Huynh Minh Khoa Is Weblog

Cơ bản về thuật toán đệ quy

1. Đệ quy là gì ?

Một đối tượng được gọi là đệ quy nếu nó được mô tả thông qua định nghĩa của chính nó. Nghĩa là, các đối tượng này được định nghĩa một cách quy nạp từ những khái niệm đơn giản nhất cùng dạng với nó. Trong toán học và tin học có rất nhiều đối tượng như thế. VD :

– Số tự nhiên được định nghĩa như sau :

- 0 là một số tự nhiên
- Nếu k là một số tự nhiên thì k+1 cũng là một số tự nhiên

Theo đó, ta sẽ có : 1=0+1 là số tự nhiên, 2=1+1 cũng là một số tự nhiên,….Cứ như vậy ta sẽ định nghĩa được các số tự nhiên khác lớn hơn. Do đó, số tự nhiên là khái niệm mang bản chất đệ quy.

– Định nghĩa giai thừa của n (n!) :

- Khi n=0, ta có 0!=1
- Khi n>0, ta có n!=(n-1)! x n

Như vậy, ta suy ra : 1! = 0! x 1, 2! = 1! x 2,… –> giai thừa cũng là một khái niệm mang tính đệ quy.

2. Bài toán đệ quy

Đó là những bài toán mang bản chất đệ quy. Nghĩa là những bài toán này có thể được phân rã thành các bài toán nhỏ hơn, đơn giản hơn nhưng có cùng dạng với bài toán ban đầu. Những bài toán nhỏ lại được phân rã thành các bài toán nhỏ hơn. Cứ như vậy, việc phân rã chỉ dừng lại khi bài toán con đơn giản đến mức có thể suy ra ngay kết quả mà không cần phải phân rã nữa. Ta phải giải tất cả các bài toán con rồi kết hợp các kết quả đó lại để có được lời giải cho bài toán lớn ban đầu. Cách phân rã bài toán như vậy gọi là "chia để trị" (devide and conquer), là một dạng của đệ quy.

3. Đệ quy trong lập trình

Một hàm được gọi là đệ quy nếu bên trong thân nó có một lời gọi đến chính nó. Nghe có vẻ vô lý nhỉ ? Một hàm làm sao có thể gọi nó mãi được, vì nếu như vậy sẽ sinh ra một vòng lặp vô tận. Nhưng trong thực tế, một hàm đệ quy luôn có điều kiện đừng được gọi là "điểm neo". Khi đạt tới điểm neo, hàm sẽ không gọi chính nó nữa.

Khi được gọi, hàm đệ quy thường được truyền cho một tham số, thường là kích thước của bài toán lớn ban đầu. Sau mỗi lời gọi đệ quy, tham số sẽ nhỏ dần, nhằm phản ánh bài toán đã nhỏ hơn và đơn giản hơn. Khi tham số đạt tới một giá trị cực tiểu (tại điểm neo), hàm sẽ chấm dứt.

Trong lập trình, một bài toán muốn giải quyết bằng đệ quy thì bản thân nó phải là một bài toán đệ quy. Tức là bài toán đó có thể được đưa về bài toán cùng dạng nhưng đơn giản hơn.

4. Một số bài toán đệ quy kinh điển

a. Bài toán tính giai thừa

Cho n là một số tự nhiên (n>=0). Hãy tính giai thừa của n (n!) biết rằng 0!=1 và n!=(n-1)! x n.

Phân tích :

– Theo giả thiết, ta có : n! = (n-1)! x n. Như vậy :

- Để tính n! ta cần phải tính (n-1)!
- Để tính (n-1)! ta phải tính (n-2)!
- …

– Cứ như vậy, cho tới khi gặp trường hợp 0!. Khi đó ta lập tức có được kết quả là 1, không cần phải tính thông qua một kết quả trung gian khác.

Cài đặt trên C# :

Code

View source

```
1. public static long GiaiThua(int n)
2. {
3.     if (n == 0) return 1;   //điểm neo
4.     return n * GiaiThua(n – 1);     //gọi đệ quy
5. }
```

– Ở đây, điểm neo chính là n=0. Sau mỗi lời gọi đệ quy, n sẽ giảm xuống 1 cho đến khi gặp điểm neo.

b. Dãy Fibonaci

Dãy Fibonaci là dãy vô hạn các số tự nhiên. Số Fibonaci thứ n, ký hiệu F(n), được định nghĩa như sau :

- *F(n) = 1, nếu n=1 hoặc n=2*
- *F(n) = F(n-1) + F(n-2), nếu n>=3*

Yêu cầu : tính số fibonaci thứ n với n cho trước.

Phân tích : theo giả thiết

– Với n<3 :="" ta="" suy="" ra="" ngay="" k="" t="" qu="" f="" n="" 1="" span="" style="font-size: 10pt;" data-mce-style="font-size: 10pt;">

– Với n>=3 :

- Để tính F(n) ta phải tính F(n-1) và F(n-2).
- Để tính F(n-1) ta lại phải tính F(n-2) và F(n-3), và để tính F(n-2) ta phải tính F(n-3) và F(n-4).
- …
- Cứ như vậy cho đến khi n=1 và n=2.

Cài đặt trên C# :

Code

```
1. public static long Fibonaci(int n)
2. {
3.     if (n < 3) return 1;    //điểm neo
4.     return Fibonaci(n – 1) + Fibonaci(n – 2);   //gọi đệ quy
5. }
```

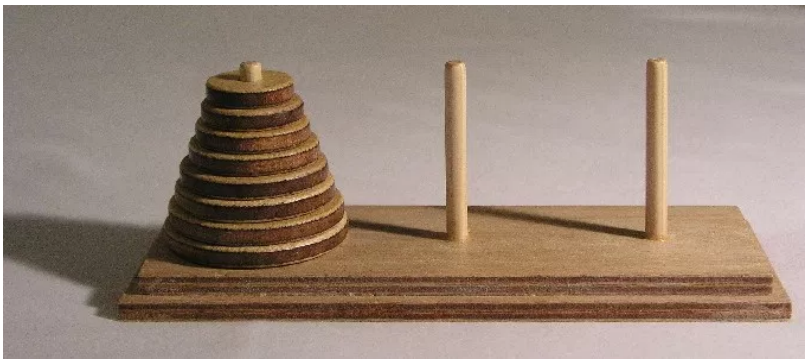
– Dưới đây là sơ đồ đệ quy khi gọi hàm tính F(5) :

Như vậy, bài toán đã được giải quyết một cách thật đơn giản bằng đệ quy.

c. Bài toán "Tháp Hà Nội" (Tower of Ha Noi)

Đây là một bài toán rất nổi tiếng và kinh điển, rất thích hợp để minh họa cho thuật toán đệ quy. Sau đây là nội dung bài toán :

Có 3 chiếc cọc được đánh dấu lần lượt là A, B, C và n chiếc đĩa. Các đĩa này có kích thước khác nhau và mỗi đĩa đều có một lỗ ở giữa để cắm vào cọc. Ban đầu, các đĩa đều nằm ở cọc A, trong đó, đĩa nhỏ luôn nằm trên đĩa lớn hơn.



Yêu cầu : chuyển n đĩa từ cọc A sang cọc đích C với các điều kiện sau :

+ Mỗi lần chỉ chuyển được 1 đĩa

+ Trong quá trình chuyển, đĩa nhỏ phải luôn nằm trên đĩa lớn hơn.

+ Cho phép sử dụng cọc B làm cọc trung gian

Phân tích : ta sẽ xét các trường hợp của n

– Trường hợp đơn giản nhất, n=1, ta chỉ cần chuyển 1 đĩa từ cọc A sang cọc C.

– Nhiều hơn một chút, n=2, ta chuyển đĩa nhỏ nhất sang cọc B, chuyển đĩa còn lại sang cọc C, và cuối cùng chuyển đĩa nhỏ ở cọc B sang cọc C.

– Bây giờ ta xét n đĩa (n>2). Giả sử ta đã có cách chuyển n-1 đĩa từ một cọc sang một cọc khác. Như vậy, để chuyển n đĩa từ cọc nguồn sang cọc đích, ta cần chuyển n-1 đĩa từ cọc nguồn sang cọc trung gian. Sau đó chuyển đĩa lớn nhất từ cọc nguồn sang cọc đích. Cuối cùng, chuyển n-1 từ cọc trung gian về cọc đích.

Cài đặt bằng C# :

Code

View source

```
1. public static void ThapHaNoi(int n, char nguon, char dich, char tgian)
2. {
3.     //điểm neo
4.     if (n == 1) Console.WriteLine("Chuyen 1 dia tu {0} sang {1}", nguon, dich);
5.     else
6.     {
7.         //chuyển n-1 đĩa từ cọc nguồn sang cọc trung gian,
8.         //lấy cọc đích làm cọc phụ
9.         ThapHaNoi(n – 1, nguon, tgian, dich);
10.        //chuyển còn lại từ cọc nguồn sang cọc đích
11.        Console.WriteLine("Chuyen 1 dia tu {0} sang {1}", nguon, dich);
12.        //chuyễn n-1 từ cọc trung gian về cọc đích,
13.        //lấy cọc nguồn làm cọc trung gian
14.        ThapHaNoi(n – 1, tgian, dich, nguon);
15.    }
16. }
```

Hàm trên có nhiệm vụ chuyển n đĩa từ cột **nguon** sang cọc **dich**, sử dụng cọc **tgian** làm cọc phụ. Ở đây ta ký hiệu các cột là A, B, C nên ta sẽ dùng kiểu char.

Như vậy, ta đã tìm hiểu xong những điểm cơ bản về đệ quy. Trên đây chỉ là những bài toán đệ quy đơn giản. Còn rất nhiều bài toán và kỹ thuật khác sử dụng phương pháp đệ quy.

Link Demo: http://csharpviet.freevnn.com/files/demo_project/DeQuyDemo.zip

Leave a Comment »