Unit 5

Stack 堆疊 (Last In First Out)

用途：後序式運算

isEmpty() 判斷 stack 是否為空

push(in newItem:StackItemType) 從頂端放入

pop() 從頂端取出

getTop() 取得頂端元素資訊

概念：

queue：佇列資料有「先進先出」(first in, first out, FIFO)的特性

※像排隊買票的概念，先排隊的客戶被服務

C. queue 內建用法 (基本)

宣告 queue<int> q;

把元素 x 加進 queue q.push(x);

取入值(不移除值) x=q.front();

移除值 q.pop();

判斷是否為空的 queue q.empty()

判斷大小 q.size()

Compare between Stack and Queue.

堆疊(Stack)是一種資料結構，遵循著資料後進先出(Last In First Out, LIFO)最後一個進去，最先出來的原則，較新的元素會靠近頂部(堆疊尾部)，較舊的元素會在堆疊的底部。

佇列(Queue)是一種資料結構，遵循著資料先進先出(First In First Out, FIFO)，第一個進去，第一個出來的原則，較新的元素會靠近頂部(佇列尾部)，較舊的元素會在佇列的底部。

氣泡排序法(Bubble Sort)

又稱泡沫排序，是一種簡單的排序演算法，它重複走訪過要排序的數列，一次比較兩個元素，如果他們的順序錯誤就把它們交換過來，走訪數列的工作是重複地進行直到沒有再需要交換，也就是說該數列已經排序完成。這個演算法的名字由來是因為越小的元素會經由交換慢慢「浮」到數列的頂端。

選擇排序法(Select Sort)原理是反覆從未排序數列中找出最小值，將它與左邊的數做交換，我們有兩種方式排序，一為由大到小排序時，將最值放到末端，若由小至大排序時則將最小值放到前端，例如：未排序的數列中找到最小值的資料，和第一筆資料交換位置，再從剩餘排序的資料中找到最小值的資料，和第2筆資料交換位置，以此類推。

插入排序法(Insert Sort)原理是在一筆原始資料加入已排序好的資料中，並在一與已排序好的資料作比較，找到對的位置插入，例如：已有2筆排序好的資料，將第3筆資料與前面已排序好的2筆資料作比較，找到對的位置插入，再將第4筆資料與前面已排序好的3筆資料作比較，找到對的位置插入以此類推。

合併排序法(Merge Sort)原理是會先將原始資料分割成兩個資料列，接著再將兩個資料繼續分割成兩個資料列，依此類推，直到無法再分割，也就是每組都只剩下一筆資料時，再兩兩合併各組資料，合併時也會進行該組排序，每次排序都是比較最左邊的資料，將較小的資料加到新的資料列中，依此類推，直到最後合併成一個排序好的資料列為止。

快速排序是對泡沫排序的一種改進。通過一趟排序將要排序的數據分割成獨立的兩部份，其中一部分的數據都比另外一部分的數據要小，然後再按此方法對這兩部分數據分別進行快速排序，整個排序過程可以遞迴進行，以此達到整個數據變成有序。

基數排序(Radix Sort)

又叫基底排序是一種分配式排序，可以多鍵值排序，只有一個鍵值時，可以利用分解鍵值來進行基數排序，將數值切割以進行排序，利用桶子(Bucket)來分類，以r為基底(Base)時，需準備r個桶子⇒數值時，基底即為進制

1. 選定好本回合使用的基數
2. 依序用基數對每一筆資料取得該回合的LSD or MSD
3. 依取得的LSD/MSD將資料放到對應的桶中
4. 依桶子內存放的順序，將資料回存原本的陣列
5. 重複上面四個步驟，直到範圍內的基數皆已使用

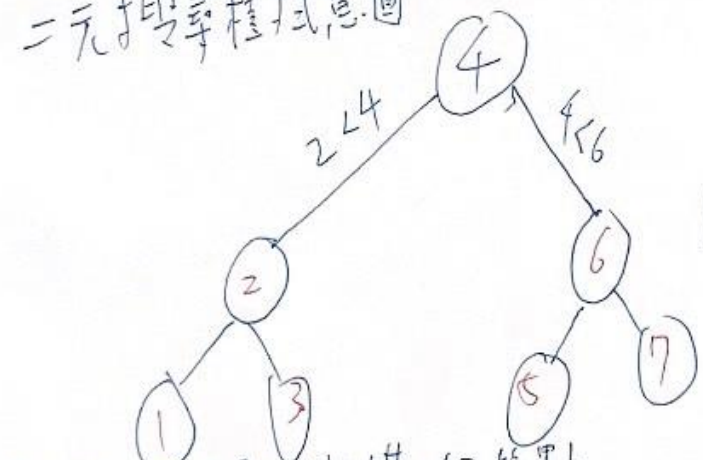
常用排序時間複雜度

Sorting Algorithm	Best Case	Average Case	Worst Case
Selection Sort	$O(N^2)$	$O(N^2)$	$O(N^2)$
Insertion Sort	$O(N^2)$	$O(N^2)$	$O(N^2)$
Bubble Sort	$O(N^2)$	$O(N^2)$	$O(N^2)$
Merge Sort	$O(N \cdot \log_2 N)$	$O(N \cdot \log_2 N)$	$O(N \cdot \log_2 N)$
Quick Sort	$O(N \cdot \log_2 N)$	$O(N \cdot \log_2 N)$	$O(N^2)$

單元八

二元搜尋樹：右子節點大於左子節點，我們可利用此特性遞迴搜尋

二元搜尋樹示意圖



In Order Traversal: 1 2 3 4 5 6 7

刪除二元搜尋樹中的某一個節點