Y

……

ALLOCATE (Y (2 : 10 , 3 : k , - 2 : k – 1) , STAT = IER)

Параметр STAT = 0 – выделение прошло успешно; STAT > 0 – выделение не удалось

NULLIFY (список_ссылочных_объектов) – открепить ссылку от адресата

7) Размеры массива и его форма. SIZE SHAPE

Пример. ALLOCATE (Y (N, N, N))

ALLOCATE (X (SIZE (Y))

8) Справочные функции для массивов. ALLOCATED (ARRAY)

LBOUND (ARRAY [, DIM])

UBOUND (ARRAY [, DIM])

9) Функции для поиска в массиве.

a) MAXLOC (ARRAY [, MASK] (MINLOC)

b) MAXVAL (ARRAY [, MASK] (MINVAL)

10) Операции над массивами.

а) функции перемещения эл-тов массива;

б) операции упаковки и распаковки массивов

в) логические и числовые редукции массивов

г) и др.

ALL (MASK)

PRODUCT (ARRAY)

SUM (ARRAY)

TRANSPOSE (MATRIX)

MATMUL (MATRIX_A, MATRIX_B)

Правила установления соответствия между формальными и фактическими аргументами при обращении к подпрограммам

1) Необходимо, чтобы количество, тип и порядок следования фак. аргументов в указателе ф-ции или в операторе CALL соответствовали количеству, типу и порядку следования аргументов в операторе FUNCTION или SUBROUTINE
2) Между форм. и факт. аргументами должно быть установлено следующее соответствие

Формальный аргумент	***Фактический аргумент***
Переменная	Константа, переменная, элемент массива, выражение
Массив	Массив , элемент массива
Имя внешней подпрограммы	Имя внешней подпрограммы

3) Если форм. аргумент – массив, то соответствующий факт. аргумент должен быть либо массивом либо эл-том массива. Размер факт. массива >= форм.

Размер и ранг фактического массива и формального могут ***не совпадать.***

Пример.
```
DIMENSION A(7)                    SUBROUTINE  S2 ( B )
……                                DIMENSION  B (3, 2 )
CALL   S2 ( A )
```

Тогда	A (1)	A (2)	A (3)	A (4)	A (5)	A (6)	A (7)
	B(1,1)	B(2,1)	B(3,1)	B(1,2)	B(2,2)	B(3,2)	

4) Формальный аргумент – массив может быть описан в подпрограмме оператором

DIMENSION с ***max*** значениями всех измерений, заданными целыми переменными.

Пример. SUBROUTINE S3 (X, N, M)

DIMENSION X (N, M)

Обращение к S3 может быть таким : DIMENSION A (4, 2)

CALL S3 (A, 4, 2) тогда полное ***соответствие*** между эл-тами A(4,2) и X(4,2)

Но если при обращении к S3 оператором CALL S3 (A, 2, 2) , получим

A(1,1) A(2,1) A(3,1) A(4,1) A(1,2) A(2,2) A(3,2) A(4,2)

X(1,1) X(2,1) X(1,2) X(2,2)

5) Константа, выражения и имя подпрограммы могут быть использованы в качестве фактического аргумента только если в подпрограмме соответствующему формальному аргументу ***не присваивается значение.***

6) Фактический аргумент, являющийся именем внешней подпрограммы, должен быть определён в операторе EXTERNAL вызывающей программы.

Пример.

Составить программу вычислений : $S_1=\sum_{i=10}^{40} \frac{2i}{i^2+1}$; $S_2=\sum_{i=1}^{20} \frac{i-10}{2i^2+10}$; $S_3=\sum_{i=5}^{15} \frac{i+1}{2i^2-1}$

В подпрограмме нужно описать вычисление суммы значений некоторой формальной функции

$$S = \sum_{i=k}^{n} f(i)$$

```
      EXTERRNAL  F1, F2, F3
      S1 = SUM ( 10, 40, F1 )
      S2 = SUM ( 1, 20, F2 )
      S3 = SUM ( 5, 15, F3 )
      ……..
      STOP
      END
      FUNCTION  SUM (K, N, F )
      SUM = 0
      DO  10   I = K, N
                  SUM = SUM + F ( I )
10    CONTINUE
      RETURN
      END
      FUNCTION  F1 ( I )
      F1 = 2. * I / ( I*I + 1 )
      RETURN
      END
      FUNCTION  F2 ( I )
      ……..
      FUNCTION  F3 ( I )
      ……..
```

Основная программа

Подпрограмма SUM

Подпрограмма F1

Подпрограмма F2

Подпрограмма F3