Time complexity

Recursive

Στο iterative
το $O_n = n$
είναι n φορές που θα εκτελέσει το for

```
Fib(n) {
  • IF n ≤ 1            — 1
      └→ return
  • Else.
      └→ return [Fib(n-1) + Fib(n-2)]     — 1  1  1
```

$T(n) = T(n-1) + T(n-2) + 4$

$T(0) = T(1) = 1$
$\downarrow \quad \downarrow$
$T(n-2) = T(n-1) = X$

$T(n) = 2X + C$
$= 2(T(n-2)) + C$
$= (T(n-2) + T(n-1)) + C$
$= 2X + C$
↳ $T(n-2)$

$n \to 2[n-2$ — δυο φορές το περιορίζει + C

$T(n) = 2T(n-2) + C$ ($n-2 \to 2[n-4]$)
$= 2\{2T(n-4) + C\} + C$
$= 4T(n-4) + 2C + C$
$= 4T(n-4) + 3C$
$= 4(2T(n-6) + C) + 3C$
$= 8T(n-6) + 4C + 3C$
$= 8(T(n-6)) + 7C$

(μαθηματικά) ⟹

$T(n) = 2^k (T(n-2k)) + (2^k - 1) \cdot C$ $\quad k = \frac{n}{2}$

Αν $n - k = 0 \to k = n$

$T(n) = 2^n (T(0)) + (2^n - 1) \cdot C$

⟸

$T(n) = 2^n$

$O_n = 2^n$