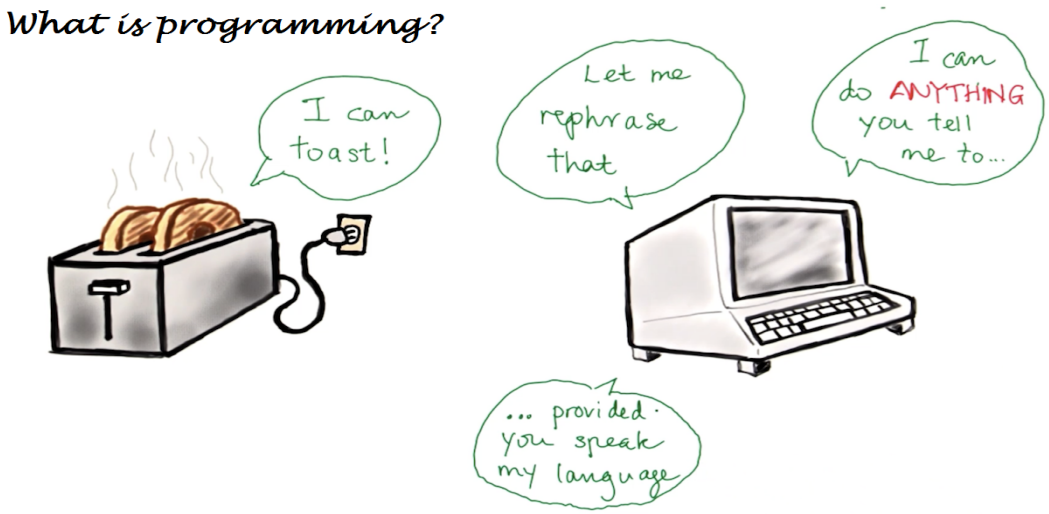


Ôn tập – Hàm đệ quy

CT101 – Lập trình căn bản

Khoa CNTT&TT – ĐHCT

Giới thiệu

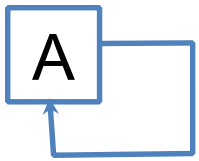
- Để lập trình tính ra n! ta phải chạy qua vòng lặp để tính: n! = n*(n-1)*(n-2)*…*1
- Câu hỏi đặt ra là: Có cách nào để ta biểu diễn cách tính n! một cách toán học và gọn gàng hơn không?
- Nghĩa là ta chỉ cần tính n! = n*(n-1)! với trường hợp đơn giản nhất là 0! = 1. Trong trường hợp này, ta sử dụng hàm đệ quy!
- Nếu sử dụng hàm đệ quy thì thông thường thời gian ta bỏ ra để giải quyết một bài toàn là ngắn hơn so với việc dùng các vòng lặp đơn thuần. Tuy nhiên thời gian chạy là không ngắn hơn, thậm chí là dài hơn nhiều!

Nội dung

- Hàm đệ quy là gì?
- Viết hàm đệ quy như thế nào?
- Việc thực thi hàm đệ quy diễn ra như thế nào?

Hàm đệ quy (Recursive function)

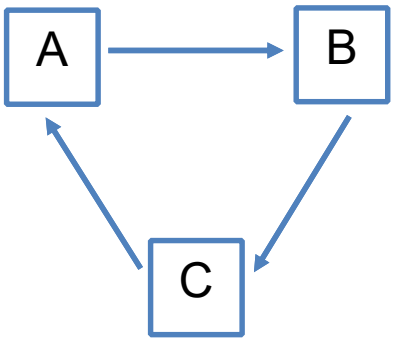
- Hàm đệ quy là hàm mà nó gọi chính nó hoặc ở trong một chu trình xoay vòng các lời gọi hàm.
- Nếu một hàm gọi chính nó, ta gọi là hàm đệ quy tức thì (immediate recursion).




```
void A(){
    A();
    return;
}
```

Hàm đệ quy (Recursive function)

- Trường hợp một hàm ở trong một chu trình các lời gọi hàm:




```
void C(){
    A();
    return;
}

void B(){
    C();
    return;
}

void A(){
    B();
    return;
}
```

Hàm đệ quy (Recursive function)

- Để giải quyết một bài toán, việc sử dụng hàm đệ quy là phương thức không hiệu quả - xét về thời gian chạy.
- Tuy nhiên việc nghiên cứu về hàm đệ quy là thú vị. Ở đây ta chỉ nghiên cứu về hàm đệ quy tức thời vì nó cũng đủ gây rất nhiều khó khăn trong quá trình lập giải thuật và cài đặt chương trình.

Viết hàm đệ quy

- Nhìn chung, một hàm đệ quy có khuôn dạng sau:

```
ReturnType Function( tham số thích hợp ) {
    if trường hợp đơn giản
        return giá trị đơn giản;   // trường hợp cơ sở, điều kiện dừng
    else
        gọi hàm với phiên bản đơn giản hơn của bài toán;
}
```

Ví dụ: tính n!

- Tính n! (n>=0)

  Cách 1 – không đệ quy:

  n! = 1 với n=0

  n! = 1 * 2 * 3 * ... * n với n>0

  Cách 2 – đệ quy:

  n! = 1 với n=0

  n! = n * (n-1)! với n>0

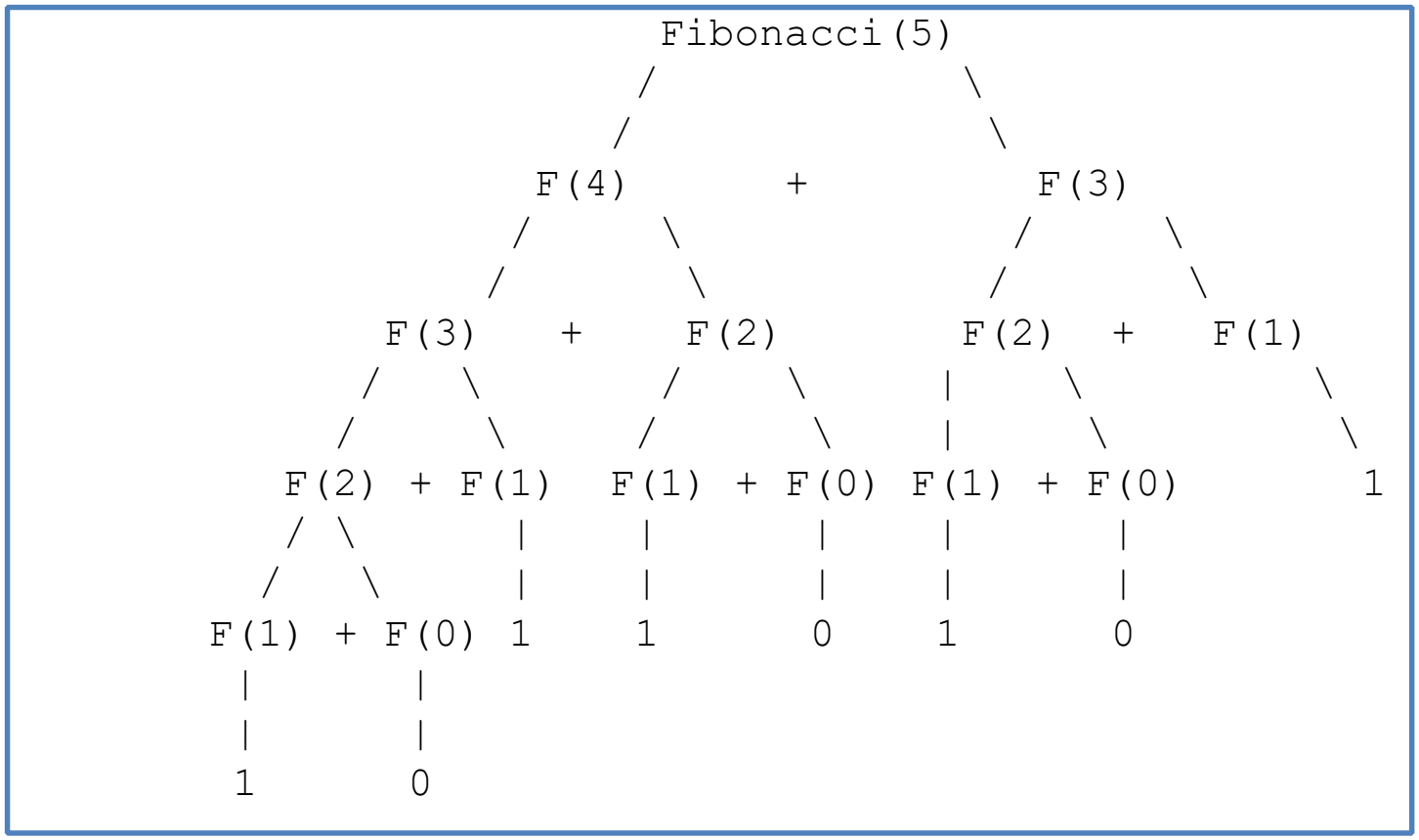
Duyệt qua trình tự thực thi đệ quy

- Cho hàm tính số Fibonacci như sau:

```
int Fibonacci(int x) {
    if (x == 0) return 0;  // Stopping conditions
    if (x == 1) return 1;
    return Fibonacci(x - 1) + Fibonacci(x - 2);
}
```

Duyệt qua trình tự thực thi đệ quy

- Nếu n=5 thì dãy các lời gọi đệ quy sau sẽ được thực thi:



Tổng kết

- Ta dùng hàm đệ quy nhằm mục tiêu viết chương trình rõ ràng hơn với thời gian lập trình ít hơn. Tuy nhiên dung lượng bộ nhớ và thời gian chạy lại cao hơn phương pháp dùng vòng lặp!
- Nhớ rằng, bất kỳ hàm đệ quy nào cũng phải có trường hợp cơ sở và hàm đệ quy phải diễn tiến về trường hợp cơ sở này.

CT101 – Lập trình căn bản

Khoa CNTT&TT – ĐHCT