Những yếu tố cơ bản của Fortran

Bởi:

PGS. TS. NGƯT Phạm Văn Huấn

Dữ liệu và cách biểu diễn dữ liệu trong Fortran

Fortran có thể thao tác với sáu loại (kiểu) dữ liệu cơ bản thường gặp trong thực tế là: các số nguyên, số thực, số phức, số thực độ chính xác gấp đôi, các giá trị lôgic và dữ liệu văn bản. Trong chương này ta sẽ làm quen với các dữ liệu kiểu số nguyên, số thực, giá trị lôgic và văn bản (chuỗi ký tự).

Số nguyên là liệt các số thập phân với dấu +, ? hoặc không có dấu. Thí dụ:

0 ; 6 ; ?400 ; +1234

Các số nguyên được biểu diễn dưới dạng *I*. Giá trị cực đại của số nguyên gọi là khả năng biểu diễn số nguyên của máy tính.

Trong Fortran có hai dạng biểu diễn *số thực*. Dưới *dạngF* số thực gồm phần nguyên và phần thập phân, cách nhau bởi dấu chấm. Số thực có thể có dấu dấu +, ? hoặc không có dấu. Nếu phần nguyên hoặc phần thập phân bằng không, có thể không cần viết ra các phần đó. Dấu chấm thập phân nhất thiết phải có mặt. Thí dụ:

?2.583 ; 14.3 ; 0.8 ; 12. ; .7 ; 14.

Giá trị cực đại và số chữ số có nghĩa cực đại trong dạng *F* phụ thuộc vào dạng, hay kiểu (kind) khai báo của số thực.

DạngE biểu diễn số thực thành hai phần: phần hằng thực nằm trong khoảng từ 0,1 đến 1,0 và phần bậc. Bậc bắt đầu bằng chữ *E*, tiếp sau là hằng nguyên gồm không quá hai chữ số thập phân, có thể có dấu hoặc không dấu. Thí dụ số 25000 có thể viết dưới dạng *E* là 0.25E05. Số chữ số có nghĩa của phần hằng thực và hằng nguyên cũng tùy thuộc loại số thực khai báo.

Hằng với độ chính xác gấp đôi (dạng *D*) có thể viết như số với dấu chấm thập phân, chứa từ 8 đến 16 chữ số có nghĩa, hoặc như số dạng mũ với chữ *D* thay vì *E*, trong đó phần hằng thực có thể chứa tới 16 chữ số có nghĩa. Thí dụ:

2.71828182 ; 0.27182818D+1

Trị tuyệt đối cực đại của các số thực thường và độ chính xác gấp đôi bằng 10^{-79} đến 10^{75} .

Số phức biểu diễn bằng một cặp hằng thực trong dấu ngoặc đơn và cách nhau bởi dấu phảy. Thí dụ (2.1, 0.5E2) biểu diễn số phức $2{,}1 + 50i$ trong toán học.

Hai số trong dấu ngoặc ứng với các phần thực và phần ảo phải cùng độ chính xác biểu diễn.

Các giá trị dữ liệu văn bản dùng để biểu diễn các đoạn văn bản như tên các đại lượng, các khái niệm, thí dụ cụm chữ "Toc do", "Temperature", "BAO CAO SO 1"... Người ta còn gọi dữ liệu văn bản là dữ liệu ký tự, xâu ký tự, dữ liệu chữ.

Các chữ số 1, 2, ..., 9, 0 khi dùng với tư cách là để biểu diễn các giá trị số tương ứng thì chúng cũng là những dữ liệu kiểu văn bản.

Dữ liệu lôgic dùng để chỉ khả năng có hay không của một sự kiện, đúng hay sai của một biểu thức quan hệ. Người ta dùng hai giá trị lôgic là **.TRUE.** và **.FALSE.** để chỉ hai trạng thái đối lập nhau trong những thí dụ trên và ngôn ngữ Fortran có thể xử lý với những giá trị lôgic, tức thực hiện những phép tính đối với các giá trị lôgic như trong toán học có thể thực hiện.

Sở dĩ máy tính làm được những việc như chúng ta thấy là vì nó có thể xử lý thông tin, so sánh, tính toán được với những kiểu dữ liệu này và đưa ra những kết luận, thông báo... Tất cả những thông tin chúng ta gặp trong đời sống thực tế đều có thể được biểu diễn bằng những dữ liệu kiểu này hoặc kiểu khác.

Trên đây là *những kiểu dữ liệu cơ bản* của ngôn ngữ lập trình Fortran. Sau này và ở các chương khác, chúng ta sẽ thấy còn có những kiểu dữ liệu khác được tổ chức dựa trên những kiểu dữ liệu cơ bản vừa trình bày.

Ở đây chúng ta cần lưu ý rằng những khái niệm dữ liệu trong máy tính như số nguyên, số thực... nói chung giống với những khái niệm tương ứng trong đời sống hoặc trong toán học. Nhưng đồng thời cũng có những nét khác biệt. Thí dụ, Fortran chỉ hiểu và tính toán được với những số nguyên loại thường không lớn hơn $2 \cdot 10^9$, ngôn ngữ lập trình Pascal chỉ làm việc với những số nguyên không lớn hơn 32767 và không nhỏ hơn ?32768, trong khi hàng ngày chúng ta có thể viết trên giấy hoặc tính toán các phép tính với những số nguyên có giá trị tùy ý. Tình hình cũng tương tự như vậy đối với các số

thực. Vậy trong máy tính có những giới hạn nhất định trong việc biểu diễn các số, không phải số nào máy tính cũng biểu diễn được và tính toán được. Tuy nhiên, với những giới hạn như hiện nay, Fortran vẫn cho phép chúng ta lập các chương trình để tính toán, xử lý với tất cả những giá trị số gặp trong đời sống và khoa học kỹ thuật.

Hằng và biến

Máy tính xử lý dữ liệu hay thực hiện những tính toán với những đại lượng. Tất cả những đại lượng đó phải được lưu giữ trong máy tính. Những đại lượng không đổi trong suốt quá trình thực hiện của chương trình gọi là *các hằng*, còn những đại lượng có thể nhận những giá trị khác nhau gọi là *các biến*. Với mỗi hằng hoặc biến, trong bộ nhớ máy tính giành ra một địa chỉ để lưu giá trị. Tên chính là ký hiệu quy ước của địa chỉ đó.

Tên biến và tên hằng

Tên biến trong Fortran chuẩn được biểu diễn bằng tập hợp từ 1 đến 6 các chữ cái trong bảng chữ cái la tinh (26 chữ cái) hoặc các chữ số 0, 1, ..., 9, nhưng phải bắt đầu bằng chữ cái.

Trong một chương trình các tên biến không được trùng nhau. Trong các phiên bản Fortran hiện nay, để dùng làm tên không phân biệt chữ cái hoa và chữ cái thường. Ngoài ra, còn một vài ký tự khác cũng có thể dùng để cấu tạo tên. Phiên bản Fortran 90 cho phép đặt tên với số ký tự dài hơn 6 và trong tên có thể có một số ký tự khác nữa. Tuy nhiên, sinh viên nên tập thói quen đặt tên gọn gàng theo Fortran chuẩn, bởi vì tập hợp 6 ký tự đã rất đủ để chúng ta mô tả các bài toán, kể cả những bài toán lớn và phức tạp.

Thí dụ, các tên sau đây

X ; A ; X1 ; B2T5 ; SOHANG ; SUM là hợp lệ, còn các tên sau đây là sai:

1NGAY ; HE SO ; B*T

vì trong tên thứ nhất ký tự đầu tiên là chữ số, trong tên thứ hai có ký tự dấu cách, trong tên thứ ba có ký tự (*) không phải là những ký tự dùng để đặt tên.

Quy tắc đặt tên biến trên đây cũng áp dụng đối với tên chương trình, tên hằng, tên các chương trình con và tên file. (Riêng với tên file có thể có thêm phần mở rộng gồm không quá ba chữ cái hoặc chữ số ngăn với phần tên chính bởi dấu chấm).

Mô tả (khai báo) kiểu biến và kiểu hằng

Kiểu của biến tương ứng với kiểu dữ liệu mà nó biểu diễn. Các biến nguyên biểu diễn các dữ liệu số nguyên, các biến thực - số thực... Trong chương trình phải chỉ rõ các biến

được sử dụng biểu diễn dữ liệu kiểu nào (nguyên, thực, lôgic, phức, văn bản, số thực độ chính xác thường hay độ chính xác gấp đôi...).

Mỗi biến chỉ lưu giữ được những giá trị đúng kiểu của nó. Một biến đã mô tả kiểu là số nguyên thì không thể dùng để lưu giá trị số thực hay giá trị lôgic.

Cách *mô tả ẩn* chỉ dùng đối với các biến nguyên và thực: dùng tên biến nguyên bắt đầu bằng một trong sáu chữ cái I, J, K, L, M, N, còn tên biến thực bắt đầu bằng một trong những chữ cái ngoài sáu chữ cái trên. Nói chung, người mới học lập trình không bao giờ nên dùng cách mô tả ẩn.

Cách *mô tả hiện* dùng các lệnh mô tả hiện như INTEGER, REAL, CHARACTER, LOGICAL, DOUBLE PRECISION, COMPLEX... để chỉ kiểu dữ liệu mà các biến biểu diễn. Dưới đây là quy tắc viết những lệnh mô tả kiểu dữ liệu: tuần tự nguyên, thực, lôgic, phức, thực độ chính xác gấp đôi và ký tự văn bản:

INTEGER *D anh sách các biến nguyên*

REAL *D anh sách các biến thực*

LOGICAL *D anh sách các biến lôgic*

COMPLEX *D anh sách các biến phức*

DOUBLE PRECISION *D anh sách các biến độ chính xác đôi*

CHARACTER *D anh sách các biến ký tự*

Trong danh sách các biến sẽ liệt kê các tên biến, nếu có hơn một biến thì các biến phải cách nhau bởi dấu phảy.

Thí dụ:

INTEGER I, TT, DEM

REAL X1, APSUAT, MAX, TIME, DELTA

COMPLEX P1, P2, SOPH

chỉ rằng các biến I, TT, DEM biểu diễn các giá trị số nguyên, các biến X1, APSUAT, MAX, TIME, DELTA biểu diễn các giá trị số thực, còn ba biến P1, P2, SOPH - số phức.

Những giá trị được giữ nguyên nhất quán trong suốt chương trình (tức các hằng số) thường được gán vào các địa chỉ nhớ thông qua tên trong lệnh khai báo hằng có dạng:

PARAMETER (ten1 = biểu thức 1, tên 2 = biểu thức 2, ...)

Thí dụ, trong chương trình nếu ta nhiều lần dùng đến giá trị số π = 3,141593 thì ta có thể gán giá trị 3,141593 cho một tên hằng là PI bằng lệnh

PARAMETER (PI = 3.141593)

Lệnh sau đây

PARAMETER (HSMSD = 0.0026, RO = 1.0028)

khai báo hai hằng số: HSMSD và RO, HSMSD được gán giá trị bằng 0,0026, còn RO được gán giá trị 1,0028.

Trong chương trình tất cả những lệnh khai báo (mô tả) vừa giới thiệu trên đây thuộc loại các lệnh không thực hiện và chúng phải nằm ở đầu chương trình, trước tất cả các lệnh thực hiện.

Khái niệm về tên, kiểu dữ liệu của biến, của hằng là những khái niệm cơ bản, quan trọng trong ngôn ngữ lập trình.

Ở đầu mục này đã nói một tên thực chất là ký hiệu quy ước của một địa chỉ trong bộ nhớ của máy tính để lưu giá trị. Lệnh khai báo biến mới chỉ đặt tên cho một địa chỉ trong bộ nhớ và quy định trong địa chỉ đó có thể lưu giữ dữ liệu kiểu gì. Còn cụ thể trong ô nhớ đó đã có chứa giá trị chưa hay chứa giá trị bằng bao nhiêu thì tùy thuộc vào các lệnh thực hiện ở trong chương trình, tại từng đoạn của chương trình. Điều này giống như ta quy ước định ra một ngăn trong tủ văn phòng để chuyên giữ các công văn, còn trong ngăn ấy có công văn hay không, hoặc có mấy công văn thì tùy thuộc lúc này hay lúc khác. Dưới đây nêu một thí dụ để minh họa ý nghĩa của việc đặt tên biến và mô tả kiểu (dữ liệu) của biến, đồng thời theo dõi giá trị của biến tại từng thời điểm của chương trình. Giả sử ta viết một chương trình để tính diện tích *s* của hình tam giác khi giá trị độ dài đáy *b* bằng 5,0 cm, chiều cao *h* bằng 3,2 cm, in kết quả tính lên màn hình. Chương trình sau đây sẽ thực hiện những việc đó:

```
REAL DAY, CAO ! (1)

DAY = 5.0 ! (2)

CAO = 3.2 ! (3)

DAY = 0.5 * DAY * CAO ! (4)

PRINT *, 'DIEN TICH TAM GIAC BANG', DAY ! (5)
```

END ! (6)

Trong chương trình này có sáu lệnh. Lệnh (1) khai báo hai biến tên là DAY và CAO dự định để lưu giá trị số thực tương ứng của đáy *b* và chiều cao *h* của tam giác. Lệnh (2) gán giá trị $b = 5{,}0$ (cm) cho biến DAY. Lệnh (3) gán giá trị $h = 3{,}2$ (cm) cho biến CAO. Lệnh (4) tính giá trị của biểu thức $0{,}5 \times b \times h$, tức diện tích *s* của tam giác, bằng 8 (cm^2) và gán cho biến DAY. Lệnh (5) in lên màn hình dòng chữ DIEN TICH TAM GIAC BANG và sau đó là giá trị của biến DAY. Lệnh (6) là lệnh kết thúc chương trình. Sinh viên mới học lập trình thường có thể không hiểu lệnh thứ năm, khi thấy in diện tích hình tam giác mà lại in giá trị của biến DAY. Trong đầu họ quen nghĩ khai báo DAY có nghĩa DAY là độ dài cạnh đáy tam giác. Nhưng nếu hiểu được rằng lệnh (1) khai báo REAL DAY, CAO thực ra mới chỉ dự định dùng hai tên DAY và CAO để lưu các số thực, không cần biết số thực đó bằng bao nhiêu. Ở chương trình trên, khi lệnh (2) thực hiện xong thì trong biến DAY (trong ô nhớ có tên là DAY) mới thực sự có số 5,0, tức độ dài đáy tam giác. Nhưng khi chương trình chạy xong lệnh (4) thì trong biến DAY đã là số 8,0 chứ không phải là số 5,0 nữa. Và khi thực hiện xong lệnh (5) thì trên màn hình sẽ in đúng giá trị diện tích tam giác. Nắm vững được điều này có nghĩa là đã hiểu được ý nghĩa của biến, tên biến và tuần tự làm việc của chương trình, tức các giá trị được lưu trong máy tính như thế nào trong khi chương trình chạy.

Dưới đây là hai lời khuyên đầu tiên có lẽ quan trọng nhất đối với sinh viên mới học lập trình:

1) Sau khi tìm hiểu xong bài toán cần giải, phải cân nhắc từng đại lượng trong bài toán có kiểu dữ liệu là số nguyên, số thực, ký tự văn bản... để đặt tên và khai báo kiểu cho đúng. Kinh nghiệm cho thấy rằng sinh viên nào viết được những lệnh khai báo hệ thống các tên biến đúng, vừa đủ, sáng sủa trong phần khai báo ở đầu chương trình thì thường là sau đó viết được chương trình đúng. Còn những sinh viên không biết đặt tên cho các biến, vừa bắt tay vào soạn thảo chương trình đã loay hoay với lệnh mở file dữ liệu, tính cái này cái kia, thì thường là không hiểu gì và không bao giờ làm được bài tập.

2) Nên tuân thủ cách đặt tên của Fortran chuẩn. Ta có quyền chọn những chữ cái, chữ số nào để tạo thành tên là tùy ý, song nên đặt tên có tính gợi nhớ đến những đại lượng tương ứng trong bài tập. Thí dụ, với bài toán vừa nói tới trong mục này ta có ba đại lượng là: độ dài cạnh đáy, đường cao và diện tích tam giác. Nên khai báo tên ba biến tương ứng bằng ba từ tắt của tiếng Việt với lệnh sau:

REAL DAY, CAO, DTICH

hoặc bằng ba từ tắt của tiếng Anh với lệnh:

REAL BASE, HEIGHT, SQRE

hoặc bằng ba chữ cái đúng như trong đầu đề bài tập với lệnh:

REAL B, H, S

đều là những lời khai báo đúng, dễ hiểu, trong đó lời khai báo trên cùng có lẽ là tốt nhất, lời khai báo sau cùng thì hơi quá ngắn gọn. Còn với cùng mục đích khai báo mà dùng lệnh sau đây thì mặc dù không sai, nhưng hoàn toàn không nên, rất dễ gây nhầm lẫn, mệt mỏi trong khi kiểm tra chương trình:

REAL X, IC, DT

Biến có chỉ số (mảng)

Khái niệm mảng

Mảng là tập hợp có sắp xếp của các đại lượng được ký hiệu bằng một *tên* duy nhất. Các thành phần của tập hợp gọi là những *phần tử mảng*. Mỗi phần tử được xác định theo *tên* của mảng và vị trí của phần tử đó trong mảng, tức trị số của các *chỉ số*. Tên mảng được đặt tuân theo quy tắc như tên biến. Các chỉ số nằm trong dấu ngoặc đơn và nếu có hơn một chỉ số thì các chỉ số phải cách nhau bởi dấu phảy.

Thí dụ: A(1), A(2), A(3) tương ứng với cách viết thông thường cho các biến a_1,a_2,a_3 trong toán học. Vậy ở đây ta đã đặt cho tập hợp cả 3 giá trị này một tên chung là A, nhưng để chỉ giá trị thứ nhất ta thêm chỉ số 1 vào tên - A(1), để chỉ giá trị thứ hai ta thêm chỉ số 2 - A(2) và để chỉ giá trị thứ ba ta thêm chỉ số 3 - A(3).

Tương tự, các phần tử của ma trận hai chiều trong đại số

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix}$$

được viết trong Fortran là A(1,1), A(1,2), A(1,3), A(2,1), A(2,2), A(2,3) (chỉ số thứ nhất - số hiệu dòng, chỉ số thứ hai - số hiệu cột).

Thêm một thí dụ nữa về mảng. Một năm có 12 tháng, mỗi tháng có một tên, thí dụ trong tiếng Việt: Tháng Giêng, Tháng Hai, ..., Tháng Mười hai, trong tiếng Anh: January, February, ..., December. Ta hoàn toàn có thể gộp 12 tên tiếng Anh của các tháng trong năm vào thành một mảng có tên chung là EMONTH. Vậy mảng EMONTH sẽ là mảng có 12 giá trị (12 phần tử), mỗi phần tử là một từ chỉ tên một tháng. Khi nói đến January tức là nói tới giá trị thứ nhất của mảng EMONTH, ta viết EMONTH(1), nói đến December là nói tới giá trị thứ 12 của mảng EMONTH, ta viết EMONTH(12).

Trong Fortran IV, một phiên bản trước đây của ngôn ngũ Fortran, cho phép dùng các mảng tối đa 7 chỉ số. *Chiều* của mảng ứng với số chỉ số,còn *kích thước* của mảng ứng với số phần tử chứa trong mảng.

Chỉ số của mảng có thể được xác định bằng các hằng hoặc biến nguyên dương với trị số lớn hơn 0. Cũng có thể chỉ số xác định bằng biểu thức số học bất kỳ. Nếu dùng biểu thức kiểu thực, thì sau khi tính giá trị của biểu thức, giá trị số thực được chuyển thành số nguyên, tức cắt bỏ phần thập phân.

Trong mục 2.1 chúng ta đã nói về các kiểu dữ liệu cơ bản. Mỗi một biến kiểu dữ liệu cơ bản trong một thời điểm chạy chương trình chỉ lưu (chứa) được một giá trị. Bây giờ ta thấy mảng là một thí dụ về kiểu dữ liệu mới cấu tạo từ các kiểu cơ bản - một biến mảng trong một thời điểm có thể lưu được nhiều giá trị số nguyên, số thực, chuỗi ký tự... Nhưng cần lưu ý rằng tất cả các phần tử của mảng, tức tất cả các giá trị của mảng phải có cùng kiểu dữ liệu. Thí dụ với mảng EMONTH vừa xét, ta không thể đưa một giá trị ký tự January vào phần tử EMONTH(1) và số thực 1.27 vào EMONTH(2).

Mảng là một yếu tố rất quan trọng trong Fortran. Sau này ta sẽ thấy sử dụng mảng trong ngôn ngữ lập trình có thể giúp viết những đoạn chương trình rất ngắn gọn, trong sáng. Đặc biệt trong các vòng lặp, chỉ bằng vài dòng lệnh có thể khiến máy tính thực hiện nhiều triệu phép tính số học.

Mô tả mảng

Mô tả mảng thực hiện ngay ở đầu chương trình và chứa thông tin về tên, chiều và kích thước mảng với toán tử DIMENSION:

DIMENSION A(n_1,n_2,...,n_ℓ),MAT(m_1,m_2,...,m_k)

trong đó A, MAT- tên các mảng; n_1,n_2,...,n_ℓ,m_1,m_2,...,m_k − các giới hạn trên của các chỉ số ? chỉ ra bằng *các hằng nguyên dương* (giới hạn dưới luôn bằng 1 và không cần chỉ định trong mô tả).

Theo mô tả này, máy tính sẽ giành trong bộ nhớ những vùng địa chỉ để lưu tất cả các phần tử của các mảng. Các phần tử của mảng nhiều chiều được lưu liên tiếp nhau sao cho chỉ số thứ nhất biến đổi nhanh nhất, chỉ số sau cùng biến đổi chậm nhất.

Có thể mô tả mảng bằng các lệnh mô tả kiểu hiện như đối với các biến thông thường, thí dụ:

REAL MAX, L(7), A(20,21)

Trong lệnh mô tả này biến MAX được khai báo là biến số thực, có thể gọi là biến đơn, còn mảng L (biến có chỉ số) là mảng một chiều với 7 phần tử số thực, mảng A là mảng

hai chiều (hai chỉ số) với giới hạn trên của chỉ số thứ nhất là 20, của chỉ số thứ hai là 21, nó gồm 420 phần tử.

Vì các giới hạn chỉ số (kích thước mảng) phải được chỉ định trước ở phần khai báo bằng các hằng nguyên dương, không thể là các biến, nên trong thực tiễn lập trình phải chú ý cân nhắc chọn các giới hạn chỉ số sao cho chúng không quá lớn làm tốn bộ nhớ, nhưng cũng phải vừa đủ để biểu diễn hết các phần tử có thể có của mảng. Thí dụ cần biểu diễn một bảng số các giá trị nhiệt độ trung bình từng tháng trong 100 năm thì ta khai báo mảng TEM(100,12) là hợp lý. Nếu dự định giải hệ phương trình đại số tuyến tính không quá 20 phương trình, ta nên khai báo các mảng REAL A(20,21), X(20) là vừa đủ để biểu diễn ma trận các hệ số $a_{i,j}$ (kể cả các hệ số tự do) và các nghiệm x_i. Với mảng EMONTH vừa nhắc trong mục này thì lệnh khai báo sau:

```
CHARACTER*9 EMONTH(12)
```

là hoàn toàn hợp lý vì một năm chỉ có 12 tháng và tên tháng dài nhất (với tiếng Anh) là September gồm 9 chữ cái.

Các hàm chuẩn

Một số phép tính như lấy căn bậc hai của một số, tính trị tuyệt đối của một số, tính hàm sin của một góc... thường xuyên gặp trong nhiều thuật toán, nên được xây dựng sẵn thành các hàm gọi là các hàm riêng có của Fortran (intrinsic functions) hay còn gọi là *các hàm chuẩn*.

Bảng 2.1 liệt kê một số hàm chuẩn của Fortran thường dùng trong sách này.

Mỗi hàm chuẩn có một tên của nó. Tên của hàm được tiếp nối với đầu vào, gọi là *đối số* của hàm, nằm trong cặp dấu ngoặc đơn. Đối số của các hàm chuẩn có thể là các hằng, biến, hay biểu thức. Nếu một hàm có nhiều đối số thì các đối số được viết cách nhau bằng dấu phảy. Khi cho các giá trị cụ thể vào các đối số thì hàm tính ra một giá trị của hàm. Vì vậy các hàm thường dùng để tính một giá trị nào đó để gán vào một biến khác, người ta gọi là *gọi hàm ra để tính*. Hàm không bao giờ có mặt ở bên trái dấu ‘ = ’ của lệnh gán.

Thí dụ, những lệnh sau đây gọi các hàm để tính một số giá trị:

```
S = SIN (0.5)

TG = TAN (S)

C = COS (ANGLE * 3.141593 / 180.0)
```

Một số hàm chuẩn của Fortran

Tên hàm và đối số	Giá trị hàm
SQRT (X)	$\sqrt{x}$ Căn bậc hai của x
ABS (X)	$\|x\|$ Trị tuyệt đối của x
SIN (X)	$\sin(x)x$ tính bằng rađian
COS (X)	$\cos(x)x$ tính bằng rađian
TAN (X)	$\mathrm{tg}(x)x$ tính bằng rađian
EXP (X)	$e^x e$ nâng lên luỹ thừa x
LOG (X)	$\ln(x)$ Logarit tự nhiên của x
LOG10 (X)	$\lg(x)$ Logarit cơ số 10 của x
INT (X)	Chuyển phần nguyên của số thực x thành số nguyên
REAL (I)	Giá trị thực của I (chuyển một giá trị nguyên thành giá trị thực)
MOD (I,J)	Lấy phần dư nguyên của phép chia hai số I / J

Trong lệnh thứ nhất ta gửi giá trị hằng 0,5 (rađian) cho đối số của hàm SIN để nó tính ra giá trị sin của góc 0,5 và gán giá trị đó cho biến S. Trong lệnh thứ hai, ta đã gửi giá trị của biến S vào đối số của hàm TAN để tính ra tang. Còn trong lệnh thứ ba, ta đã gửi một biểu thức vào đối số của hàm COS để nó tính ra giá trị côsin của một góc có độ lớn bằng giá trị của biểu thức đó. Trong trường hợp này, máy tính trước hết phải tính (ước lượng) giá trị của biểu thức đối số, sau đó mới tính côsin theo giá trị nhận được.

Thấy rằng một hàm biểu diễn một giá trị. Giá trị này có thể được dùng trong các tính toán khác hoặc lưu ở địa chỉ nhớ khác. Một hàm chuẩn cũng có thể làm đối số của một hàm chuẩn khác:

```
XLG = LOG(ABS(X))
```

Trong Fortran có một số hàm chuẩn cho ra giá trị với kiểu cùng kiểu với đối số của mình, chúng được gọi là *các hàm tự sinh* (*generic function*). Thí dụ hàm ABS(X), nếu đối số X là số nguyên thì giá trị hàm ABS(X) cũng là số nguyên, nếu X là số thực - ABS(X) cũng là số thực. Một số hàm chỉ định kiểu của đầu vào và đầu ra. Thí dụ hàm IABS là hàm đòi hỏi đối số nguyên và cho ra giá trị tuyệt đối là số nguyên. Danh sách đầy đủ hơn về các hàm chuẩn của Fortran được dẫn trong phụ lục 1.

Khi dùng một hàm chuẩn nào đó phải đọc kỹ lời mô tả xem nó tính ra giá trị gì, điều kiện của các đối số ra sao. Thí dụ các hàm lượng giác phải dùng đối số là rađian, nếu ta cho giá trị đối số là độ thì kết quả tính sẽ sai.

Lệnh gán và các toán tử số học

Lệnh gán

Các tính toán trong Fortran có thể chỉ định bằng lệnh gán với dạng tổng quát như sau:

Tên biến = Biểu thức

Bên trái dấu lệnh gán (dấu =) là tên một biến. Biểu thức bên phải có thể là một hằng, một biến, một biểu thức số học gồm các toán tử số học (bảng 2.2) thực hiện giữa các toán hạng là các hằng, biến và hàm chuẩn hay một biểu thức lôgic. Khi thực hiện lệnh gán, trước hết máy ước lượng (tính) giá trị của biểu thức bên phải, rồi gán giá trị đó cho biến bên trái, tức lưu giá trị tính được của biểu thức bên phải vào địa chỉ nhớ có tên biến bên trái. Kiểu dữ liệu của biến và của biểu thức phải phù hợp.

Thí dụ các lệnh gán:

PI = 3.141593

S = PI * BKINH **2

I = I +1

Lệnh thứ nhất gán hằng số 3,141593 cho biến có tên là PI. Lệnh thứ hai gán giá trị của biểu thức PI × $(BKINH)^2$ cho biến có tên là S. Lệnh thứ ba lấy giá trị hiện tại của biến I cộng thêm một đơn vị và lại gán cho chính biến I.

Ở trên đã nói, kiểu dữ liệu của biến và của biểu thức phải phù hợp. Trường hợp biến bên trái là biến thực, còn biểu thức bên phải là giá trị nguyên thì máy tính sẽ chuyển giá trị nguyên đó thành giá trị thực (số thực với phần thập phân bằng không) rồi mới gán cho biến. Khi biến bên trái là biến nguyên, biểu thức bên phải có giá trị thực, thì máy tính cắt bỏ phần thập phân của giá trị thực, đổi số thực nhận được thành số nguyên rồi mới gán nó cho biến nguyên. Các trường hợp gán sai khác chương trình dịch sẽ báo lỗi.

Không nên quan niệm lệnh gán như dấu bằng trong toán học.

Các phép tính số học đơn giản

Các phép tính số học hay còn gọi là các toán tử số học gồm có các phép tính cộng, trừ, nhân, chia và nâng lên luỹ thừa được ký hiệu bằng các toán tử trong Fortran như trong bảng 2.2.

Gọi là những phép tính số học bởi vì các toán hạng của các phép tính là những giá trị số, thí dụ số nguyên, số thực, số phức. Sau này chúng ta sẽ thấy máy tính có thể tính toán với những giá trị kiểu khác như giá trị lôgic, giá trị văn bản...

Các phép tính số học

Phép tính	Dạng đại số	Trong Fortran
Cộng	$A + B$	A + B
Trừ	$A - B$	A ? B
Nhân	$A \times B$	A * B
Chia	$\frac{A}{B}$	A / B
Luỹ thừa	A^3	A** 3

Ước lượng biểu thức số học

Khi tính giá trị của biểu thức số học, nếu biểu thức đó gồm nhiều phép tính đơn, thì máy sẽ tính toán từng phép tính đơn để nhận các kết quả trung gian, sau đó tính giá trị cuối cùng của biểu thức gọi là ước lượng. Mức ưu tiên khi ước lượng giá trị của một biểu thức số học gồm nhiều phép tính đơn nêu trong bảng 2.3.

Nếu dấu âm đứng trước tên biến đầu tiên trong biểu thức, thì nó được tính với cùng mức ưu tiên như phép trừ. Thí dụ: ?A**2 bằng ?(A**2), ?A*B bằng ?(A*B) và ?A+B bằng (?A)+B.

Mức ưu tiên các phép tính số học

Ưu tiên	Phép tính
1	Dấu ngoặc
2	Nâng lên luỹ thừa
3	Nhân và chia
4	Cộng và trừ

Khi các phép tính ở cùng mức ưu tiên thì tất cả các phép tính được thực hiện từ trái sang phải, thí dụ:

B ? C + D được ước lượng bằng (B ? C) + D

Riêng phép nâng lên luỹ thừa thì thực hiện từ phải sang trái:

A ** B ** C được ước lượng bằng A ** (B ** C)

Thí dụ: 2**3**2 bằng 2^9 hay 512 chứ không phải là

(2**3)**2 = 8^2 = 64.

Khái niệm về cắt và các phép tính hỗn hợp

Khi một phép tính số học thực hiện với hai số thực thì đưa ra kết quả là giá trị thực. Thí dụ, khi tính chu vi hình tròn với đường kính DKINH là số thực, ta có thể dùng một trong hai lệnh sau:

CHUVI = PI * DKINH

CHUVI = 3.141593 * DKINH

Phép tính số học giữa hai số nguyên cho ra kết quả là số nguyên. Thí dụ, cho hai số nguyên I và J, trong đó I nhỏ hơn hoặc bằng J, tính số số nguyên INTERV nằm trong khoảng [I, J] có thể thực hiện bằng lệnh:

INTERV = J ? I + 1

Giả sử SIDE biểu diễn giá trị thực và LENGTH biểu diễn giá trị nguyên. Bây giờ xét lệnh:

LENGTH = SIDE * 3.5

Phép tính nhân giữa hai giá trị thực sẽ cho kết quả số thực. Tuy nhiên, giá trị thực được lưu vào biến nguyên. Khi đó máy tính sẽ bỏ qua phần thập phân và chỉ lưu phần nguyên của số thực; kiểu làm tròn này gọi là *cắt,* nó khác với làm tròn thông thường cho kết quả là số nguyên gần nhất với giá trị của số thực.

Khi các phép tính số học thực hiện giữa các biến có kiểu khác nhau (hỗn hợp) thường cho kết quả rất bất ngờ. Ta xét thí dụ tính thể tích V của hình cầu bán kính thực R. Nếu dùng lệnh:

V = (4/3)*3.141593*R**3

ta sẽ thu được kết quả sai do nguyên nhân phép chia hai số nguyên 4/3 cho giá trị trung gian bằng 1, không phải 1,333333. Do đó, lệnh đúng để tính V sẽ là:

V = (4./3.)*3.141593*R**3

Vì các phép tính hỗn hợp đôi khi cho kết quả bất ngờ, ta nên cố gắng tránh dùng những biểu thức số học có phép tính hỗn hợp.

Khái niệm về số quá bé và số quá lớn (underflow và overflow)

Vì các giá trị lớn nhất và bé nhất có thể lưu trong một biến tùy thuộc vào chính hệ máy tính, một phép tính có thể đưa ra kết quả quá lớn hoặc quá bé. Xét các thí dụ sau:

1) X = 0.25E20 2) A = 0.25E?20

Y = 0.10E30 B = 0.10E+20

Z = X * Y C = A / B

Kết quả số của phép nhân trong thí dụ 1 bằng 0.25E49, rõ ràng là có thể quá lớn, không lưu giữ được trong máy tính với bậc cực đại là 38, còn kết quả số của phép chia trong thí dụ 2 bằng 0.25E?49 sẽ quá bé. Trong những trường hợp này các lệnh Fortran hoàn toàn đúng, nhưng lỗi sẽ phát sinh khi chạy chương trình. Các lỗi do bậc quá lớn hoặc quá bé thường bị gây bởi những lỗi ở những đoạn trước của chương trình, thí dụ một biến chưa được gán giá trị đúng lại có mặt trong biểu thức số học.

Bài tập

1. Hãy biểu diễn thành dạng F và dạng E những số thực sau:

a) 3,14 b) 3,141593 c) 0,0026 d) $2{,}5 \times 10^3$

e) ?14,0 f) 28,34 g) $6{,}023 \times 10^{23}$

2. Xác định những tên sai trong những tên sau đây:

a) AVERG b) PTBACHAI c) REAL

d) 2LOG đ) GPTB2 e) HS-A1

f) X1 g) THANG*1 h) MONTH2

3. Viết thành dạng Fortran những biểu thức tính sau đây:

a) Thể tích V của hình cầu theo công thức

$V = \frac{4}{3}\pi R^3$ (R − bán kính).

b) Hai nghiệm x_1 và x_2 của phương trình bậc hai

$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$ (a,b,c − các hệ số của phương trình)

c) Giá trị hàm $y = \frac{1}{2}\sin^2 x\cos(2x -)$ (khi x cho bằng độ).

d) Giá trị hàm mật độ phân bố Gauss $F(x) = \frac{1}{2\pi}e^{-\frac{x^2}{2}}$

e) Thêm một đơn vị vào biến nguyên I và lưu vào biến I

f) Khoảng cách DIST giữa hai điểm A và B nếu biết các toạ độ tương ứng của hai điểm đó là $(x_a,y_a),(x_b,y_b)$.

4. Ước lượng giá trị của các biểu thức Fortran sau đây:

4/3*3.141593*(3/2)**3

SQRT (I+I/2) (nếu I = 1)

SIN ((30/180)*PI) (nếu PI = 3.141593)

COS (60/180*3.141593)

5. Hãy đọc chính xác bằng ngôn ngữ Fortran những lệnh viết dưới đây:

a) I = I + K +1

b) SS = 0.5 * SIN (A*3.1416 / 180.)

c) ERR = ABS (X1 - X2)

6. Hai đoạn chương trình sau nhằm tính trị số trung bình A của ba số nguyên $i_1 = 1,i_2 = 2,i_3 = 3$ và in kết quả lên màn hình. Hãy thử xem kết quả có đúng không. Nếu thấy sai thì chỉ ra tại sao và khắc phục bằng cách nào?

a) I1 = 1

I2 = 2

I3 = 3

PRINT 4 , (I1 + I2 + I3)/3

4 FORMAT (1X, ' A = ',F4.1)

b) I1 = 1

I2 = 2

I3 = 3

PRINT 2, 1/3*(I1+I2+I3)

2 FORMAT (3X, F4.1)

7. Giả sử các cung địa lý (tám cung) được đánh số hiệu theo qui ước như sau: 1 - bắc; 2 - đông bắc; 3 - đông; 4 - đông nam; 5 - nam; 6 - tây nam; 7 - tây; 8 - tây bắc. Hướng gió quan trắc được bằng 165^{o}. Hãy viết biểu thức Fortran để tính số hiệu cung của hướng gió đó.