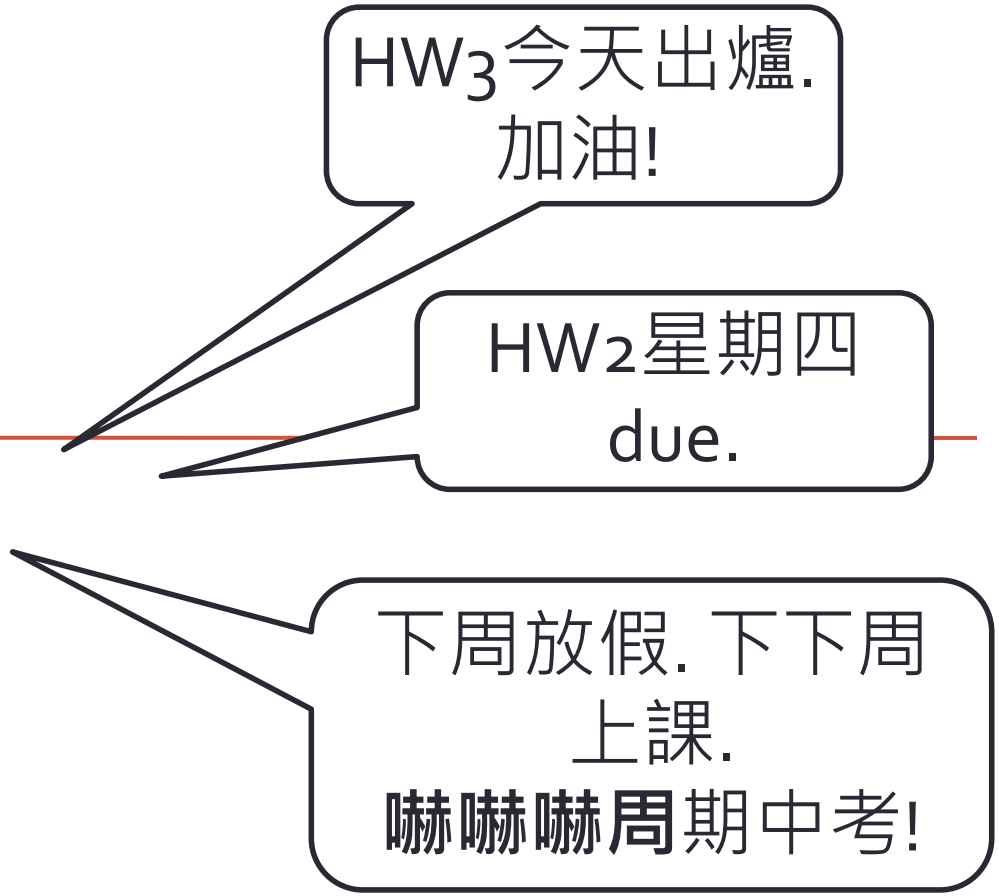
樹 2

Michael Tsai

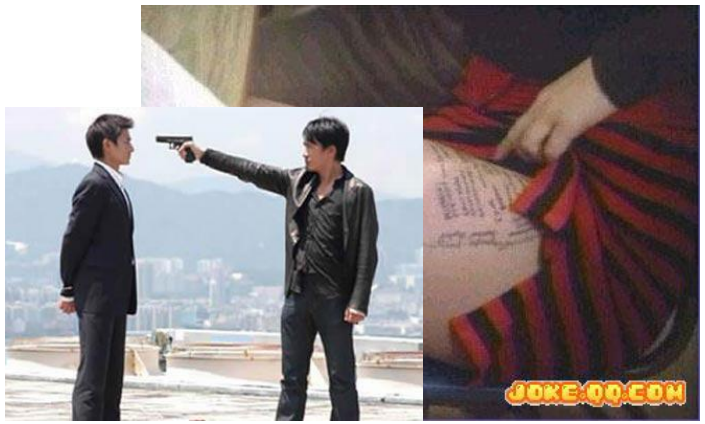
2012/3/27



期中考 (4/17)!!

- 我的想法:

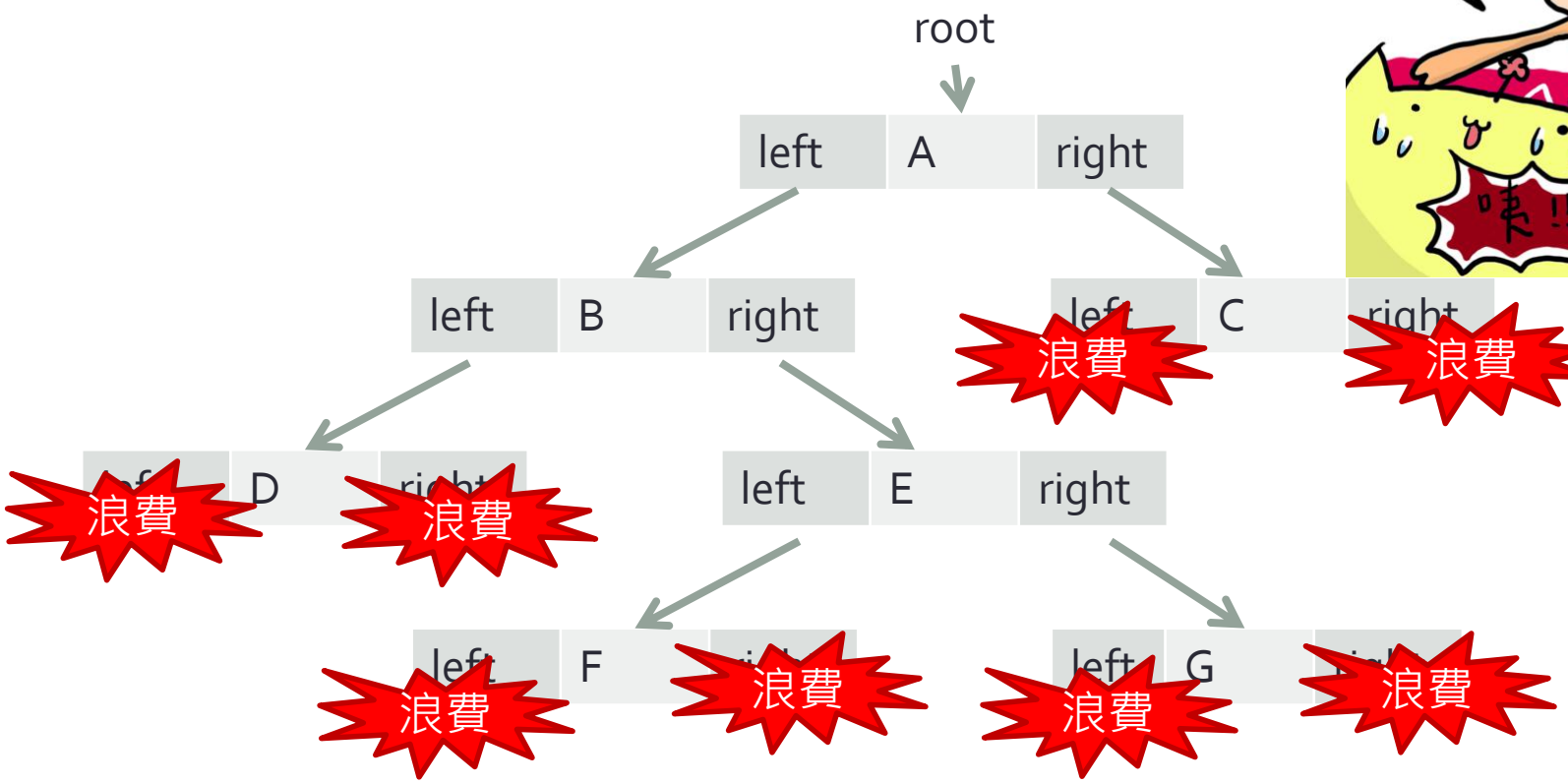
- 關書
- A4 大小一張, 雙面, 抄到你開心為止 (期末考沿用)
- 禁止使用放大鏡、顯微鏡 (供過小字體辨識用) XD
- 題目可能有
  - 是非題 (並解釋原因)
  - 填空題
  - 問答題(寫algorithm, 證明題, 問complexity)
- 請把答案寫清楚, 部分正確就有部分給分



- 作弊的直接砍頭 (當掉+送學校議處)

線頭樹??

http://www.wretch.cc/blog/z314159/7248666

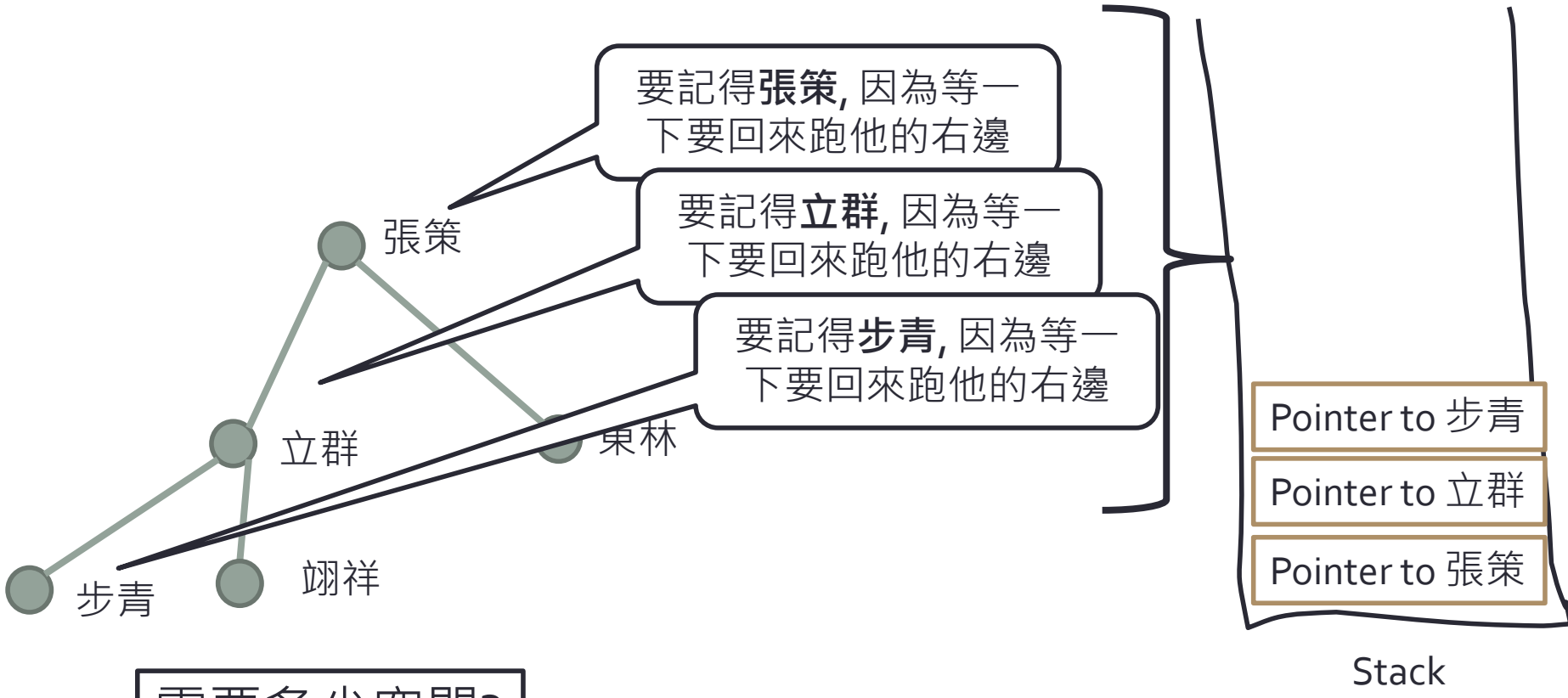


複習: 浪費了幾個pointer? (總共有n個node的話)

A: 總共有2n個pointer, (n-1)個edge/pointer不是NULL, 所以2n-(n-1)=n+1.

所以可以把這些NULL pointer欄位拿來做什麼?

怎麼做Inorder Traversal?



需要多少空間?

O(h)=O(n)

我們就拿NULL pointers來幫忙這件事!

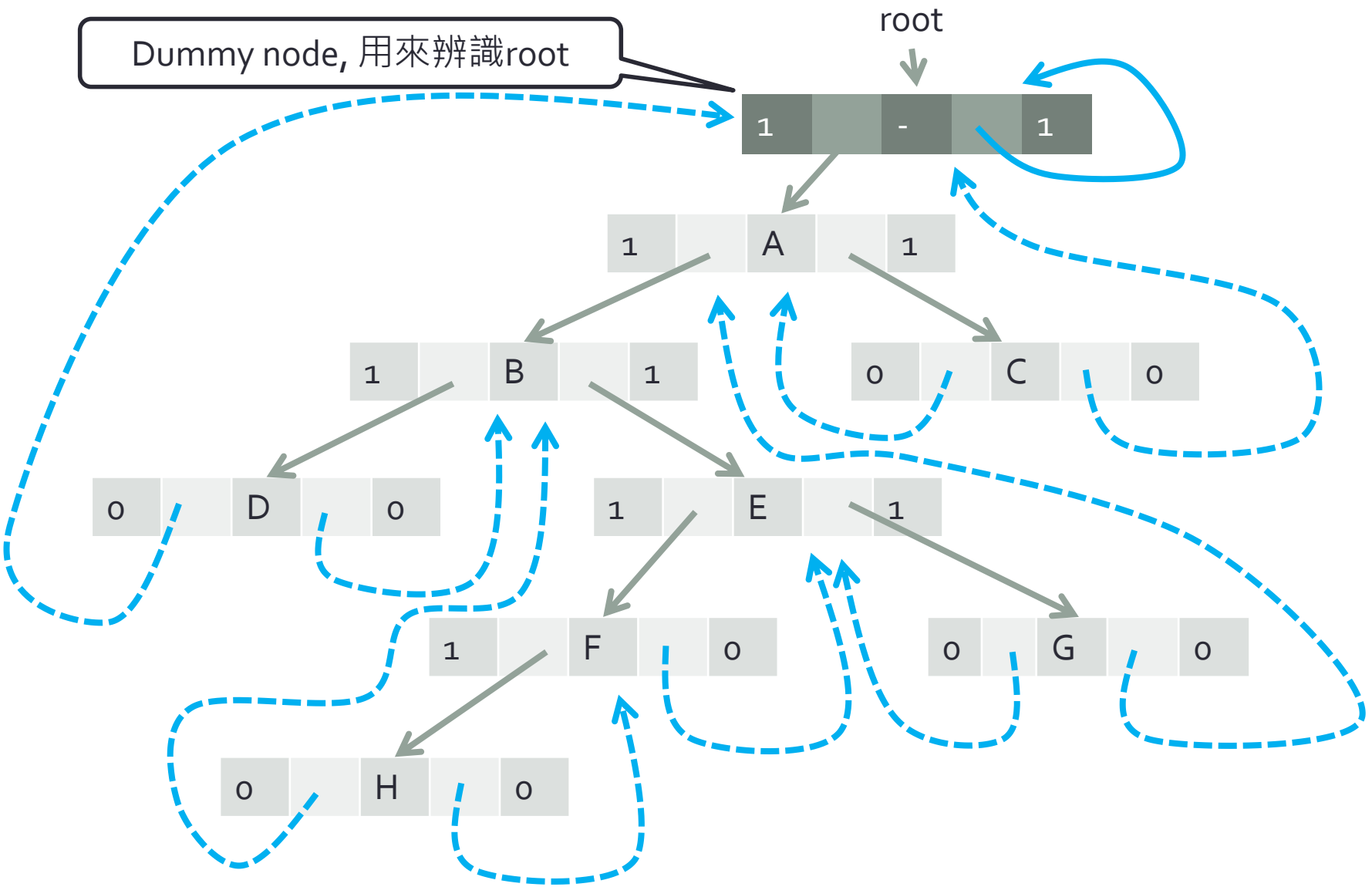
浪費掉的pointer們...

- 線頭樹: (Inorder) Threaded Binary Tree

- 1. 如果leftChild是null, 那就改成指到inorder traversal的前一個node (又稱為inorder predecessor) (此為**線頭**)
- 2. 如果rightChild是null,那就改成指到inorder traversal的後一個node (又稱為inorder successor) (此為**線頭**)
- 3. node的structure裡面放兩個額外的boolean欄位, 說明是link還是thread

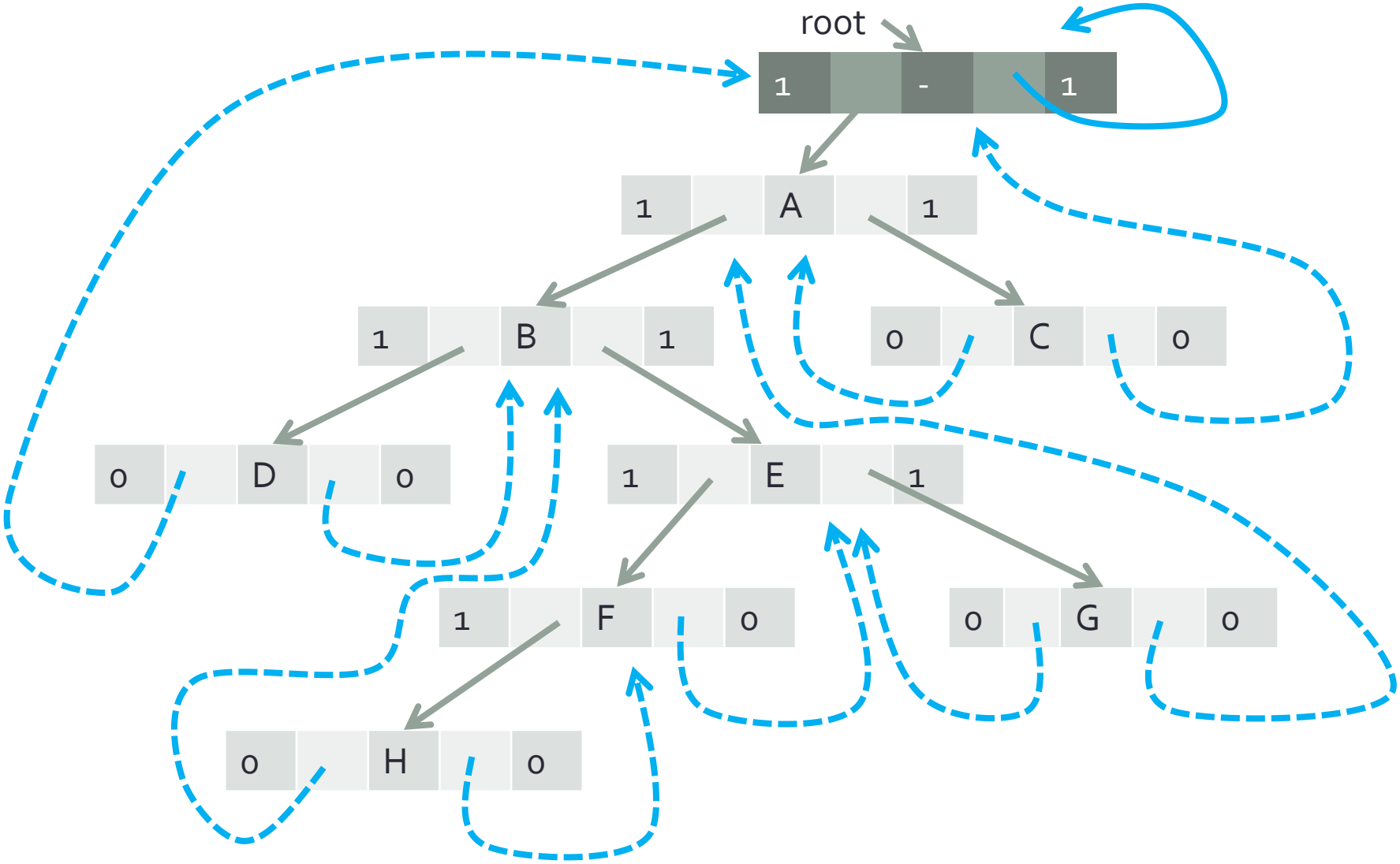
- 效果: 之後做inorder traversal不需要stack了! O(1)!

Threaded binary tree長這樣



- Inorder Traversal?
- 怎麼找到inorder successor?

1. 如果右邊不是thread, 那就找rightChild的leftChild一直走到底
2. 如果右邊是thread, 那麼就是thread指到的地方



Threaded Binary Tree

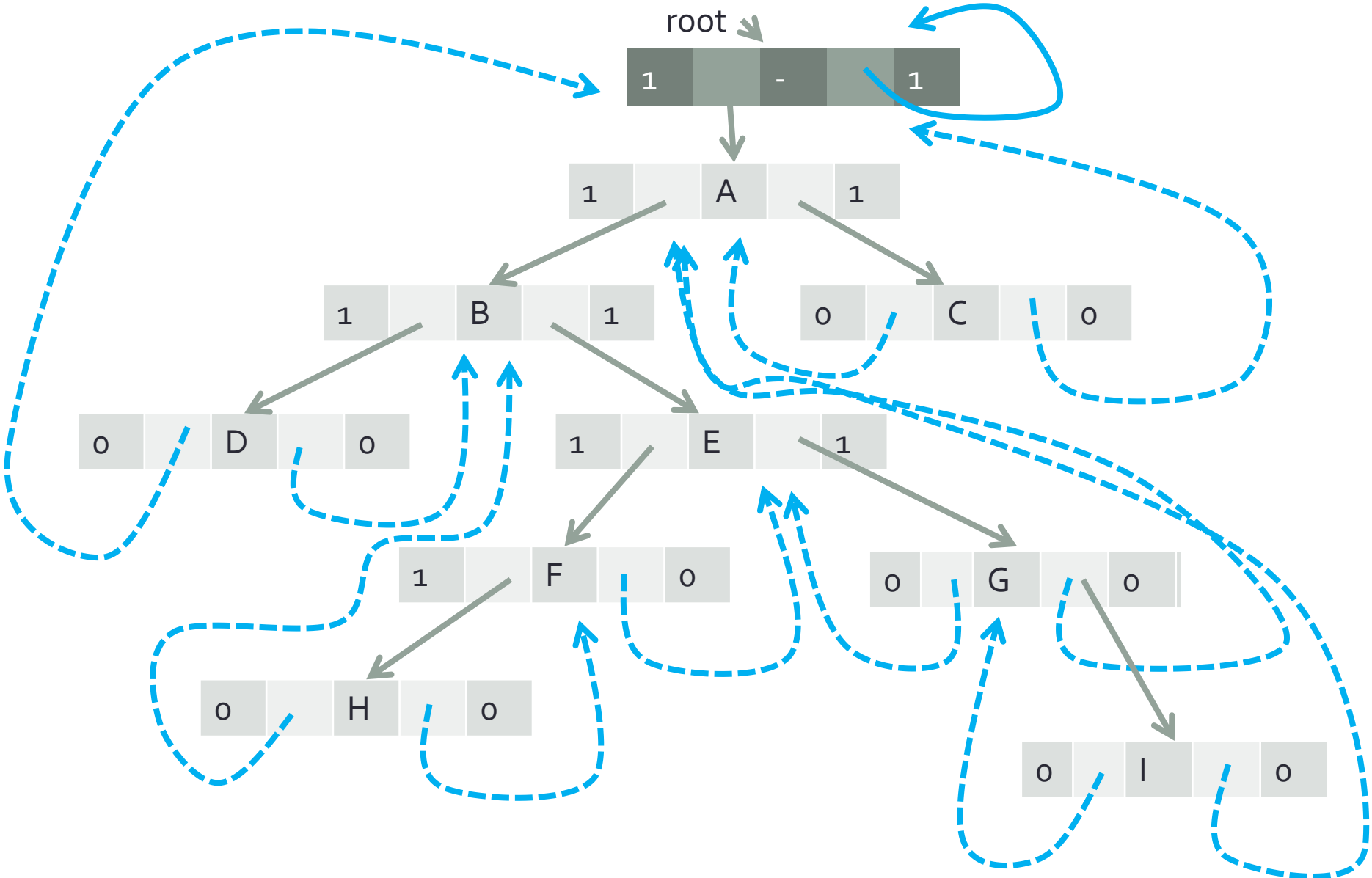
- 接著, 要做inorder traversal就很簡單了

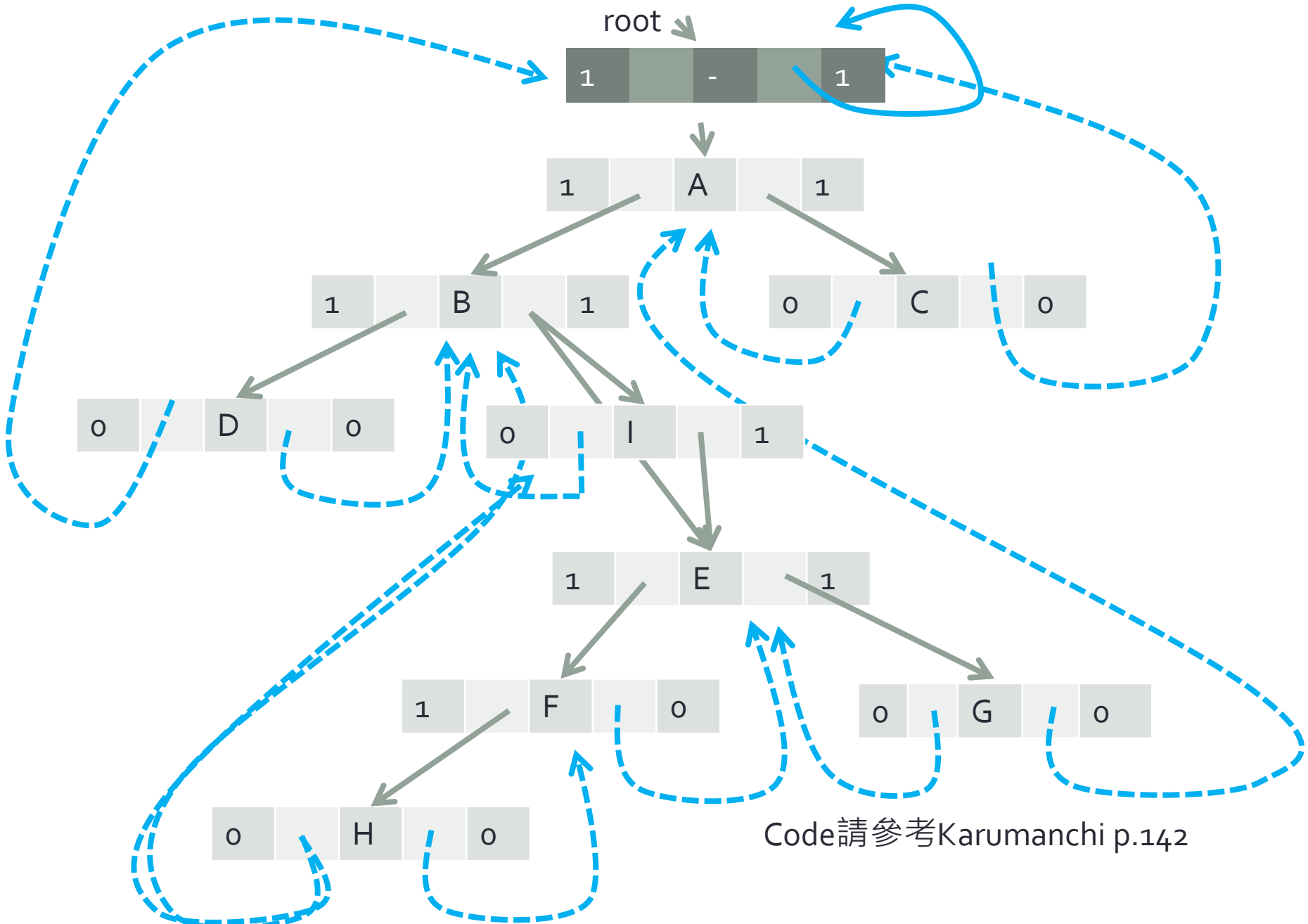
```
void InOrderTraversal(struct TreeNode *root) {
        struct TreeNode *temp=root;
        while(1){
                temp=inorderSuccessor(temp);
                if (temp==root) return;
                printf("%d", temp->data);
        }
}
```

Space Complexity: O(1)

- <動腦時間> 如果要用threaded binary tree做preorder traversal呢?
- 首先要先想想怎麼找到preorder successor
- Postorder仍然無法不使用stack來做traversal!
- Karumanchi p.141

在Threaded Binary Tree裡面加一個node





Code請參考Karumanchi p.142

Priority Queue

- 一種每次都可以拿到priority最高的element的queue
- 直接來定義operations

- Insert(element) 把element放進queue裡面
- DeleteMax() 把element拿出來. 這個element有最高的priority

- (可以想像, 放進去的時候有做一些排序)

- 另外也有FindMax(), isEmpty(), isFull等等的operation
- 請同學想想看, 要怎麼用已經學過的東西來做priority queue?

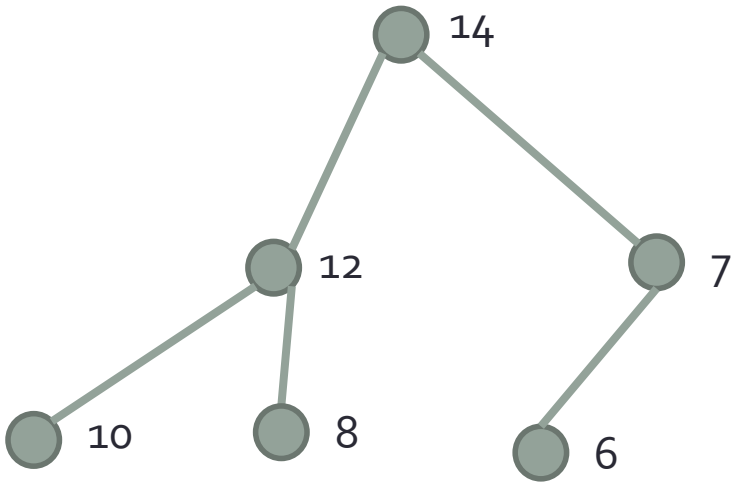
用以前學過的方法效果如何?

	Insert	DeleteMax	FindMax
Unordered Array	我是遮板		
Unordered Linked List	我是遮板		
Ordered Array	我是遮板		
Ordered Linked List	我是遮板		
Binary Search Tree	我是遮板		
Binary Heap	我是遮板		

Heap

- Definition: A **max tree** is a tree in which the key value **in each node is no smaller (larger) than the key values in its children** (if any).
- Definition: A **max heap** is a **complete binary tree** that is also a **max tree**. A **min heap** is a **complete binary tree** that is also a **min tree**.

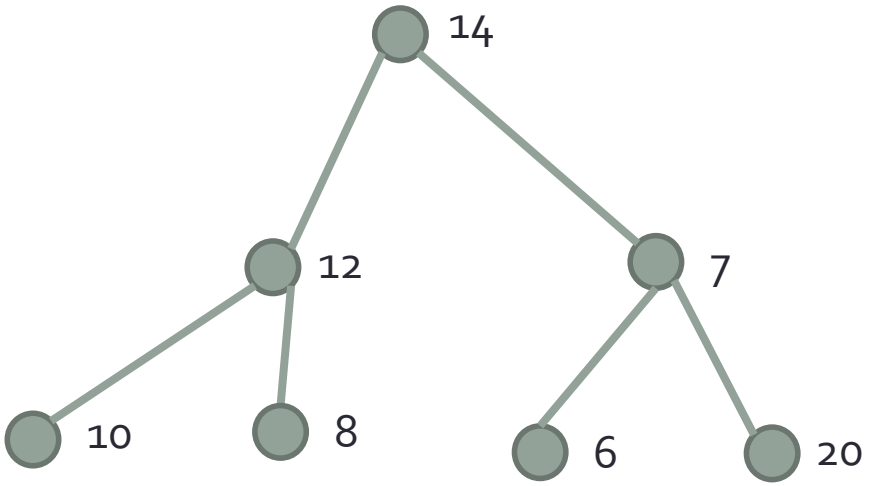
- 有了heap, 我們要怎麼用它來做priority queue?
- Root是不是永遠都是最大?



Insert一個element到Heap

- 加入的時候每次都能夠繼續維持是一個max heap
- 怎麼加?
- 1. 既然是complete binary tree, 所以一定要加在下一個該出現的地方, 把新的element放在那邊.
- 2. 循序往root的方向移動, 一直到不違反parent > child的規則為止
- Time complexity?

$O(\log n)$



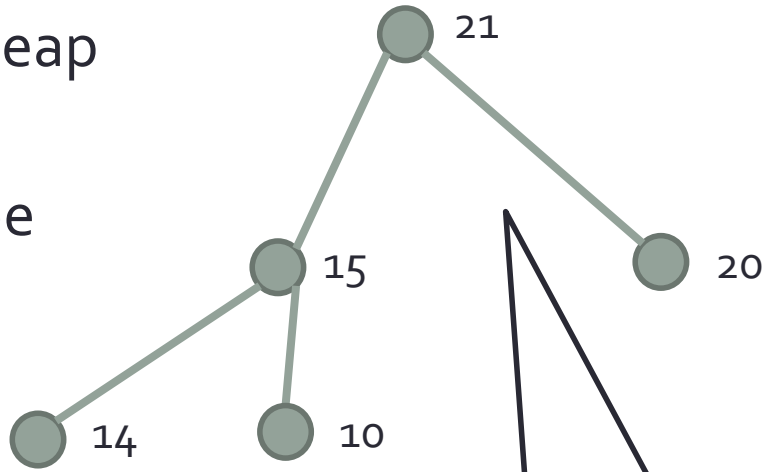
需要和6比嗎?
不用! 因為原本6的parent就會比它大了! 因此20只會更大!

從Heap DeleteMax一個element

- 從root拿走一個element (最大的)
- 調整位置, 繼續維持是一個max heap

- 1. 首先既然是complete binary tree
  拿掉的位置就沒有別的選擇.

- 2. 把拿掉的位置的element,
  拿到root的地方. 和child中比較大
  的比較. 如果比其小則與其交換.
  重複以上步驟直到不再違反
  parent > child的規則為止.
- Time complexity?



需要跟15和20都比!
因為如果拿到不是最大的
child,
則可能違反heap原則!

$O(\log n)$

用什麼data structure來implement heap?

- Array? 比較簡單? 偷懶?
- 可以. 因為每次都只會加在最後面 (complete binary tree)

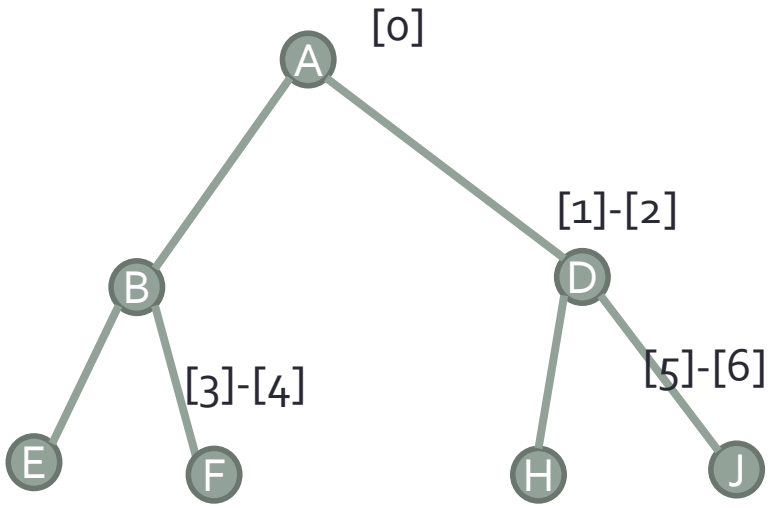
- 複習 "怎麼在記憶體裡面記一棵樹呢? Array法", Binary Tree 版

怎麼在記憶體裡面記一棵樹呢? Array法

- Binary Tree
  (每個node最多有2個children)
- 某個node的parent?
- 觀察:
  Index為i的node, 其parent之index為
  $\lfloor (i-1)/2 \rfloor$

- 怎麼找某個node的children
- 觀察:
  Index為i的node, 其children之index為
  $2i+1 \sim 2i+2$

- 這樣要找parent或是child都非常方便!



討論時間

- 給一個沒有處理過的array, 怎麼用最少的時間把它變成heap?

- Reading Assignment: p. 182 “Heapifying the Array”

- 概念:
  - 原本的array可以看成一個還沒排好的binary tree(還不是heap)
  - Leaf的部分不用處理 (因為它們沒有children, 不會違反heap原則)
  - 從最後一個(index最大的)非leaf node開始處理 (跟下面的比)
  - Time complexity=O(n log n)

Today's Reading Assignments

- Karumanchi 6.11, Problem [49-52],59
- Karumanchi 7.1-7.6, Problem 7,12,13