DSA through C++

Introduction to Heap

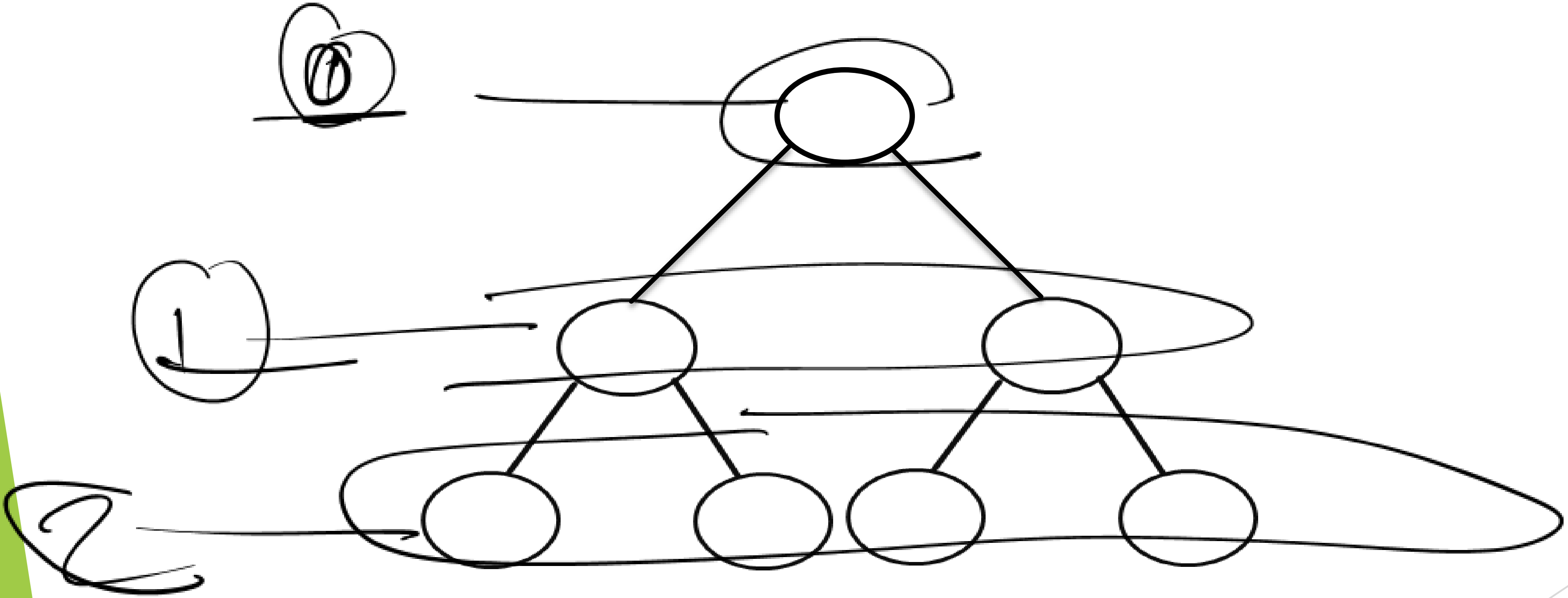
Mohammad Tasin (Tasin Coder)

Agenda

- Complete Binary Tree
- Almost Complete Binary Tree
- Introduction to Heap
- Insertion in heap
- Deletion in heap
- Heap sort

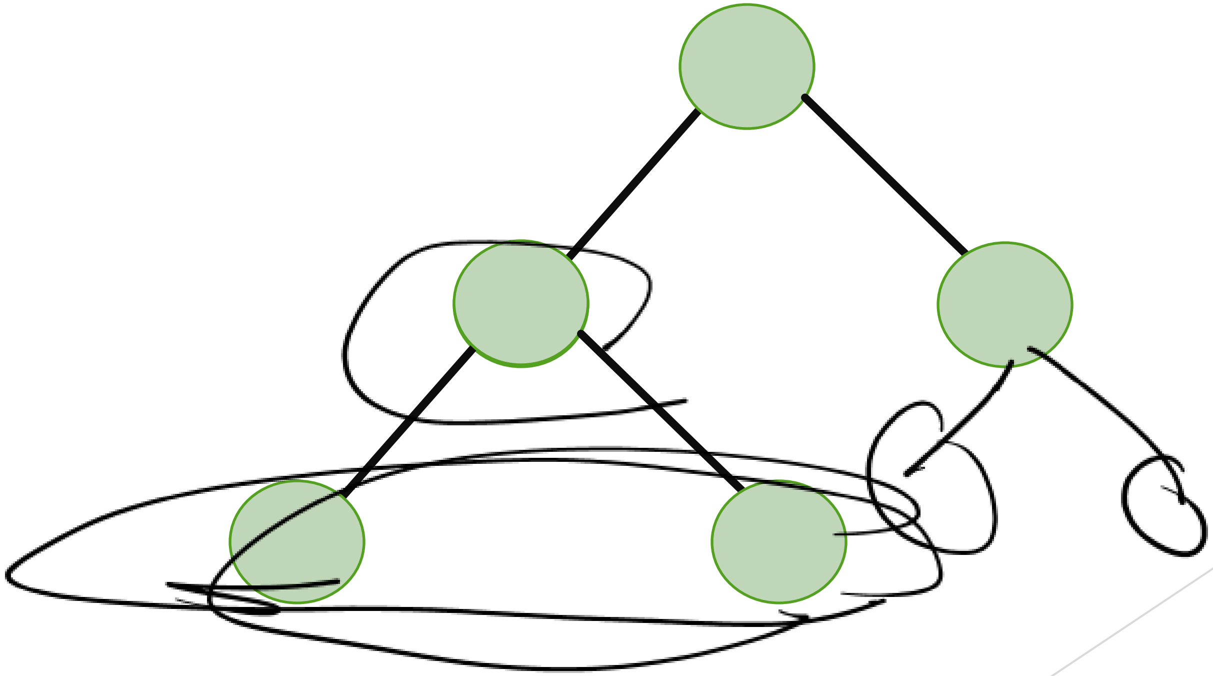
Complete Binary Tree

- **All levels must be completely filled called Complete Binary Tree.**



Almost Complete Binary Tree

- All level must be completely filled except possibly the last level and all the nodes in the last level must be left aligned.



Introduction to Heap

- **Heap is a data structure.**
- **Heap is used in a sorting algorithm known as heap sort.**
- **Heap must be an almost complete binary tree.**
- **Heap are of two types :-**
  - **Max Heap (default).**
  - **Min Heap.**

- A heap is a specialized tree-based data structure that satisfies the heap property.
- Heaps are commonly used to implement priority queues and are often used in algorithms that require quick access to the maximum or minimum element in a collection.
- The heap property depends on whether it is a max heap or a min heap.
- The value at Node is greater than or equal to value at each children of Node

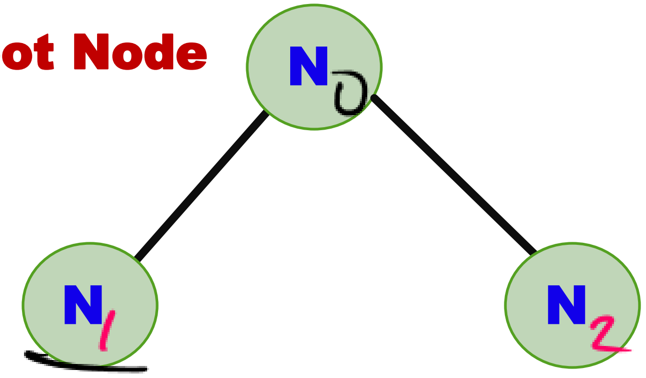
Heap Properties

- **Root Node का value आपके children से बड़ा या बराबर होना चाहिए।**
- **इसमें duplicate value allow है |**

$If(N_0 >= N_1 \ \&\& \ N_0 >= N_2)$

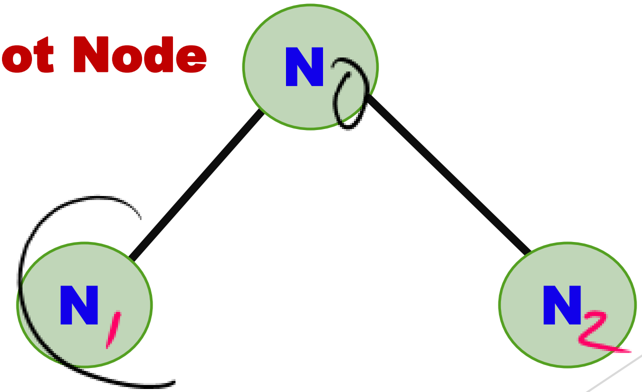
$If(N_0 <= N_1 \ \&\& \ N_0 <= N_2)$

Root Node



Max Heap

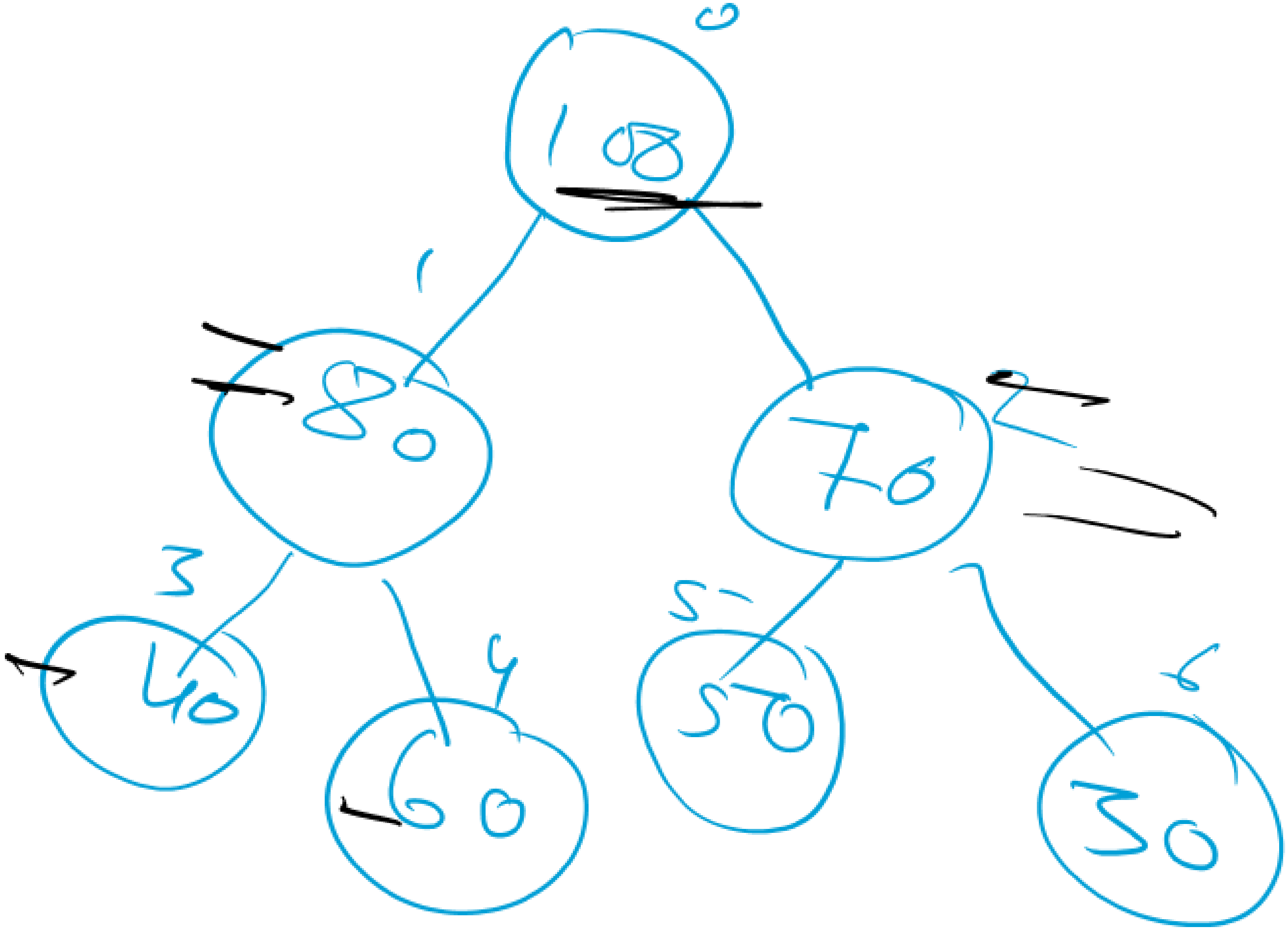
Root Node



Min Heap

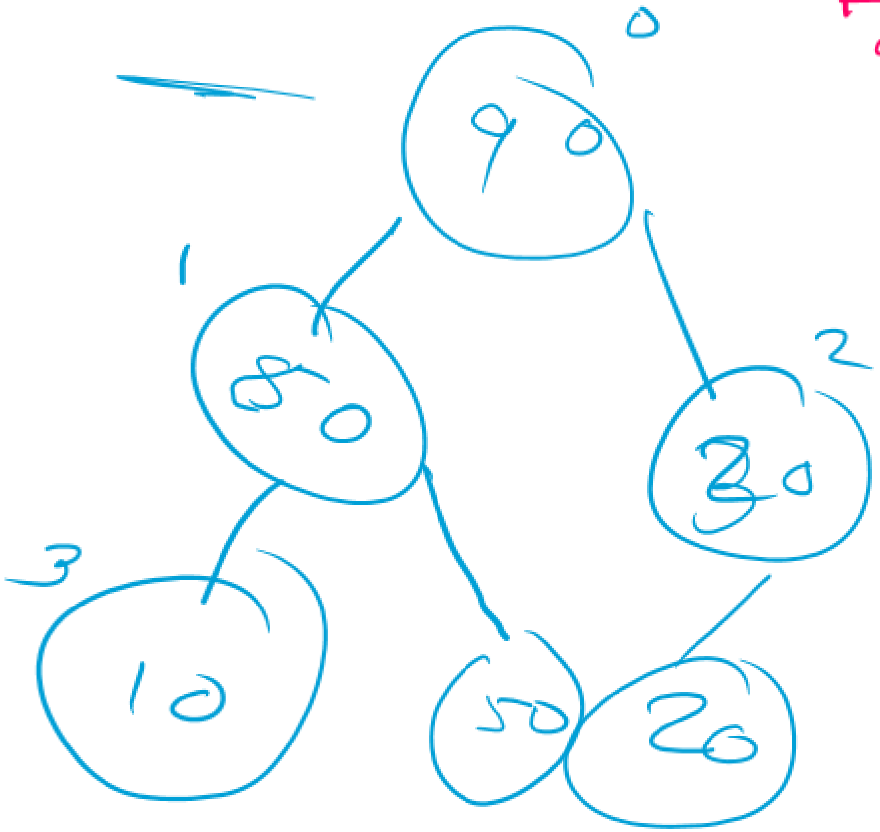
Heap Representation

0	1	2	3	4	5	6
100	80	70	40	60	50	30



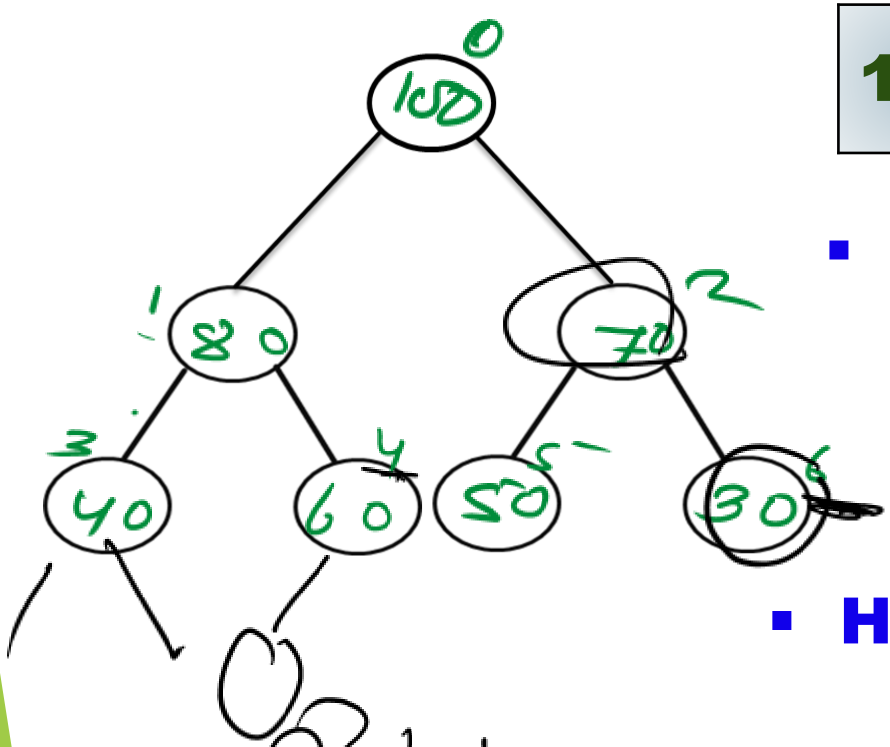
80 10 20 50 90 30

90	80	30	10	50	20
0	1	2	3	4	5



How to Find Parent or child Node

0	1	2	3	4	5	6
100	80	70	40	60	50	30



- **How to find index of child Node?**

index of left child = 2*index+1

index of right child = 2*index+2

- **How to find index of parent node?**

index of Parent node of N = $\frac{(\text{ index} - 1\ )}{2}$

8+1

= 9

2x2 = 4+2 = 6 -1

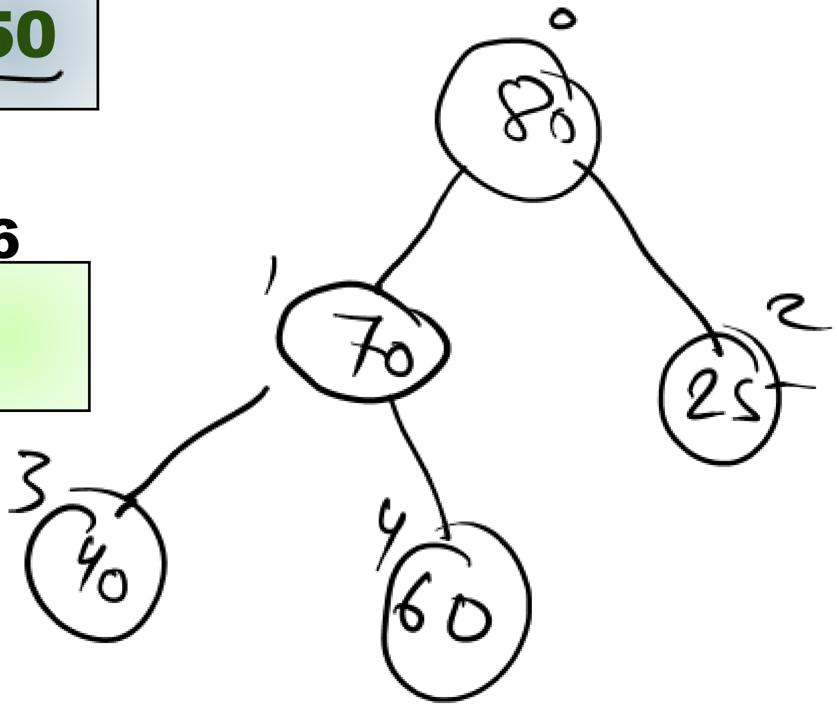
5/2 = 2

Insertion in heap

0	1	2	3	4	5	6
40	70	25	60	80	20	50

0	1	2	3	4	5	6
80	70	25	40	60		

Heap= ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ 5



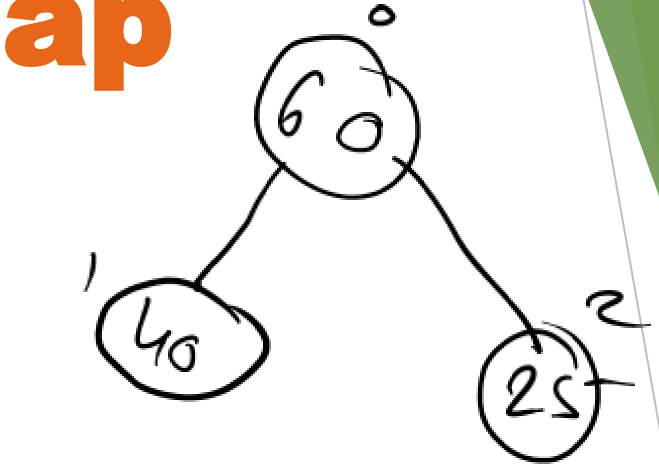
temp

Deletion in heap

0	1	2	3	4	5	6
60	40	25				

Heap = ~~8~~ ~~4~~ 3

temp



Heap Sort

- Delete values from the heap (**Max - heap**) and store them in an array from right to left.
- As a result, at the end of deleting all the elements of heap, array becomes sorted.