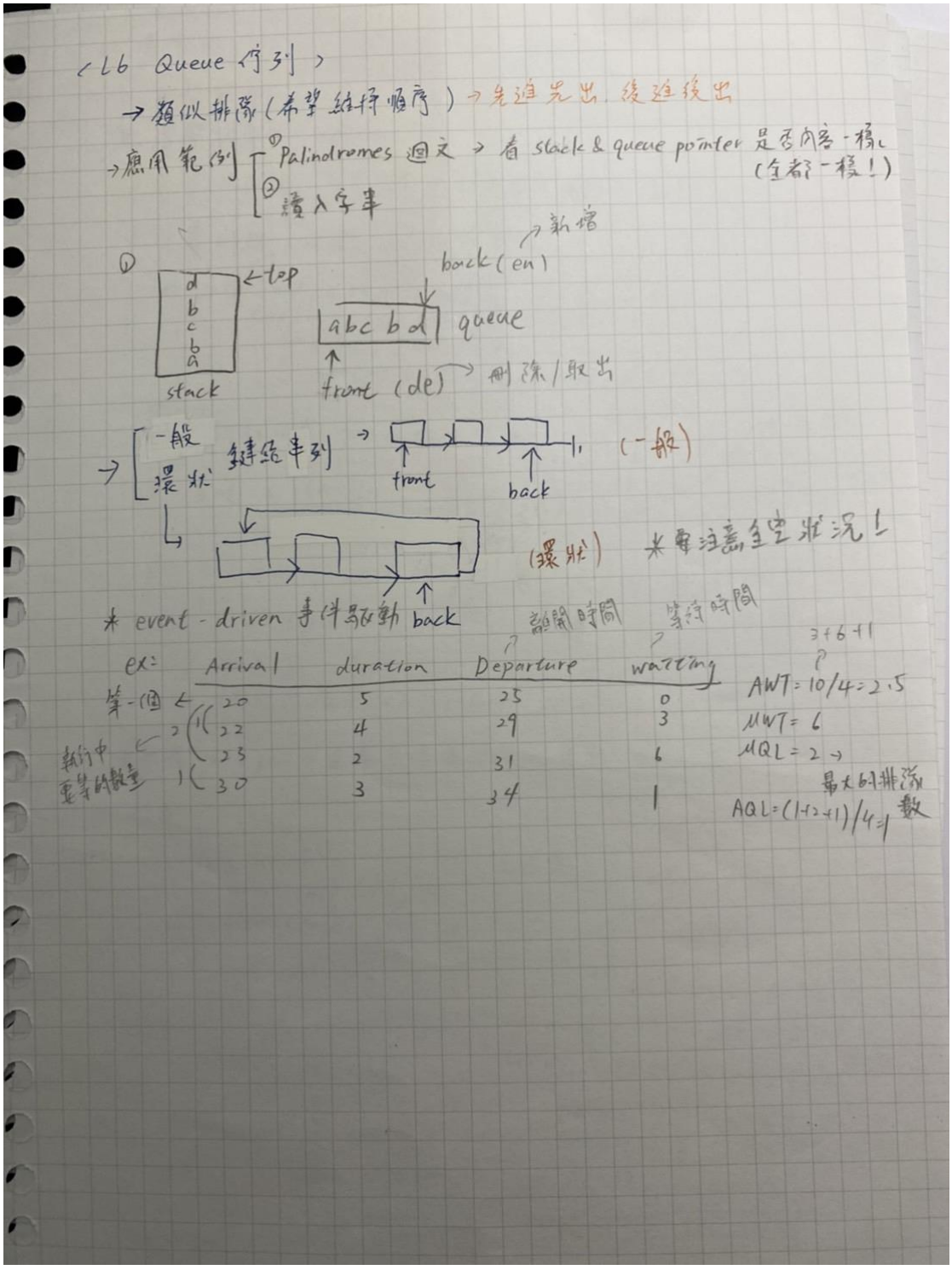
資結筆記 L6-L8

10826223_資工三甲_廖悅婷



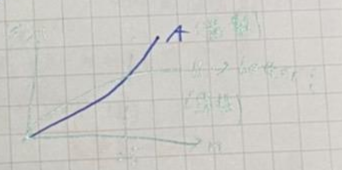
〈4〉演算法〉

- 時間效率
- 空間效率

1) 硬件 2) 電腦 3) 資料 → 和演算法有關的 3 件事

* 計算時間、次數 // Growth Rate

ex:
```
for(a=1; a<=n; a++)
  for(b=1; b<=a; b++)
    for(c=1; c<=5; c++)
      cout << a << b << c << endl;
```



⇒ $\Sigma(t \times 5 \times a)$, for a 從 1 到 n

⇒ $5 \times t \times \frac{(n+1)n}{2} = 2.5t(n^2+n)$

如一程式要執行 $0.2n^2$ 的時間(次數)（→ 演算法 A），而 k f(n) 可以涵蓋之（k → 常數）

⇒ $0.2n^2 \le kf(n)$

⇒ 稱 A 為 order f(n)（大小位階）⇒ $O(f(n))$

// 如有 $n^3 + 3n$ → 通常忽略低位階 or 常數

ex: $(n-1)+(n-2)+\cdots+1$

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

$= 0.5n^2 - 0.5n$

⇒ $O(n^2)$

* 搜尋

sequential search:

- worst case: n 次 → total 資料數
- average case: $\frac{1+n}{2}$ 次
- best case: 1 次

binary search:

- worst: $\log_2 n$

ex: $2.5n^2 - 2.5n$, O(?)

when $n \ge n_0$, $(2.5n^2 - 2.5n) \le k \times f(n)$

when $n \ge 10$, $(2.5n^2 - 2.5n) \le 1 \times n^{10}$

$n \ge 0$, ⋯⋯ $\le 3 \times n^2$

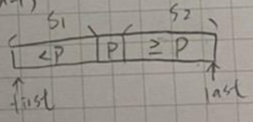
stable sort	unstable sort
bubble	selection → worst $O(n-1)$
insertion	quick → 類似 merge
merge	heap
radix	

stable sort ⇒ 相同的值 維持不變的排序

merge(Data, first, mid, last)

merge: 先分組 // 排序 → 再合併 // 排序 ⇒ level: $\log_2 n$ ⇒ $O(n \times \log_2 n)$

radix → 不做比較，每個值只看自己，從每個值取一部分去分



Bubbe sort [best : O(n)
worst : $O(n^2)$

select sort [best : O(n)
選最大/小
來排列 worst : $O(n^2)$

Insertion sort [best : O(n)
一張張劃分
sort / unsort worst : $O(n^2)$
把小到往 sort
處塞

Mergesort [best : $O(n\log_2 n)$
每次都分一半
排序完再
合併排序 worst : $O(n\log_2 n)$

Quick sort [best : $O(n\log_2 n)$
選一pivot worst : $O(n\log_2 n)$

<p	=p	>p

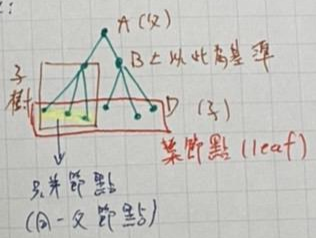
Radix sort [best : O(kn), O(n)
找一位置比較
且分類 worst : O(kn), O(n)

L8 樹 (Tree)

* Data-Management
  - 位置導向 (Position-oriented) // 依位置存
    ex: list, stack, queue, binary tree
  - 內容導向 (Value-oriented) // 依特性存
    ex: sorted list, binary search tree

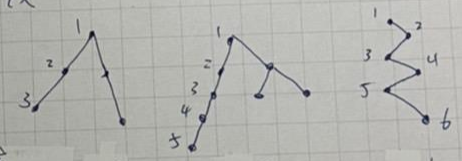
* Tree
  - parent-child (親子)
  - ancestor-descendant (祖孫)
  - subtree (子樹)
  - leaf (葉節點)
  - siblings (兄弟節點)

ex:



* complete binary Tree (完全樹)
  - → 每一個 node 在 level $\leq h-2$ 有 2 個 children
  - → 如最後一層不滿，須都往左靠

ex

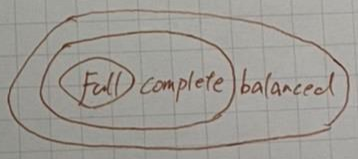


高: 3, 5, 6

↳ 路徑最長者 // 最大階層

* balanced binary Tree (平衡樹)

  如 2 個 subtree 的高不超過 1 則為平衡樹



* binary tree
  - min height: $\lceil \log_2(n+1) \rceil$
  - max height: $\lfloor \log_2(n) \rfloor + 1$

$N_0 = N_2 + 1$ (Leaf node) (Internal nodes)

* 邊比點數少 1

合重

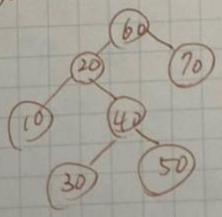
前序 preorder ⟷ [ ] (遞迴呼叫)

中序 inorder ⟷ [ ] (遞迴呼叫)

後序 Postorder ⟷ [ ]

(8-6)

ex:



前：60, 20, 10, 40, 30, 50, 70

中：10, 20, 30, 40, 50, 60, 70

後：10, 30, 50, 40, 20, 70, 60