CẤU TRÚC DỮ LIỆU & GIẢI THUẬT 1

ĐỆ QUY

Nội dung

1. Khái niệm đệ quy
2. Đặc điểm của hàm đệ quy
3. Điều kiện viết đệ quy
4. Khi nào không nên sử dụng đệ quy
5. Bài toán Tháp Hà Nội

Đệ quy

- Một đối tượng được gọi là đệ quy nếu nó hoặc một phần của nó được định nghĩa thông qua khái niệm về chính nó.
- Ví dụ: Định nghĩa số tự nhiên:
  - 0 là một số tự nhiên.
  - n là số tự nhiên nếu n - 1 là số tự nhiên

Giải thuật đệ quy

- Nếu lời giải của bài toán T được thực hiện bởi lời giải của một bài toán T', có dạng như T thì đó là một lời giải đệ quy. Giải thuật chứa lời giải đệ quy được gọi là giải thuật đệ quy (T'<T).
- Ví dụ: Tính n!
  - Nếu n=0, 0!=1.
  - Nếu n>0, n!=n.(n-1)!

Đặc điểm của hàm đệ quy

- Trong hàm đệ quy có lời gọi đến chính nó.
- Mỗi lần có lời gọi thì kích thước của bài toán được thu nhỏ hơn trước.
- Vấn đề nhỏ hơn này, đến một lúc nào đó sẽ đơn giản tới mức chương trình có thể tự giải quyết được mà không cần gọi tới chính nó nữa.
- Đệ quy gồm:
  - Đệ quy trực tiếp (hàm chứa lời gọi đến chính nó): **func1** calls **func1**.
  - Đệ quy gián tiếp (hàm chứa lời gọi đến hàm khác mà hàm này lại chứa lời gọi đến chính nó): **func1** calls **func2**, **func2** calls **func1**.

Ưu, nhược điểm của đệ quy

Ưu điểm:

- Chương trình trong sáng, dễ hiểu (Tùy từng trường hợp).
- Có thể thực hiện một số lượng lớn các thao tác tính toán thông qua 1 đoạn chương trình ngắn gọn.
- Định nghĩa một tập hợp vô hạn các đối tượng thông qua một số hữu hạn lời phát biểu.

Nhược điểm:

- Tốn nhiều dung lượng
- Chậm

Điều kiện để có thể viết đệ quy

- Vấn đề cần xử lý phải được giải quyết một cách đệ quy.
- Ngôn ngữ dùng để viết chương trình phải hỗ trợ đệ quy (ngôn ngữ lập trình có hỗ trợ hàm hoặc thủ tục).
- Hạn chế việc khai báo các biến, hàng trong hàm đệ quy nếu không cần thiết.

Khi nào không nên sử dụng đệ quy

- Chương trình có thể viết dưới dạng lặp hoặc cấu trúc lệnh khác thì không nên sử dụng đệ quy.
- Xét bài toán tính các phần tử của dãy Fibonacci. Dãy Fibonacci được định nghĩa như sau:
  - F(0)=0
  - F(1)=1
  - Với n>1: F(n)=F(n-1)+F(n-2)
- Hàm đệ quy để tính dãy Fibonacci được viết như sau:

```
int  F(int n)
{
        if (n==0) return 0;
        if (n==1) return 1;
        return F (n-1)+F (n-2);
}
```

Khi nào không nên sử dụng đệ quy

- Kết quả thực hiện chương trình không có gì sai. Tuy nhiên, lời gọi đệ quy F(n) sẽ dẫn tới 2 lời gọi đệ quy khác ứng với n-1 và n-2. Hai lời gọi này lại gây ra 4 lời gọi nữa…, cứ như vậy lời gọi đệ quy sẽ tăng theo cấp số mũ.
- Điều này rõ ràng không hiệu quả vì trong số các lời gọi đệ quy đó có rất nhiều lời gọi trùng nhau.
- Ví dụ lời gọi đệ quy F(5) sẽ dẫn đến 2 lời gọi F(4) và F(3). Lời gọi F(4) sẽ gọi F(3) và F(2). Ngay chỗ này ta đã thấy có 2 lời gọi F(3) được thực hiện.

Khi nào không nên sử dụng đệ quy

- Sử dụng vòng lặp để tính giá trị các phần tử của dãy Fibonacci:
- Khai báo một mảng F các số tự nhiên để chứa các số Fibonacci. Vòng lặp để tính và gán các số này vào mảng rất đơn giản:

```
F[0]=0;
F[1]=1;
for (i=2; i<n-1; i++)
    F[i] = F[i-1] + F[i-2];
```

- Với vòng lặp này, mỗi số F(n) chỉ được tính 1 lần thay vì được tính toán chồng chéo như trên.

Bài toán Tháp Hà Nội

- Có 3 chiếc cọc A, B, C và một bộ n đĩa. Các đĩa có kích thước khác nhau. Ban đầu, tất cả các đĩa đều nằm trên cọc A, đĩa nhỏ hơn bao giờ cũng nằm trên đĩa lớn hơn.
- Yêu cầu chuyển n đĩa từ cọc A sang cọc C (có thể sử dụng cọc trung gian B), với điều kiện:
  - Mỗi lần chuyển 1 đĩa.
  - Đĩa có kích thước nhỏ luôn luôn nằm trên đĩa có kích thước lớn hơn.

Bài toán Tháp Hà Nội

- Với n=1, chuyển trực tiếp đĩa 1 từ cọc A sang cọc C.
- Với n=2, có thể thực hiện như sau:
  - Chuyển đĩa nhỏ từ cọc A sang cọc trung gian B.
  - Chuyển đĩa lớn từ cọc A sang cọc đích B.
  - Cuối cùng, chuyển đĩa nhỏ từ cọc trung gian B sang cọc đích C.
- Như vậy, cả 2 đĩa đã được chuyển sang cọc đích C và không có trường hợp nào đĩa lớn nằm trên đĩa nhỏ.

Bài toán Tháp Hà Nội

- Với n>2, giả sử ta đã có cách chuyển n-1 đĩa, ta thực hiện như sau:
  - Lấy cọc đích C làm cọc trung gian để chuyển n-1 đĩa bên trên sang cọc trung gian B.
  - Chuyển đĩa dưới cùng (đĩa n) sang cọc đích C.
  - Lấy cọc A làm cọc trung gian để chuyển n-1 đĩa từ cọc trung gian B sang cọc đích C.

Bài toán Tháp Hà Nội

```
#include <iostream.h>
#include <stdio.h>
void chuyen(int n, char a, char c);
void ThapHaNoi(int n, char a, char c, char b);
int main()
{
      int sodia;
      cout<<"Nhap so dia: ";
      cin>>sodia;
      ThapHaNoi(sodia,'A','C','B');
}
```

Bài toán Tháp Hà Nội

```
void chuyen(int n, char a, char c)
{
      cout<<"\nChuyen dia thu "<<n
          <<" tu coc "<<a<<" sang coc "<<c;
}
void ThapHaNoi(int n, char a, char c, char b)
{
      if(n==1)
            chuyen(1,a,c);
      else
      {
            ThapHaNoi(n-1,a,b,c);
            chuyen(n,a,c);
            ThapHaNoi(n-1,b,c,a);
      }
}
```

Bài tập về đệ quy

1. Tính S(n) = 1 + 2 + 3 + ... + n
2. Tính S(n) = 1 + 1/3 + 1/5 + ... + 1/(2n+1)
3. Tính S(n) = 1/2 + 2/3 + 3/4 + ... + n/(n+1)
4. Tính m^n
5. Tìm ước chung lớn nhất của 2 số.