

Binary tree:

定義左節點值小於根節點值，右節點值大於等於根節點值。

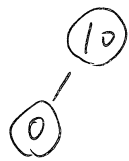
ex. 依順序有值: 10, 0, 3, 17, 75, 21, 3, 23, 5, 11

建立 Binary tree:

(1) Insert 10:



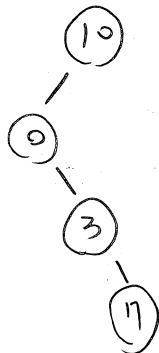
(2) Insert 0:



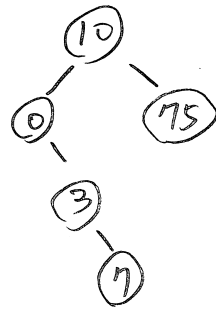
(3) Insert 3:



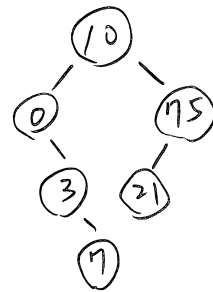
(4) Insert 17:



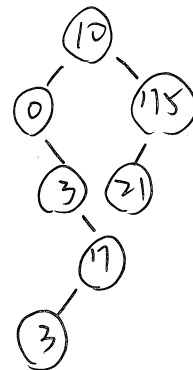
(5) Insert 75:



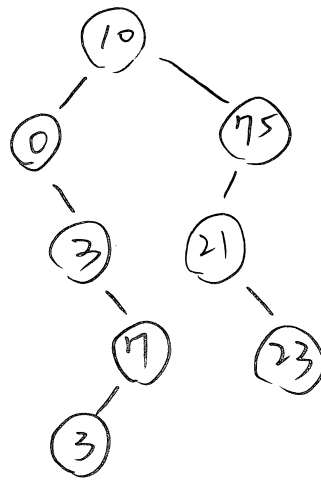
(6) Insert 21:



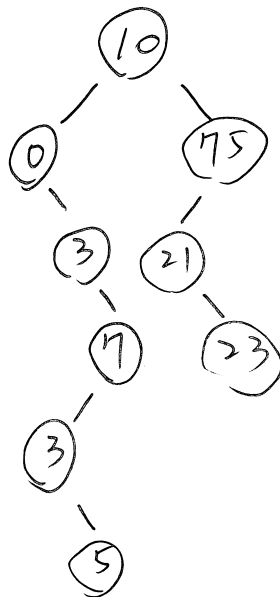
(7) Insert 3:



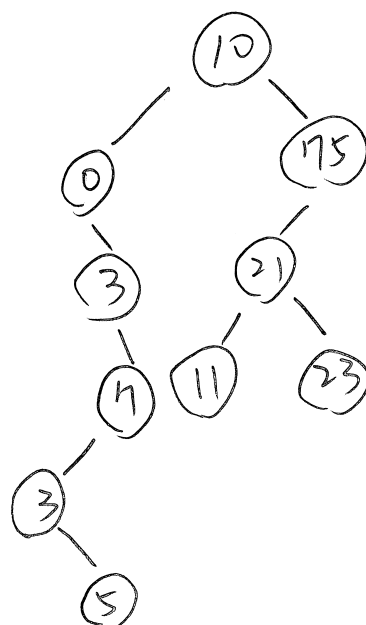
(8) Insert (23) :



(9) Insert (5) :



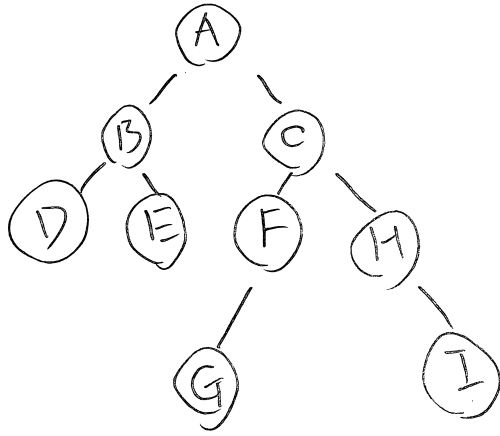
(10) Insert (11) :



Inorder traversal:

先走訪左子樹，再印根節點，最後走訪右子樹。

ex.



Inorder successor 連法:

A → D → B → E → A
→ G → F → C → H → I
→ A, 原點和終點不會
印出值來。

作法: 遞迴左子樹走到無法走時，印出此節點
再走右子樹，右子樹的根節點遞迴左子
樹走到無法走時，印出節點，以此類推。

∴ 走訪出來的值為

D B E A G F C H I

Binary tree 節點的刪除:

有三種: 1. 刪除 leaf。

2. 刪除有一子樹的 node。

3. 刪除有兩子樹的 node。

1, 2 種狀況比較單純, 第一種就直接刪除 leaf,

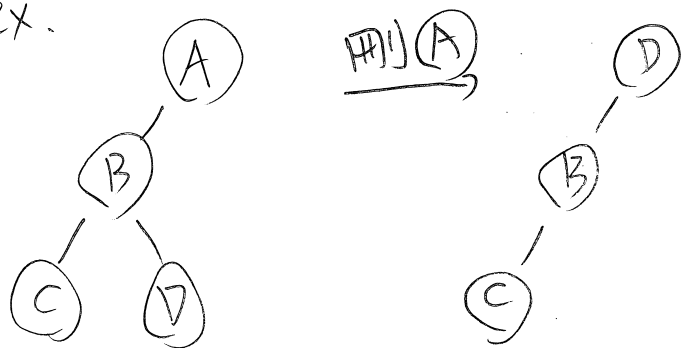
第二種, 子節點直接繼承刪除的位置。

討論第三種, 有兩種狀況:

且用 inorder successor 的方式繼承。

如果是左子樹要繼承, 就找左子樹裡的那個點下一個要走訪刪除點的那點當繼承者。

ex.



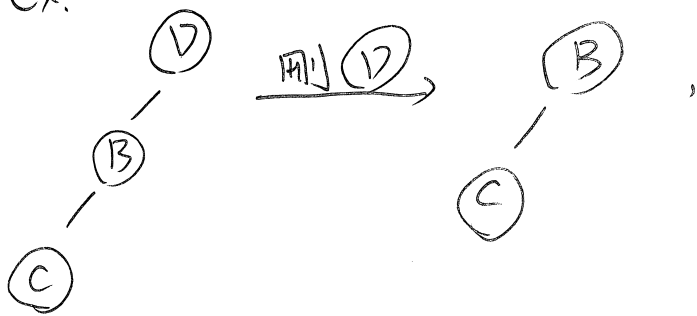
, 因為原本 Inorder 走訪為

C B D A, D 要走訪 A,

但 A 被刪除, 所以 D

繼承 A 的位置

ex.

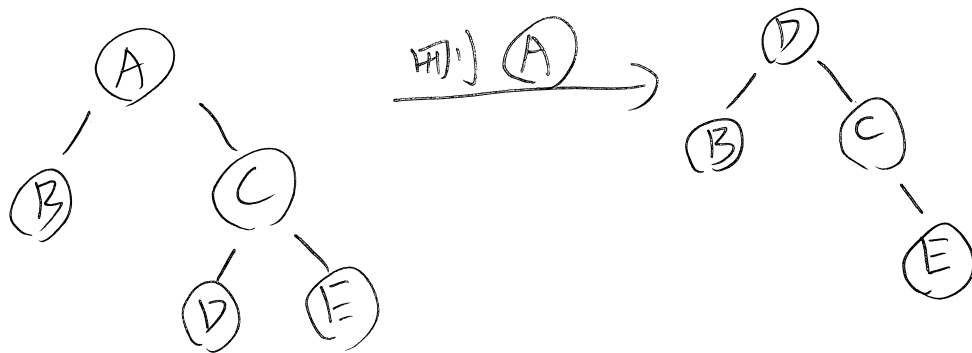


Inorder traversal:

C B D, B 下個走訪的是 D, 但 D 刪除, 所以 B 成為繼承者。

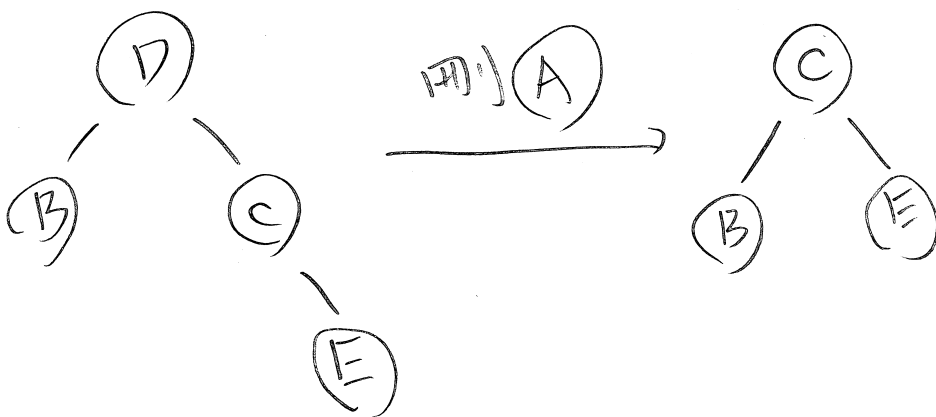
如果是右子樹要繼承: 就由 Inorder traversal 的方式, 將要刪除節下個要走訪的節點, 當作繼承者。

ex.



Inorder traversal:

B A D C E, 現在 A 刪除, 由 D 來繼承。



Inorder traversal:

B D C E, 現在 D 刪除, 由 C 來繼承。