1、

高度

def moveTower(n, A, B, C)

from to with

f(n-1) ← moveTower(n-1, A, C, B)

1 moveDisk(A, C)

f(n-1) moveTower(n-1, C, B, A)

$$f(n) = f(n-1) + 1 + f(n-1)$$

$$= 2f(n-1) + 1$$

$$= 2[f(n-2) + 1 + f(n-2)] + 1$$

$$= 4[f(n-3) + 1 + f(n-3)] + 1 + 2$$

$$= 8f(n-3) + 1 + 2 + 4 \cdots$$

$$= 1 + 2 + 4 + \cdots$$

$$= 2^0 + 2^1 + 2^2 + \cdots 2^{n-1} \quad (\text{等比})$$

$$= \frac{2^0 \times (2^n - 1)}{2 - 1}$$

$$= 2^n - 1$$

2、

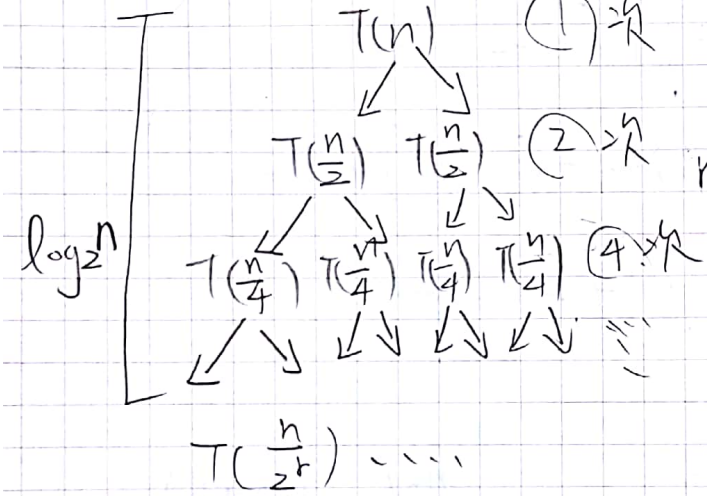
$n \to \frac{n}{2} \to \frac{n}{4} \to \frac{n}{8}$ If $n \le 1$ Stop

$T(n) = 3 \times T(\frac{n}{2}) + 8 \times T(\frac{n}{2})$ (+1)

$T(n) = 3 \times (3 \times T(\frac{n}{4}) + 8 \times T(\frac{n}{4})) + 8 \times (3 \times T(\frac{n}{4}) + 8 \times T(\frac{n}{4}))$

$= 3 \times (3 \times (3 \times T(\frac{n}{8}) + 8T(\frac{n}{8}) + 8 \times (3 \times T(\frac{n}{8}) + 8 \times T(\frac{n}{8}))$

$+ 8 \times 3 \times (3 \times T(\frac{n}{8}) + 8(\frac{n}{8})) + 8 \times 8 \times (3 \times T(\frac{n}{8}) + 8 \times T(\frac{n}{8}))$



$n \times (\frac{1}{2})^h \le 1$

$\log_2 n \le -h \times \log_2 2^{-1}$

$\log_2 n \le h$

Total cost = 1 + 2 + 4 ...

$$\Rightarrow \begin{cases} T(n) = 2 \times T(n/2) + \Theta(1) \\ T(1) = 1 \end{cases}$$

by Master Theorem $T(n) = \Theta(n)$ ※

3.

```
def fibonacci(n):
    int a = stack{}
    if n=0:
        a.push(0)
        return a
    if n=1:
        a.push(0)
        a.push(1)
        return a.
    else =
        int temp=0
        int seq=0
        seq = fibonacci(n-1)
        temp= seq.pop()
        seq.push(temp)
        seq.push(temp)
        seq.push(seq.pop()) + fibonacci(n-2).pop()
        return seq.
```

4. def fib(n):

    fib = list()

    fib[0] = 0

    fib[1] = 1

    for i in range(2, n+1)

        fib[i] = fib(i-1) + fib(i-2)

    return fib[i]

5. 由講義知 recusive 之 complexity 是 $\Theta\left(\left(\frac{1+\sqrt{5}}{2}\right)^n\right)$

而由 4 可知 dp 是用線性完成記錄表

故為 O(n)

Recusive $\Theta\left(\left(\frac{1+\sqrt{5}}{2}\right)^n\right)$

Dynamic Programming O(n)