BÀI 8

LẶP VỚI SỐ LẦN CHƯA BIẾT TRƯỚC

1. Các hoạt động lặp với số lần chưa biết trước

Trong bài trước chúng ta đã làm quen với các hoạt động lặp và cách chỉ thị cho máy tính thực hiện các hoạt động lặp với số lần đã được xác định trước. Chẳng hạn, để tính tổng các số nguyên từ 1 đến 100, ta có thể viết câu lệnh lặp để máy tính thực hiện phép cộng 99 lần.

Trong thực tế có nhiều hoạt động được thực hiện lặp đi lặp lại với số lần chưa được biết trước.

Ví dụ 1. Một ngày chủ nhật, bạn Long gọi điện cho Trang. Không có ai nhấc máy. Long quyết định gọi thêm hai lần nữa. Nếu vẫn không có ai nhấc máy thì chắc là không có ai ở nhà. Như vậy Long đã biết trước là mình sẽ lặp lại hoạt động gọi điện thêm hai lần. Một ngày khác, Long quyết định cứ 10 phút gọi điện một lần cho Trang cho đến khi nào có người nhấc máy. Lần này Long sẽ lặp lại hoạt động gọi điện mấy lần? Chưa thể biết trước được, có thể một lần, có thể hai hoặc nhiều hơn nữa. Điều kiện để kết thúc hoạt động lặp đó là *có người nhấc máy.*

Ví dụ 2. Nếu cộng lần lượt n số tự nhiên đầu tiên ($n = 1, 2, 3,...$), ta sẽ được các kết quả $T_1 = 1$, $T_2 = 1 + 2$, $T_3 = 1 + 2 + 3,...$ tăng dần. Cần cộng bao nhiêu số tự nhiên đầu tiên để ta nhận được tổng T_n nhỏ nhất lớn hơn 1000 ? Trong trường hợp này, để quyết định thực hiện phép cộng với số tiếp theo hay dừng, trong từng bước cần phải kiểm tra tổng đã lớn hơn 1000 hay chưa.

Chúng ta hãy tìm hiểu các bước của thuật toán trong ví dụ này một cách cụ thể hơn. Kí hiệu S là tổng cần tìm và ta có thuật toán như sau:

Bước 1. $S \leftarrow 0$, $n \leftarrow 0$.

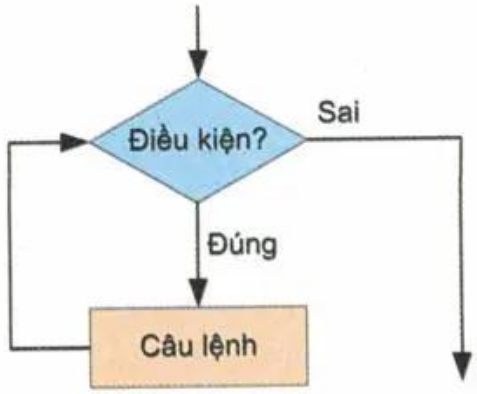
Bước 2. Nếu $S \leq 1000$, $n \leftarrow n + 1$; *Ngược lại,* chuyển tới bước 4.

Bước 3. $S \leftarrow S + n$ và quay lại bước 2.

Bước 4. In kết quả: S và n là số tự nhiên nhỏ nhất sao cho $S > 1000$. Kết thúc thuật toán.

Việc thực hiện phép cộng ở thuật toán trên được lặp lại với số lần chưa biết trước, phụ thuộc vào một *điều kiện* ($S \leq 1000$) và chỉ dừng khi điều kiện đó sai.

Nói chung, việc lặp lại một nhóm hoạt động với số lần chưa xác định trước phụ thuộc vào một *điều kiện* cụ thể có được thoả mãn hay không và có thể được mô tả như hình 39.



Hình 39

Để viết chương trình chỉ dẫn máy tính thực hiện các hoạt động lặp như trong các ví dụ trên, ta có thể sử dụng câu lệnh có dạng *lặp với số lần chưa biết trước*. Nói chung các ngôn ngữ lập trình đều có câu lệnh lặp dạng này.

2. Ví dụ về lệnh lặp với số lần chưa biết trước

Trong Pascal câu lệnh lặp với số lần chưa biết trước có dạng:

while <*điều kiện*> **do** <*câu lệnh*>;

trong đó:

- *điều kiện* thường là một phép so sánh;
- *câu lệnh* có thể là câu lệnh đơn giản hay câu lệnh ghép.

Câu lệnh lặp này được thực hiện như sau:

1. Kiểm tra *điều kiện*.

2. Nếu *điều kiện* SAI, *câu lệnh* sẽ bị bỏ qua và việc thực hiện lệnh lặp kết thúc. Nếu *điều kiện* đúng, thực hiện *câu lệnh* và quay lại bước 1.

Ví dụ 3. Chúng ta biết rằng, nếu n ($n > 0$) càng lớn thì $\frac{1}{n}$ càng nhỏ, nhưng luôn luôn lớn hơn 0.

Với giá trị nào của *n* thì $\frac{1}{n} < 0.005$ hoặc $\frac{1}{n} < 0.003$? Chương trình dưới đây tính số *n* nhỏ nhất để $\frac{1}{n}$ nhỏ hơn một sai số cho trước:

```
uses crt;
var x: real;
    n: integer;
const sai_so=0.003;
begin
    clrscr;
    x:= 1; n:= 1;
    while x >= sai_so do begin n:= n + 1; x:= 1/n end;
    writeln('So n nho nhat de 1/n < ', sai_so:6:4, 'la ',n);
    readln
end.
```

Nếu chạy chương trình này, ta sẽ nhận được kết quả $n = 334$. Thay điều kiện *sai_so = 0.003* lần lượt bằng các điều kiện *sai_so = 0.002* và *sai_so = 0.001*, ta nhận được các kết quả $n = 501$ và $n = 1001$. Có thể kiểm tra các kết quả này bằng một phép chia đơn giản.

Ví dụ 4. Chương trình Pascal dưới đây thể hiện thuật toán tính tổng n số trong ví dụ 2:

```
var S, n: integer;
begin
    S:= 0; n:= 1;
    while S <= 1000 do
        begin  S:= S + n ; n:= n + 1 end;
    writeln('So n nho nhat de tong > 1000 la ', n);
    writeln('Tong dau tien > 1000 la ', S);
    readln
end.
```

Nếu chạy chương trình này ta sẽ nhận được $n = 45$ và tổng đầu tiên lớn hơn 1000 là 1034.

Ví dụ 5. Để viết chương trình tính tổng $T = 1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{100}$ ta có thể sử dụng lệnh lặp với số lần lặp biết trước **for...do**:

```
T:= 0;
for i:= 1 to 100 do T:= T + 1/i;
writeln(T);
```

Nếu sử dụng lệnh lặp **while...do**, đoạn chương trình dưới đây cũng cho cùng một kết quả:

```
T:= 0;
i:= 1;
while i <= 100 do begin T:= T + 1/i; i:= i + 1 end;
writeln(T);
```

Ví dụ này cho thấy rằng chúng ta có thể sử dụng câu lệnh **while...do** thay cho câu lệnh **for...do**.

3. Lặp vô hạn lần – Lỗi lập trình cần tránh

Khi viết chương trình sử dụng cấu trúc lặp cần chú ý tránh tạo nên vòng lặp không bao giờ kết thúc. Chẳng hạn, chương trình dưới đây sẽ lặp lại vô tận:

```
var a:integer;
begin
    a:=5;
    while a<6 do writeln('A');
end.
```

Trong chương trình trên, giá trị của biến *a* luôn luôn bằng 5, điều kiện *a<6* luôn luôn đúng nên lệnh `writeln('A')` luôn được thực hiện.

Do vậy, khi thực hiện vòng lặp, *điều kiện* trong câu lệnh phải được thay đổi để sớm hay muộn giá trị của *điều kiện* được chuyển từ *đúng* sang *sai*. Chỉ như thế chương trình mới không "rơi" vào những "vòng lặp vô tận".

GHI NHỚ

1. Ngoài cấu trúc lặp với số lần lặp biết trước, các ngôn ngữ lập trình còn có các câu lệnh lặp với số lần chưa biết trước.

2. **while...do** là câu lệnh lặp với số lần chưa biết trước trong Pascal.

Câu hỏi và bài tập

1. Nêu một vài ví dụ về hoạt động lặp với số lần chưa biết trước.

2. Hãy phát biểu sự khác biệt giữa câu lệnh lặp với số lần biết trước và câu lệnh lặp với số lần lặp chưa biết trước.

3. Hãy tìm hiểu các thuật toán sau đây và cho biết khi thực hiện thuật toán, máy tính sẽ thực hiện bao nhiêu vòng lặp? Khi kết thúc, giá trị của S bằng bao nhiêu? Viết chương trình Pascal thể hiện các thuật toán đó.

a) Thuật toán 1

Bước 1. $S \leftarrow 10$, $x \leftarrow 0.5$.

Bước 2. Nếu $S \leq 5.2$, chuyển tới bước 4.

Bước 3. $S \leftarrow S - x$ và quay lại bước 2.

Bước 4. Thông báo S và kết thúc thuật toán.

b) Thuật toán 2

Bước 1. $S \leftarrow 10$, $n \leftarrow 0$.

Bước 2. Nếu $S \geq 10$, chuyển tới bước 4.

Bước 3. $n \leftarrow n + 3$, $S \leftarrow S - n$ quay lại bước 2.

Bước 4. Thông báo S và kết thúc thuật toán.

4. Hãy tìm hiểu mỗi đoạn lệnh sau đây và cho biết với đoạn lệnh đó chương trình thực hiện bao nhiêu vòng lặp ? Hãy rút ra nhận xét của em.

```
a) S:= 0; n:= 0;
   while S <= 10 do
        begin n:= n + 1; S:= S + n end;
b) S:= 0; n:= 0;
   while S <= 10 do
        n:= n + 1; S:= S + n;
```

5. Hãy chỉ ra lỗi trong các câu lệnh sau đây:

```
a) X:= 10; while X:= 10 do X:= X + 5;
b) X:= 10; while X = 10 do X = X + 5;
c) S:= 0; n:= 0; while S <= 10 do n:= n + 1; S:= S + n;
```