第一大題 是非題 1~4 tree高度為N

X1. full binary tree 的 right subtree 高度不一定為N-1 一定為N

O2. complete binary tree 的 right subtree 高度不一定為N-1

O3. binary search tree 的 right subtree 高度不一定為N-1

O4. max heap 的 right subtree 高度不一定為N-1

O5. max heap的高度與節點的數量有關與節點的值無關

X6. binary search tree的高度與節點的數量與節點的值有關

X7. max heap 的最大值一定在right subtree的leaf 節點 最上面

O8. binary search tree 的最大值一定在right subtree的leaf 節點

9~~13????

X14. 在？？？僅能找到一個minimum spanning tree minimum spanning tree通常不只一種

X15. graph僅能找到一個Shortest path 兩條路weight(權重)一樣時能有兩條

16. 找尋365 給出兩個搜尋過程 問是否正確(四個小題)

X (a). 2->591->299->301->452->284->340->365 畫出來就知道了 284時錯。

第二大題 填空 big o的問題 問刪除所需要的time complexity

1.

(A) sorted array based(binary tree) ????????

(B) sorted array based(linear) O(n)

(C) unsorted array based O(n)

(D) sorted link based O(n)/O(1)取決於有沒有傳入要刪除的指標。 有則O(1)。

(E) unsorted link based O(n)/O(1)取決於有沒有傳入要刪除的指標。 有則O(1)。

2.單字填空

1.節點(N…) Node

2.兄弟節點(S…) Sibling

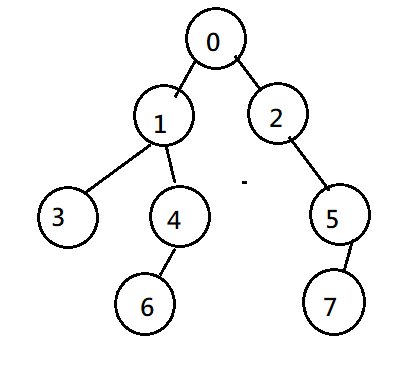
3.graph的邊(E…) Edge

4. graph的點(V…) Vertex

5. graph的路徑(P…) Path

第三大題 問答

1. 寫出 preorder inorder postorder



pre: 01346257

in: 31640275

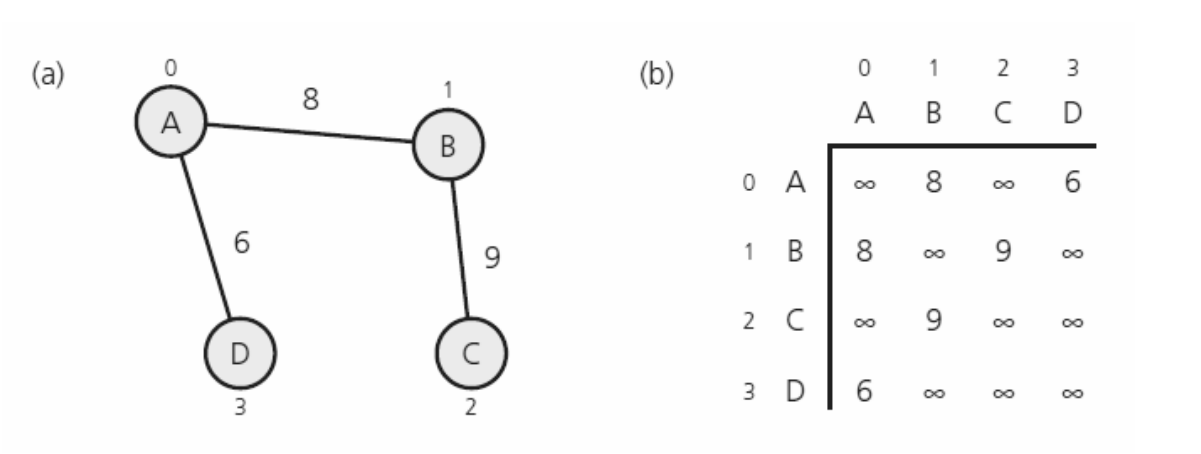
post: 36417520

2. 問BFS 與 DFS 路徑 (給adjacency matrix)

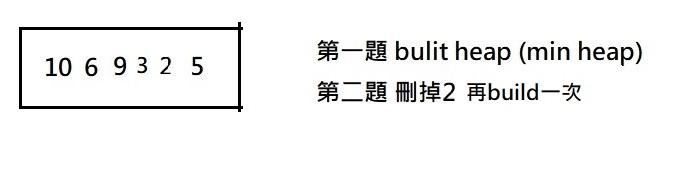
可以由給的b圖畫成a圖再進行bfs和dis(他會給從哪個點開始) 例 從點B開始BFS和DFS:

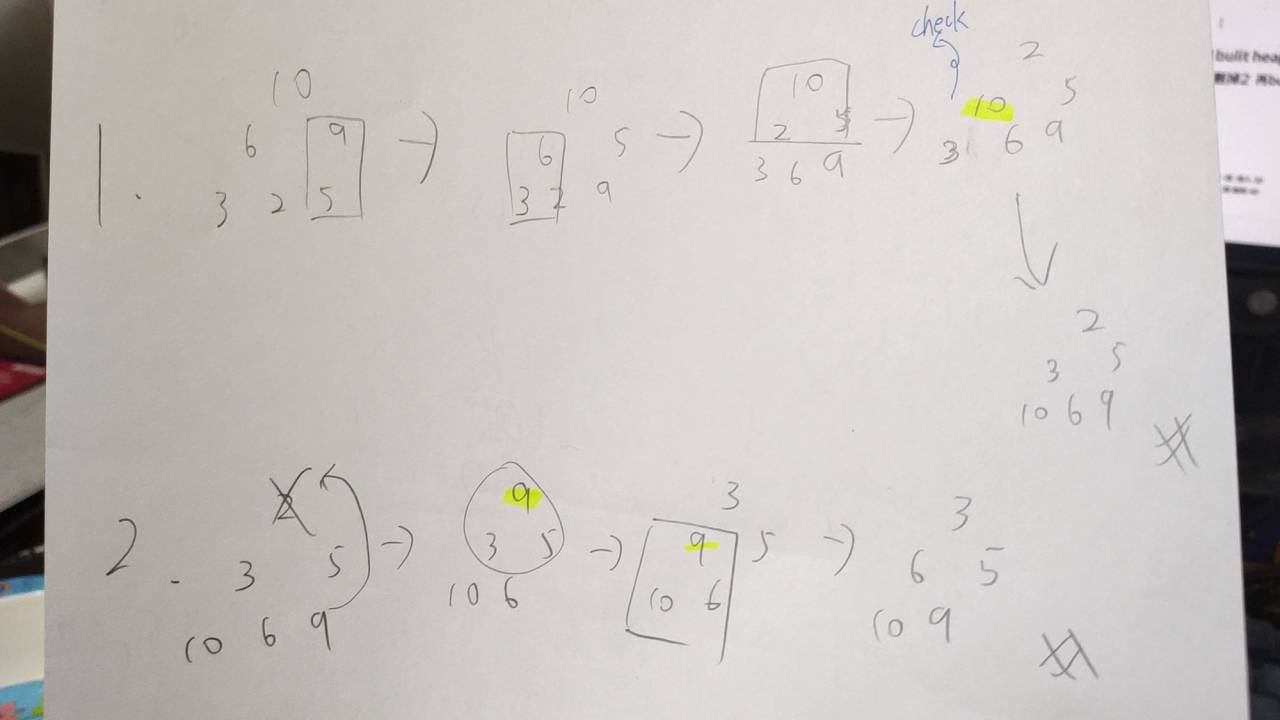
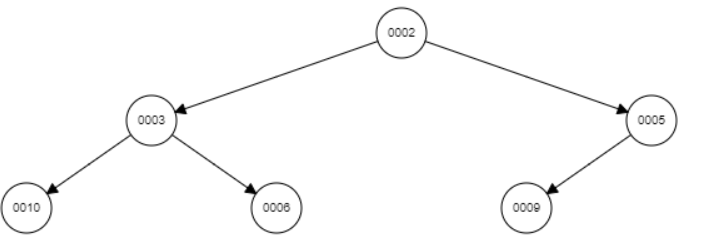
BFS:BACD

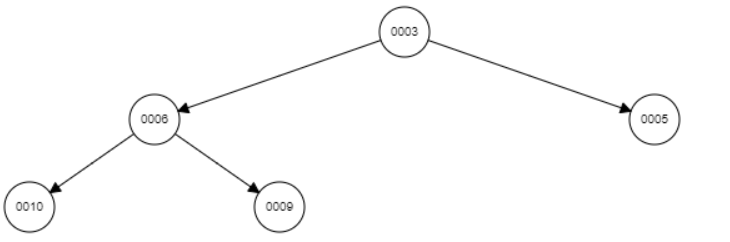
DFS:BADC



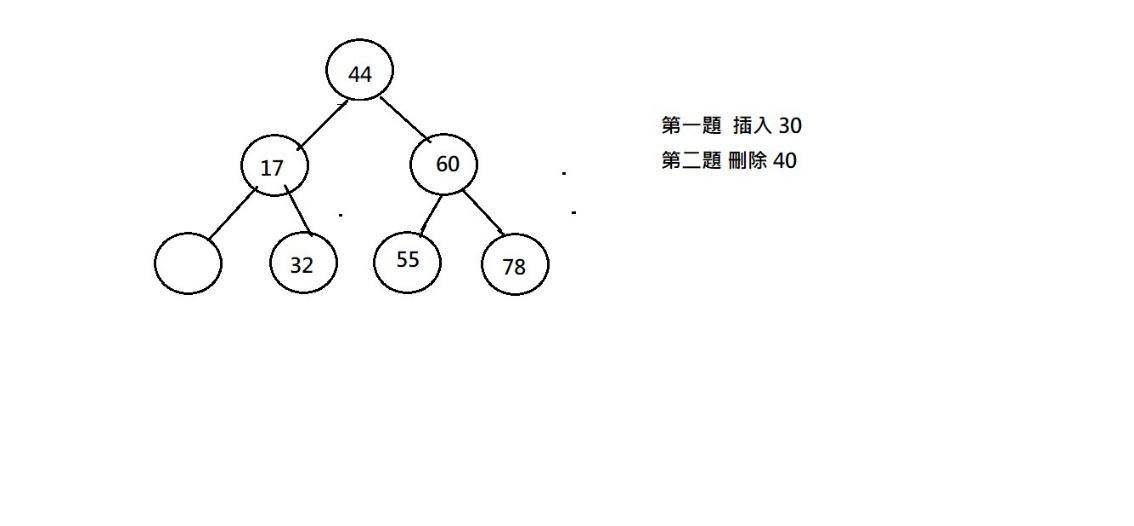
3. Heap array-based





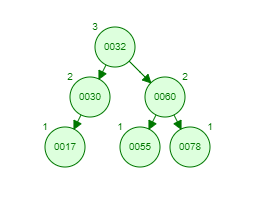


4. AVL TREE （插入跟刪除）

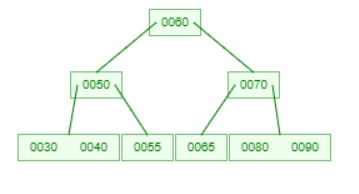
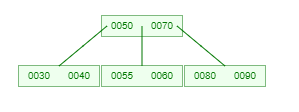
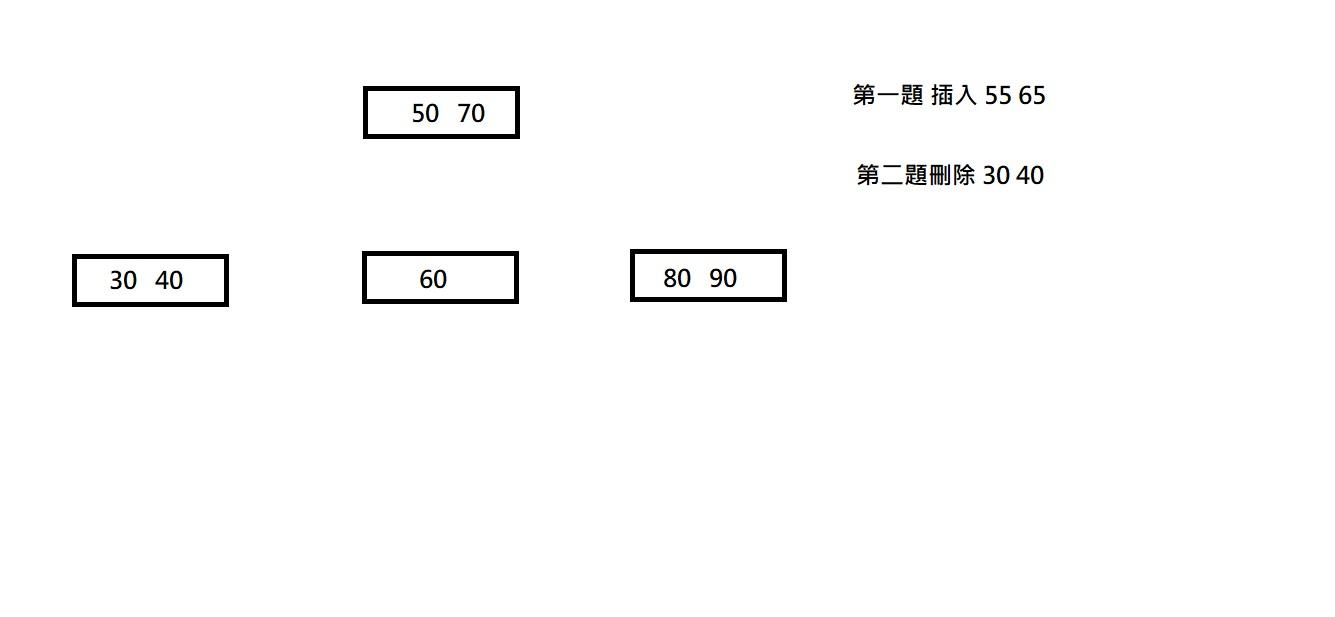


1.當作那個空圈沒有，30會先加在32左下角 在17這個點會不平衡，RL==小大中 所以30放在17的位子17放到空圈那，(應該不會考:如果旋轉完有孤兒把它放到BINARY SEARCH TREE該放的位子)

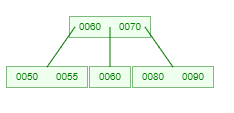
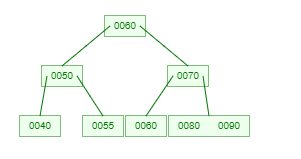
2. 32放上去



5. 2-3 TREE （插入跟刪除）



2.刪除



6. Hashing table

hash function: H(k)=(2k+5) %11

2,14,43,13,23,7,23,29,18,6,17

帶入1式:

9, 0, 3 , 9 , 7 ,8, 7, 8, 8, 6, 6

帶入2式:

5, 7, 6, 1 , 5 , 7, 5 ,6 , 3 , 1, 4

(a)linear probing

(b)Quadratic probing

(c)Double hashing -> (secondary hash function: H’(k)=7-(k%7) )

(d)Separate chaining (記得插入要從list前插）

我寫c. 不確定對不對 你可以對一下

0: 14

1: 23

2: 6

3: 43

4: 29

5: 17

6: 18

7: 23

8: 7

9: 2

10:13