Họ và tên: Nguyễn Hồ Đạt

Mã sinh viên: 20IT416

Bài tập 1:

using *System*;

class Program

{

  static *int* Fibonacci(*int* *n*)

  {

    if (*n* <= 1)

      return *n*;

    return Fibonacci(*n* - 1) + Fibonacci(*n* - 2);

  }

  static *void* Main()

  {

*int* n = 6;

*int* result = Fibonacci(n);

    Console.WriteLine($"Số Fibonacci thứ {n} là: {result}");

  }

}

Bài tập 2:

Để chứng minh rằng độ phức tạp của thuật toán là Ω(2^(n/2)), ta cần chứng minh rằng tổng số phép cộng thực hiện trong thuật toán ít nhất là c \* 2^(n/2) với một hằng số dương c và đủ lớn n0.

Giả sử chúng ta có một hằng số dương c và một số nguyên dương n0, và ta cần chứng minh rằng tổng số phép cộng thực hiện trong thuật toán ít nhất là c \* 2^(n/2) với n >= n0.

Để chứng minh điều này, ta sẽ sử dụng phương pháp quy nạp (induction).

Bước 1: Ta chứng minh cho trường hợp cơ sở (base case).

Khi n = n0, ta chứng minh rằng tổng số phép cộng thực hiện trong thuật toán ít nhất là c \* 2^(n/2):

C(n0) = c \* 2^(n0/2)

Bước 2: Ta giả định rằng điều phải chứng minh đúng cho mọi số n >= n0.

Giả sử rằng đúng cho một số k, tức là C(k) >= c \* 2^(k/2).

Bước 3: Ta chứng minh rằng nếu điều phải chứng minh đúng cho k, thì nó cũng đúng cho k+1.

Khi n = k+1, ta có:

C(k+1) = C(k) + C(k-1) (do thuật toán Fibonacci tính bằng cách gọi đệ quy)

>= c \* 2^(k/2) + c \* 2^((k-1)/2) (giả định quy nạp)

= c \* (2^(k/2) + 2^((k-1)/2))

= c \* (2^(k/2) + 2^(k/2 - 1))

= c \* 2^(k/2) \* (1 + 1/2)

= c \* 2^(k/2) \* (3/2)

= c \* 2^(k/2 + 1) \* (3/2)

= c \* 2^((k+1)/2) \* (3/2)

>= c \* 2^((k+1)/2)

Vậy ta đã chứng minh được điều phải chứng minh đúng cho k+1.

Kết luận: Theo nguyên lý quy nạp, ta đã chứng minh được rằng độ phức tạp của thuật toán Fibonacci thông qua số phép cộng là Ω(2^(n/2)).