Đệ Quy

it.kma

1. Đệ quy là gì.

Một hàm được gọi là hàm đệ quy nếu nó chứa lời gọi lại chính nó. Lời gọi lại ấy thường được thực thi trong một trạng thái khác so với trạng thái hiện tại.

Ví dụ hàm tính lũy thừa:

```
func power(a,i)
{
        if(i=1)
                return a;
        else
                return a*power(a,i-1);    //gọi lại
}
```

Ở lần thực thi đầu tiên, hàm kiểm tra i, nếu i=1 thì trả về a. Sau khi trả về, hàm này lập tức được giải phóng. Nếu i>1, hàm trả về a*pow(a,i-1). Tại lệnh trả về này, chương trình sẽ tính toán power(a,i-1) trước, sau đó nhân thêm a rồi trả về kết quả và kết thúc lời gọi hàm. Để tính power(a,i-1), chương trình thực thi một hàm power() khác với đầu vào là a và i-1. Cứ như thế, cho tới khi i=1, chương trình trả về và đóng lần lượt từng hàm đang trong chế độ chờ kết quả từ hàm khác.

Giả sử tính power(a,3), nếu ký hiệu khi thực thi hàm power(a,i) là *"(i"* và khi đóng hàm đó là *"i)"*, ta có thể mô tả các chương trình chạy như sau:

```
(3                  //chạy power(a,3)
(3 (2               //power(a,3) đợi kết quả tính power(a,2)
(3 (2 (1            //power(a,3) và power(a,2) đợi kết quả tính
                    power(a,1)
(3 (2 (1 1)         //Lúc này i đã bằng 1, không có lời gọi đệ quy nào
                    được thực thi nữa, return và đóng hàm power(a,1)
(3 (2 (1 1) 2)      // Hàm power(a,2) đã nhận được kết quả
                    power(a,1), nó nhân với a sau đó trả về và kết thúc
(3 (2 (1 1) 2) 3)   // Hàm power(a,3) đã nhận được kết quả
                    power(a,2), nó nhân với a sau đó trả về và kết thúc
Kết thúc
```

Qua ví dụ trên, ở từng lời gọi lại hàm power(), dù a vẫn vậy, nhưng biến i truyền vào ở mỗi lần gọi là khác nhau. Như vậy trạng thái ở mỗi lần gọi hàm power() là khác nhau.

Tại sao cần dùng đệ quy?

Trước hết, dùng lối diễn giải đệ quy giúp chúng ta mô tả giải thuật một cách trong sáng, ngắn gọn. Việc cài đặt cũng trở nên gọn gàng và đơn giản hơn nhiều.

Bên cạnh đó, việc xuất hiện các bài toán gối nhau, một bài toán lớn có thể được xử lý bằng một hoặc nhiều bài toán con tương tự, ví dụ, tính giai thừa, tính lũy thừa,v.v… Việc cài đặt đệ quy giúp làm nổi bật được khía cạnh này.

Ngoài ra đệ quy tỏ ra đặc biệt hiệu quả trong việc vét cạn, liệt kê tất cả các cấu hình. Một ví dụ đơn giản là bài toán tìm tất cả các nghiệm nguyên của một phương trình bậc n, với n nhập vào từ bàn phím. Nếu với bậc 2, cần dùng 2 vòng for để duyệt hết các trường hợp, thì với bậc n, không thể nào dùng n vòng for được. Điều này dẫn tới cần một phương pháp vét cạn linh động mà chỉ có đệ quy mới giải quyết được.

2. Kỹ thuật đệ quy

3. Đệ quy quay lui

4. Một số bài toán đệ quy điển hình