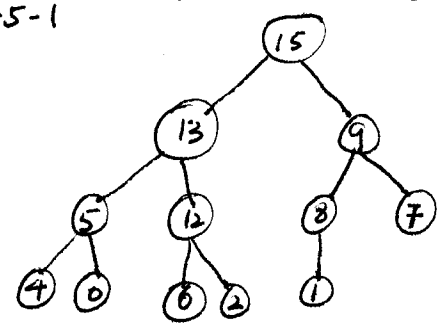
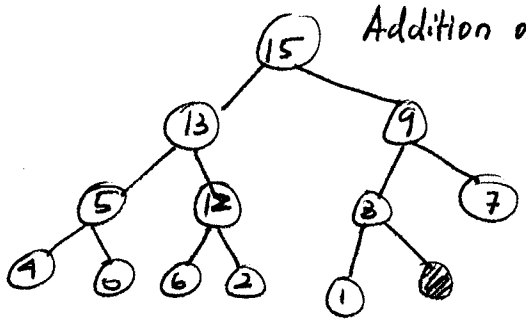
7.5-1

heap A before insert



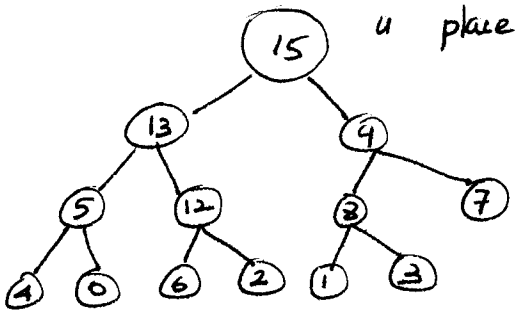
(a)

Addition of new leaf.



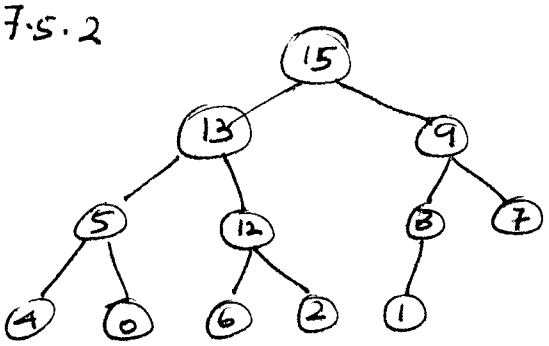
(b)

u place for '3' is found.



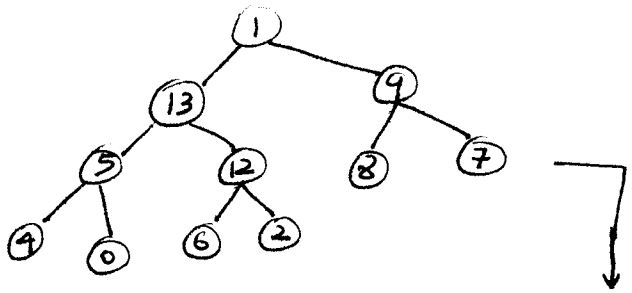
(c)

7.5.2

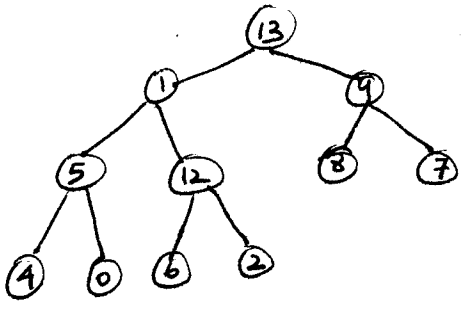


(a)

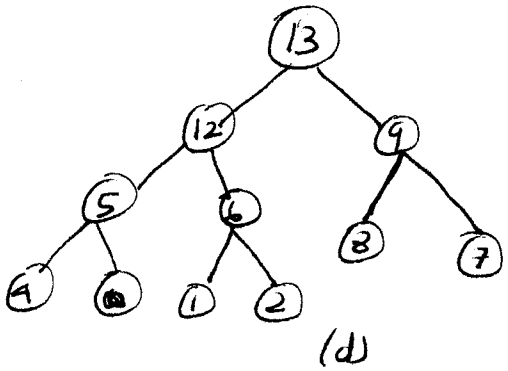
max = 15



(b)



(c)



(d)

Procedure returns 15 as the max.

and (d) gives the heapified result.

7.5.5 HEAP_DELETE [A, i]

If heap_size[A] < 1
    ERROR flag = True.

A[i] ← A[Heapsize[A]] ← O(1)

Heap_size[A] = Heap_size[A] − 1 ← O(1)

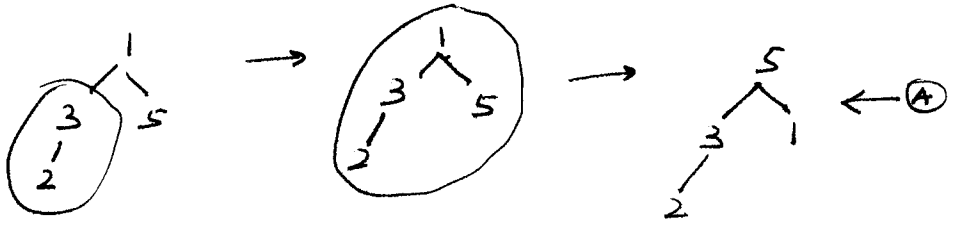
Heapify [A, i] ← O(lg n)

ORDER = O(1) + O(1) + O(lg n) ≃ O(lg n). //

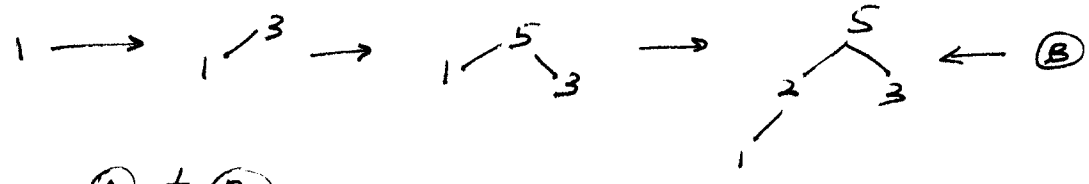
7.1 a) Counter example which shows that Build-Heap (A) and Build-Heap' (A) Need not produce the same result all the time.

take {1, 3, 5, 2}

Build-heap



Buil_heap'



Clearly (A) ≠ (B)

∴ They can produce different results.

b) BUILD-HEAP'(A)

1. Heap-size[A] ← 1 O(1)
2. for i ← 2 to length[A] O(n−1)
3. Do Heap-Insert (A, A[i]) O(lg n)

∴ Worst case (when all the elements need to be inserted

= O(1) + O(n−1) * O(lg n)

≃ O(n lg n). //