Bài tập thực hành số 3

ĐỆ QUY – RECURSION

I. Lý thuyết

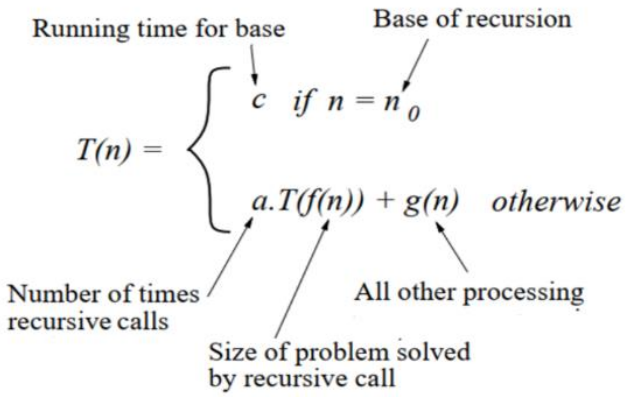
1. Lược đồ giải thuật đệ quy

Lược đồ giải thuật đệ quy là một cách biểu diễn giải thuật đệ quy bằng một đồ thị. Nó thể hiện cách các hàm đệ quy gọi đến chính nó hoặc các hàm đệ quy khác cho đến khi điều kiện dừng được đáp ứng.

```
Recursive_Algorithm(p_n) ≡ //p_n - parameter(s)
    if (p_n=p_0)    //base case
        <statements;>
    else    //general case
        <statements;>
        Recursive_Algorithm(p_n-1);
        <statements;>
    endif
End.
```

$P_n <= P_{n-1} <= P_{n-2} <= \ldots. <= P_0$

2. Độ phức tạp của thuật toán đệ quy (Công thức và ý nghĩa của từng tham số)



3. Phương pháp giải công thức truy hồi (Nêu rõ phương pháp, ví dụ và cách giải ví dụ đấy dựa trên các phương pháp đã nêu)

Phương pháp giải công thức truy hồi (Recursion) là một phương pháp được sử dụng để giải các bài toán phức tạp bằng cách chia chúng thành các bài toán con nhỏ hơn có cùng cấu trúc với bài toán ban đầu. Các bài toán con này sẽ được giải bằng cách sử dụng lại công thức truy hồi của bài toán ban đầu.

VD: Tìm số Fibonacci thứ n trong dãy Fibonacci. Số Fibonacci thứ n được định nghĩa bằng cách tổng hai số Fibonacci trước đó.

Nếu F(n) là số Fibonacci thứ n,

Thì F(n) = F(n-1) + F(n-2) với F(0) = 0 và F(1) = 1.

4. Khử đệ quy và một số dạng khử đệ quy đã được học

Khử đệ quy một thuật toán đệ quy là thay thế thuật toán đệ quybằng một thuật toán tương đương nhưng bỏ đi các lời gọi đệ quy và thay vào đó bằng việc sử dụng các vòng lặp cùng với sự hỗ trợ của ngăn xếp dữ liệu (stack)

+ Dạng ED[AP]
+ Dạng ED[APB]
+ Dạng ED[APBP]

II. Giải công thức đệ quy để xác định độ phức tạp của thuật toán

a. $T(n) = T(n-2) + 2$

TH1: n lẻ

$$T(n) = T(n-2) + 2$$
$$= T(n-4) + 4$$
$$= \ldots = T(n-(n-1)) + 2 + \ldots + 2$$
$$= T(1) + 2\left(\frac{n-1}{2}\right) = n - 1 + 1 = O(n)$$

TH 2: n chẵn

$$T(n) = T(n-2) + 2 = T(n-4) + 2 + 2$$
$$= T(n-n) + 2 + \ldots + 2$$
$$= T(0) + 2\left(\frac{n}{2}\right) = n + 1 = O(n)$$

b.

$$T(n) = a^{\log_b n} T(n/b^{\log_b n}) + \ldots + a^2 (n/b^2)^c + a(n/b)^c + n^c$$
$$= n^{\log_b a} + n^c \sum_{j=0}^{\log_b n - 1} (a/b^d)^j$$

TH1: $a < b^d$

$$T(n) < n^{\log_b a} + n^c \frac{1}{1 - (a/b^c)} = O(n^c)$$

TH2: $a = b^d$

$$T(n) = n^{\log_b a} + n^c \log_b n = O(n^c \log_b n)$$

TH 3: $a > b^d$

$$T(n) < n^{\log_b a} + n^c (a/b^c)^{\log_b n} \frac{1}{(a/b^d) - 1}$$
$$= n^{\log_b a} \left(1 + \frac{1}{(a/b^c) - 1}\right) = O(n^{\log_b a})$$

III. Viết chương trình cho bài toán : Code trong Foder "Coding"