BÀI 9

LÀM VIỆC VỚI DÃY SỐ

1. Dãy số và biến mảng

Ví dụ 1. Giả sử chúng ta cần viết chương trình nhập điểm kiểm tra của các học sinh trong một lớp và sau đó in ra màn hình điểm số cao nhất. Vì mỗi biến chỉ có thể lưu một giá trị duy nhất, để có thể nhập điểm và so sánh chúng, ta cần sử dụng nhiều biến, mỗi biến cho một học sinh. Ví dụ như trong Pascal ta cần nhiều câu lệnh khai báo và nhập dữ liệu dạng sau đây, mỗi câu lệnh tương ứng với điểm của một học sinh:

```
Var Diem_1, Diem_2, Diem_3,…: real;
…
Read(Diem_1); Read(Diem_2); Read(Diem_3);…
```

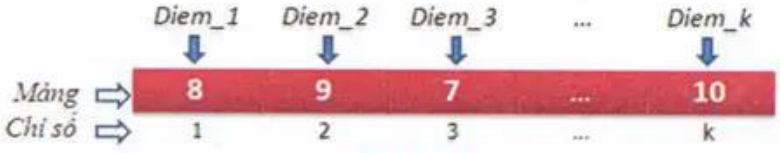
Nếu số học sinh trong lớp càng nhiều thì đoạn khai báo và đọc dữ liệu trong chương trình càng dài. Việc so sánh các điểm đã nhập còn khó khăn hơn. Ta cần nhớ hết tên các biến đã khai báo và rất khó tránh khỏi nhầm lẫn, sai sót.

Giả sử chúng ta có thể lưu nhiều dữ liệu có liên quan với nhau (như *Diem_1, Diem_2, Diem_3,...* ở trên) bằng *một biến duy nhất* và đánh "số thứ tự" cho các giá trị đó, ta có thể sử dụng quy luật tăng hay giảm của "số thứ tự" và một vài câu lệnh lặp để xử lí dữ liệu một cách đơn giản hơn, chẳng hạn:

- Với *i = 1 đến 50*: hãy nhập *Diem_i*;
- Với *i = 1 đến 50*: hãy so sánh *Max* với *Diem_i*;

Để giúp giải quyết các vấn đề trên, hầu hết các ngôn ngữ lập trình đều có một kiểu dữ liệu được gọi là *kiểu mảng*.

Dữ liệu kiểu mảng là một tập hợp hữu hạn các phần tử có thứ tự, mọi phần tử đều có cùng một kiểu dữ liệu, gọi là kiểu của phần tử. Việc sắp thứ tự được thực hiện bằng cách gán cho mỗi phần tử một *chỉ số*:



Hình 40

Trong bài này, chúng ta chỉ xét các mảng có các phần tử kiểu số nguyên hoặc số thực.

Khi khai báo một biến có kiểu dữ liệu là kiểu mảng, biến đó được gọi là *biến mảng*. Có thể nói rằng, khi sử dụng biến mảng, về thực chất chúng ta sắp thứ tự theo chỉ số các biến có *cùng kiểu* dưới một tên duy nhất.

Giá trị của biến mảng là một *mảng*, tức một dãy số (số nguyên, hoặc số thực) có thứ tự, mỗi số là giá trị của biến thành phần tương ứng.

2. Ví dụ về biến mảng

Để làm việc với các dãy số nguyên hay số thực, chúng ta phải khai báo biến mảng có kiểu tương ứng trong phần khai báo của chương trình.

Cách khai báo biến mảng trong các ngôn ngữ lập trình có thể khác nhau, nhưng luôn cần chỉ rõ: tên biến mảng, số lượng phần tử, kiểu dữ liệu chung của các phần tử.

Ví dụ, cách khai báo đơn giản một biến mảng trong ngôn ngữ Pascal như sau:

```
var Chieucao: array[1..50] of real;
var Tuoi: array[21..80] of integer;
```

Với câu lệnh thứ nhất, ta đã khai báo một biến có tên *Chieucao* gồm 50 phần tử, mỗi phần tử là biến có kiểu số thực. Với câu lệnh khai báo thứ hai, ta có biến *Tuoi* gồm 60 phần tử (từ 21 đến 80) có kiểu số nguyên.

Từ hai ví dụ trên, có thể thấy cách khai báo mảng trong Pascal như sau:

Tên mảng : **array** [*<chỉ số đầu>*.. *<chỉ số cuối>*] **of** *<kiểu dữ liệu>*

trong đó *chỉ số đầu* và *chỉ số cuối* là hai số nguyên thoả mãn *chỉ số đầu* ≤ *chỉ số cuối* và *kiểu dữ liệu* có thể là `integer` hoặc `real`.

Ví dụ 2. Tiếp tục với ví dụ 1, thay vì khai báo các biến *Diem_1*, *Diem_2*, *Diem_3*,... để lưu điểm số của các học sinh, ta khai báo biến mảng *Diem* như sau:

```
var Diem: array[1..50] of real;
```

Cách khai báo và sử dụng biến mảng như trên có lợi gì?

Trước hết, có thể thay rất nhiều câu lệnh nhập và in dữ liệu ra màn hình bằng một câu lệnh lặp. Chẳng hạn, ta có thể viết

```
for i:= 1 to 50 do readln(Diem[i]);
```

để nhập điểm của các học sinh. Thay vì phải viết 50 câu lệnh khai báo và 50 câu lệnh nhập, ta chỉ cần viết hai câu lệnh là đủ và kết quả đạt được là như nhau.

Ta còn có thể sử dụng biến mảng một cách rất hiệu quả trong xử lí dữ liệu. Để so sánh điểm của mỗi học sinh với một giá trị nào đó, ta cũng chỉ cần một câu lệnh lặp, chẳng hạn

```
for i:=1 to 50 do
if Diem[i]>8.0 then writeln('Gioi');
```

Điều này giúp tiết kiệm rất nhiều thời gian và công sức viết chương trình.

Hơn nữa, mỗi học sinh có thể có nhiều điểm theo từng môn học: điểm Toán, điểm Văn, điểm Lí,... Để xử lí đồng thời các loại điểm này, ta có thể khai báo nhiều biến mảng:

```
var DiemToan: array[1..50] of real;
var DiemVan: array[1..50] of real;
var DiemLi: array[1..50] of real;
```

hay

```
var DiemToan, DiemVan, DiemLi: array[1..50] of real;
```

Khi đó, ta cũng có thể xử lí điểm thi của *một* học sinh cụ thể (ví dụ như tính điểm trung bình của Lan, tính điểm cao nhất của Châu,...) hoặc tính điểm trung bình của cả lớp,...

DiemLi	8	6	7	6	...	...	...	...
DiemVan	7	8	6	9	...	...	...	...
DiemToan	9	7	8	7	...	...	...	...
Chỉ số	1	2	3	4	...	*i*	...	50

Hình 41

Sau khi một mảng đã được khai báo, chúng ta có thể làm việc với các phần tử của nó như làm việc với một biến thông thường như gán giá trị, đọc giá trị và thực hiện các tính toán với các giá trị đó.

Ví dụ 2 cũng cho thấy rằng, chúng ta gán giá trị, đọc giá trị và tính toán với các giá trị của một phần tử trong biến mảng thông qua chỉ số tương ứng của phần tử đó. Chẳng hạn, trong câu lệnh trên *Diem[i]* là phần tử thứ *i* của biến mảng *Diem*.

Ta có thể gán giá trị cho các phần tử của mảng bằng câu lệnh gán:

```
A[1]:= 5;
A[2]:= 8;
```

hoặc nhập dữ liệu từ bàn phím bằng câu lệnh lặp:

```
for i:= 1 to 5 do readln(a[i]);
```

3. Tìm giá trị lớn nhất và nhỏ nhất của dãy số

Ví dụ 3. Viết chương trình nhập *N* số nguyên từ bàn phím và in ra màn hình số nhỏ nhất và số lớn nhất. *N* cũng được nhập từ bàn phím (xem lại thuật toán trong ví dụ 6, bài 5).

Trước hết ta khai báo biến *N* để nhập số các số nguyên sẽ được nhập vào. Sau đó khai báo *N* biến lưu các số được nhập vào như là các phần tử của một biến mảng *A*. Ngoài ra, cần khai báo một biến *i* làm biến đếm cho các lệnh lặp và biến *Max* để lưu số lớn nhất, *Min* để lưu số nhỏ nhất.

Phần khai báo của chương trình có thể như sau:

```
program MaxMin;
uses crt;
var i, n, Max, Min: integer;
    A: array[1..100] of integer;
```

Phần thân chương trình sẽ tương tự dưới đây:

```
begin
    clrscr;
    write('Hay nhap do dai cua day so, N = '); readln(n);
    writeln('Nhap cac phan tu cua day so:');
    for i:= 1 to n do
          begin
          write('a[',i,'] ='); readln(a[i]);
          end;
    Max:= a[1]; Min:= a[1];
    for i:= 2 to n do
          begin if Max < a[i] then Max:= a[i];
                if Min > a[i] then Min:= a[i]
          end;
    writeln('So lon nhat la Max = ',Max);
    writeln('So nho nhat la Min = ',Min);
    readln
end.
```

Trong chương trình này, chúng ta hãy lưu ý điểm sau: Số tối đa các phần

tử của mảng (còn gọi là *kích thước* của mảng) phải được khai báo bằng một số cụ thể (ở đây là 100, mặc dù số các số nhập vào sau này có thể nhỏ hơn nhiều so với 100).

GHI NHỚ

1. Dữ liệu kiểu mảng là một tập hợp hữu hạn các phần tử có thứ tự và mọi phần tử đều có cùng một kiểu dữ liệu.

2. Việc gán giá trị, nhập giá trị và tính toán với các giá trị của một phần tử trong biến mảng được thực hiện thông qua chỉ số tương ứng của phần tử đó.

3. Sử dụng các biến mảng và câu lệnh lặp giúp cho việc viết chương trình được ngắn gọn và dễ dàng hơn.

Câu hỏi và bài tập

1. Hãy nêu các lợi ích của việc sử dụng biến mảng trong chương trình.

2. Các khai báo biến mảng sau đây trong Pascal đúng hay sai?

a) `var X: Array[10,13] Of Integer;`

b) `var X: Array[5..10.5] Of Real;`

c) `var X: Array[3.4..4.8] Of Integer;`

d) `var X: Array[10..1] Of Integer;`

e) `var X: Array[4..10] Of Real;`

3. "Có thể xem biến mảng là một biến được tạo từ nhiều biến có cùng kiểu, nhưng chỉ dưới một tên duy nhất". Phát biểu đó đúng hay sai?

4. Câu lệnh khai báo biến mảng sau đây máy tính có thực hiện được không?

```
var N: integer;
 A: array[1..N] of real;
```

5. Viết chương trình Pascal sử dụng biến mảng để nhập từ bàn phím các phần tử của một dãy số. Độ dài của dãy cũng được nhập từ bàn phím.