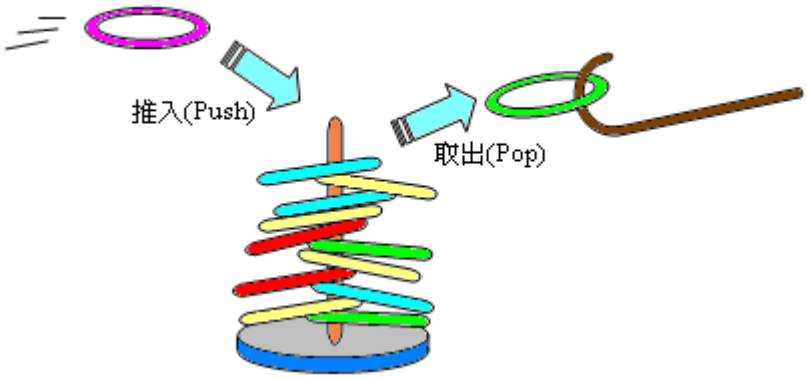
資料結構第二次筆記

單元五

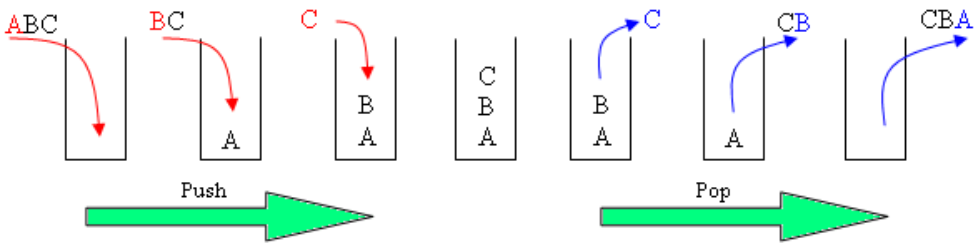
Stack：後進先出(Last In First Out) 可解決括號匹配、後序式運算

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

stack 概念圖 1



stack 概念圖 2

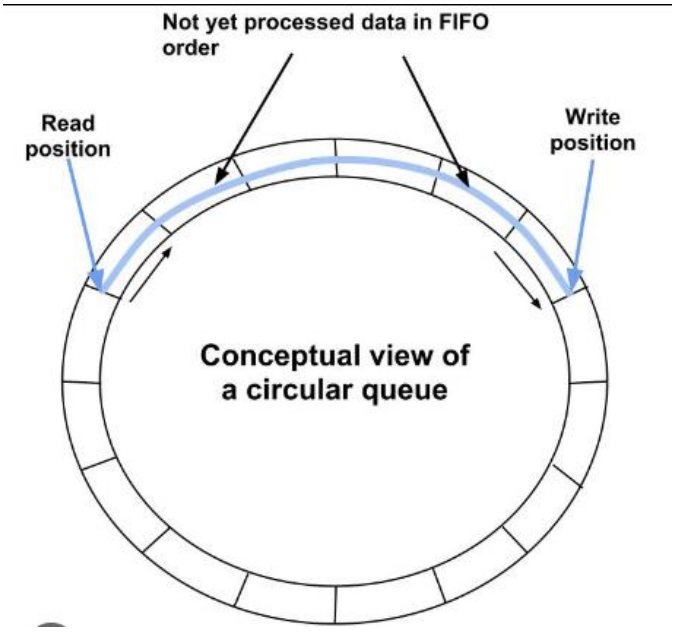


單元六

Queue: 先進先出(First In First Out) 可解決 bar 台模擬、BFS、

isEmpty() 判斷 queue 是否為空
enqueue(in newItem:QueueItemType) 排入佇列尾端
dequeue() 從頭取出
getFront() 取得頭元素資訊

queue 概念圖 1



單元七

預估程式效率: 分別為時間複雜度與空間複雜度，時間複雜度指執行的次數，空間複雜度指使用多少記憶體。

常見時間複雜度如下:

1. O(1) constant time
2. O(log2 n) logarithmic time
3. O(n) linear time 4. O(n log2n)
5. O(n^2) quadratic time
6. O(n^3) cubic time
7. O(2^n) exponential time

時間複雜度比較

常用排序時間複雜度:

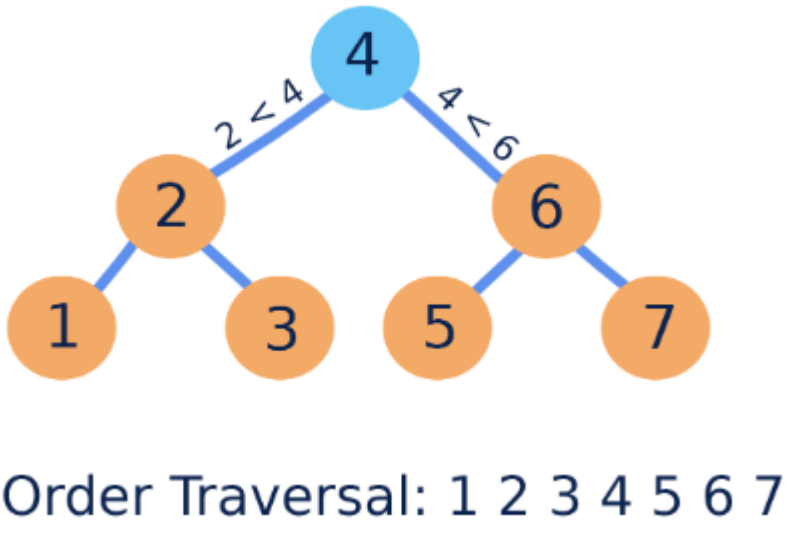
Complexity Summary

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	$O(N.\log_2 N)$	$O(N.\log_2 N)$	$O(N.\log_2 N)$
Quick Sort	$O(N.\log_2 N)$	$O(N.\log_2 N)$	$O(N^2)$
Tree Sort	$O(N.\log_2 N)$	$O(N.\log_2 N)$	$O(N^2)$

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。

二元搜尋樹式意圖



刪除二元搜尋樹中的某一個節點