$F_n = F_{n-1} + F_{n-2}$, $n > 4$

basis step, $n = 5$

$L.S. = F_5 = 5$

$R.S = (4/3)^5 = 4.21$

$\therefore F_5 > (4/3)^5$

induction step,

for $n = k$, $F_k > (4/3)^k$

for $n = k+1$,

$$F_{k+1} = F_k + F_{k-1}$$

$$> (4/3)^k + (4/3)^{k-1}$$

$$= (4/3)^k \left[1 + \frac{3}{4}\right]$$

$$= (7/4)(4/3)^k$$

$$> (4/3)(4/3)^k$$

$$= (4/3)^{k+1}$$

$\therefore F_n > (4/3)^n$

1. Recursive Algorithm for fibonacci

Algorithm fibo(n):

Input: a natural number n

Output: Fibonacci number at n^{th} position

```
if n <= 1 then
    return n
return fibo(n-1) + fibo(n-2)
```

Analysis:

$$T(0) = 2$$
$$T(1) = 2$$
$$T(2) = T(0) + T(1) + 2$$
$$T(3) = T(1) + T(2) + 2$$
$$\vdots$$
$$T(k) = T(k-1) + T(k-2) + 2$$

If we look at the recursion tree, we can intuitively draw the following conclusion,

$$T(k-1) = O(2^{k-1})$$

$$\therefore T(n) = O(2^{n-1}) + O(2^{n-2}) + O(1)$$

$$= O(2^n)$$

Therefore, recursive algorithm for fibonacci is exponentially slow. The tight bound for the algorithm is also not polynomial - as $\Theta(\varphi^n)$ where φ is the golden ratio (≈1.618).