Heap\_sort

簡介:

基於樹狀結構的一種排序方法

時間複雜度:

NlogN

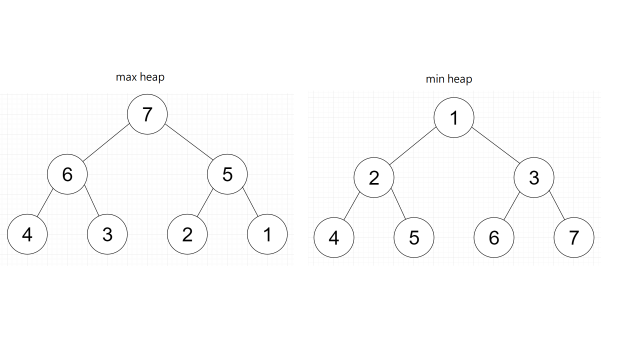
特徵:

完全二元樹，即父節點下有只有兩個子節點

類型:

max heap 和 min heap

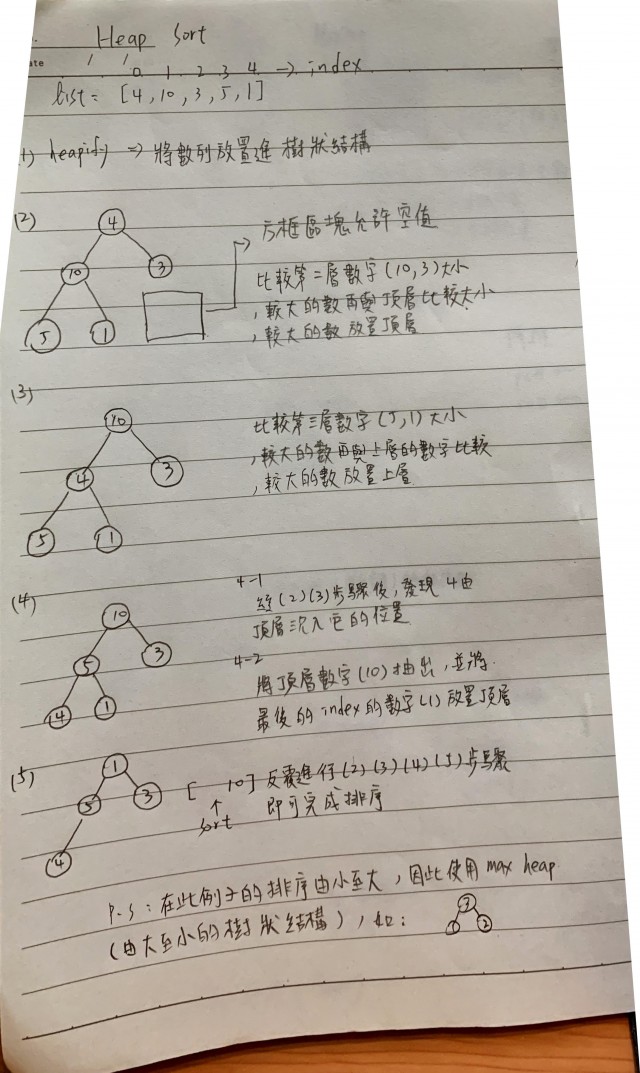
父節點大於子節點即為max heap;反之，則為min heap以及樹根(root)必為最大或是最小

[](https://ibb.co/6RBdPZS)程式實作概念:

1.heapify=>將串列轉為樹狀結構，並反覆比較父節點與子節點間大小，將最大(小)的作為父節點

2.排序=>將頂端的值與最後一個值互換，繼續讓目前位於頂端的新值進行heapify的動作

排序流程:

[](https://ibb.co/fSWr7wL)

p.s上述流程圖以max heap作為範例

在進行實作之前，我整理了一下我目前還記得的部分heapsort 內容。

For I in range(x,-1,-1):

J=2\*x+1

K=2\*x+2

Largest=i，跟其他數值比大小

If arr[largest]<arr[j]:

Largest=j

嗯….大概就剩這些了

那這次助教強調原創性，我也嘗試用我剩下的內容去製作我的程式碼。

由於在實作之前，有先用手寫heapify 程式碼的過程。

那程式碼基本完成了，但偵錯永遠都是最花時間的。

在進行第一次偵錯時，我使用的測值為[1,2,3,4,5]，想要產生出[5,4,3,2,1]的max heap(想說這樣的測值應該足以測試程式碼是否完善，min\_heap->max \_heap)，其中出了一些小差錯，大多都是條件設定不夠嚴謹，產生index 超出list的範圍，例如:if k>n:進行數值交換，但是沒設想到k=n時的狀況。

偵錯完，heapify基本已經成功了。

接下來要更完善程式的功能，沿用前面測值的結果[5,4,3,2,1]，想要產生出[1,2,3,4,5]的結果。

heapify產生max heap ，我們需要將root值與最後一個index做交換，使用arr[0],arr[num-1]=arr[num-1],arr[0]，其中num=陣列長度。

那最大的數值就會放置在最後一個index，為了不動到這個已完成的數值，我就將num-1， arr=arr[0:num]使得要heapify的新陣列沒有我們原本陣列的最大數值(5)

至於排序的部分原本是想要用append丟進去一個額外設定的陣列，但是會產生由大至小的結果，所以改成用insert，在index=0插入數值，就能保證我們後丟進去的數值都能方在第一位。

-----------結束-------------