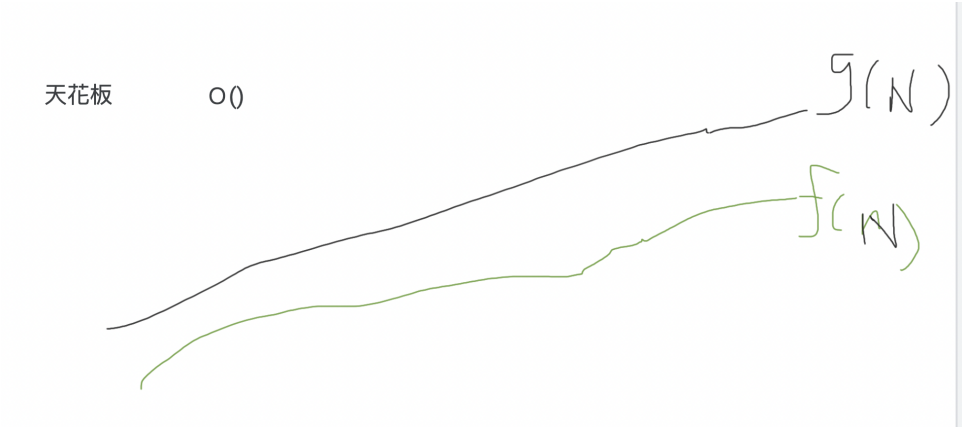
Phill-CPP-0823

algorithm

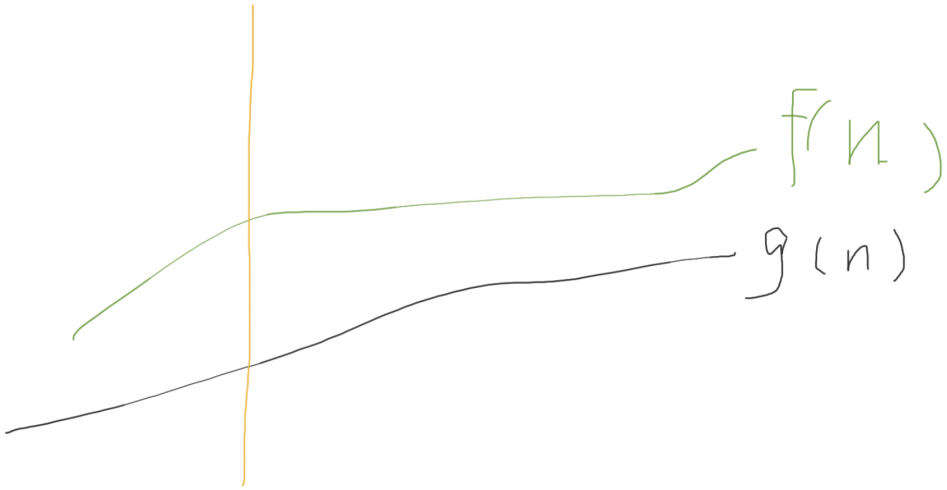
- able to be evaluation → performance evaluation → 1. time 2. space
- finite steps(有限指令)
- sequence (instruction sequence) 正確而有序的步驟指令解決問題
- A → P
  - P ‘s instance → A → solved correctly
  - deterministic (確定性)
  - halting problem → 有限步驟停機 → finite loop constraint
  - for each instance → A → solved
- performance
  - time complexity (時間複雜度)
  - 指令 → 分類基本指令
  - 運算次數統計法 → 1. 困難統計 2. non-deterministic
  - 運算次數統計 ←→ 輸入資料的量級
    - 不看運算指令數 → analysis function → f(x) → X:input size → y = time
    - f(100), f(100000)
  - time complexity → 輸入資料量級的函數
- 複雜度原則,符號
  - 複雜度上限(upper bound) → O → big O 天花板



f(n) = O(g(n))

→ C, N_0 → $\|f(n)\| \leq c\|g(n)\|, n \geq N_0$

- 複雜度下限(lower bound) → Ω → omega 地板



$f(n) = \Omega(g(n))$

→ C, N_0 → $\|f(n)\| \geq c\|g(n)\|, n \geq N_0$

- 複雜度等級
  - O(1) → 常數複雜度 → $g(n) = c \cdot 1$ → 神級演算法
  - $O(log_2 n)$ → 對數複雜度 → 蠻好的
  - O(n) → 線性複雜度 → reasonable
  - O(n log n) → 次線性複雜度
  - $O(n^2)$ → 平方複雜度 → 100 (10000 sec) → 100萬 (100萬*100 萬秒)

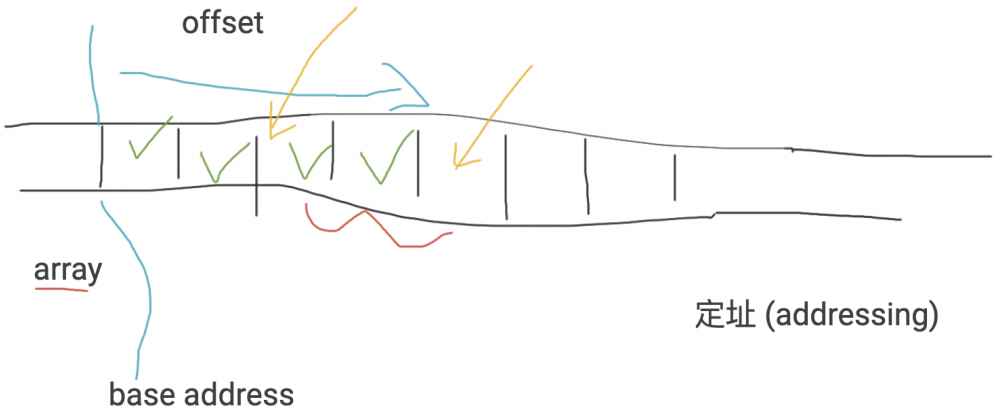
```
for (int i=0; i<=n; i++)
  for (int j=0; j<=m; j++){
      cout << i<< j<<endl;
  }
```

- $O(n^3)$
- $O(2^n)$ → 指數複雜度
- O(n!) → 階乘複雜度

Data Structure

Linear List 線性序列

array



- 取值 O(1)
- 新增 插入 -> O(n)
- 刪除 O(n)
- 搜尋 O(n)

del a[2]
a[2] <- a[3]