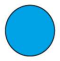
Binomial tree note U section

06 February 2025 20:38

A **Binomial Tree** B_k is an ordered tree defined recursively, where k represents the order of the binomial tree.

- If the binomial tree is of order **0** (B_0), it consists of a single node.
- In general, a binomial tree of order **k** (B_k) consists of **two binomial trees** of order $k-1$, where one is linked as the **left subtree** of the other.

If B_0, where k is 0, there would exist only one node in the tree.

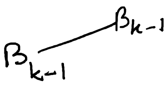


B_0

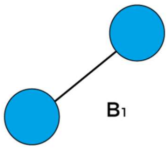
$B_k \rightarrow$ Binomial Tree of order k.

Draw B_k

Defn



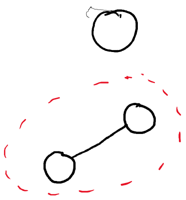
If B_1, where k is 1. Therefore, there would be two binomial trees of B_0 in which one B_0 becomes the left subtree of another B_0.

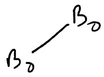


B_1

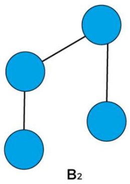
$B_0 \longrightarrow$ Binomial Tree of order 0.

$B_1 \longrightarrow$ " " " order 1



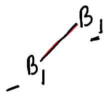


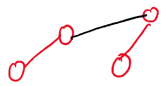
If B_2, where k is 2. Therefore, there would be two binomial trees of B_1 in which one B_1 becomes the left subtree of another B_1.



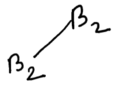
B_2

$B_2 \rightarrow$

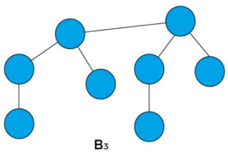




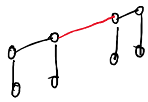
$B_3 \rightarrow$



If B_3, where k is 3. Therefore, there would be two binomial trees of B_2 in which one B_2 becomes the left subtree of another B_2.

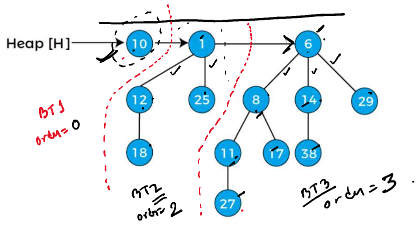


B_3



A binomial heap is a collection of binomial trees that satisfies the following binomial heap properties:

1. No two binomial trees in the collection have the same order.
2. Every binomial tree in the heap must follow the min-heap property, i.e., the value of a child node is greater than parent node.



10

Heap $\rightarrow$ Binomial Heap.

Binomial Heap Union Operation

BH1 BH2

$\Downarrow$ Merge

BH

To perform the union of two binomial heaps, we have to consider the below cases -

Case 1: If degree[x] is not equal to degree[next x], then move pointer ahead.

Case 2: if degree[x] = degree[next x] = degree[sibling(next x)] then,

Move the pointer ahead.

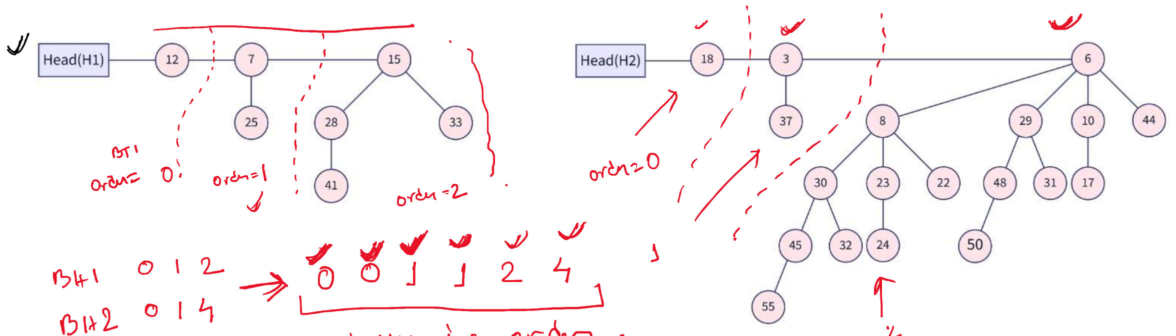
Case 3: If degree[x] = degree[next x] but not equal to degree[sibling[next x]]

and key[x] < key[next x] then remove [next x] from root and attached to x.

Case 4: If degree[x] = degree[next x] but not equal to degree[sibling[next x]]

and key[x] > key[next x] then remove x from root and attached to [next x].

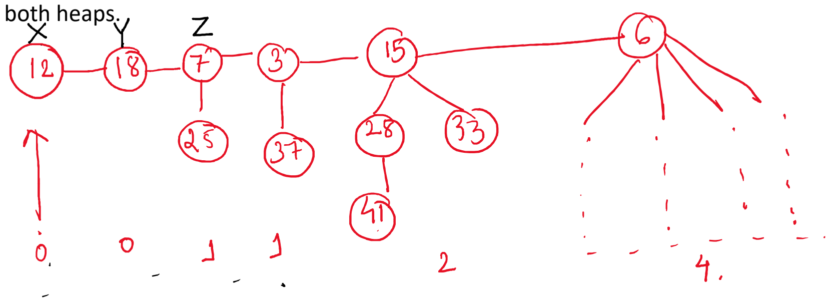
and key[x] > key[next x] then remove x from root and attached to [next x].



BH1 0 1 2
BH2 0 1 4 → 0 0 1 1 2 4

$\underbrace{\qquad\qquad}_{\text{increasing order}}$

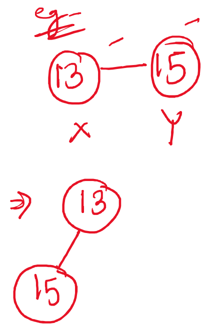
Combine both heaps.



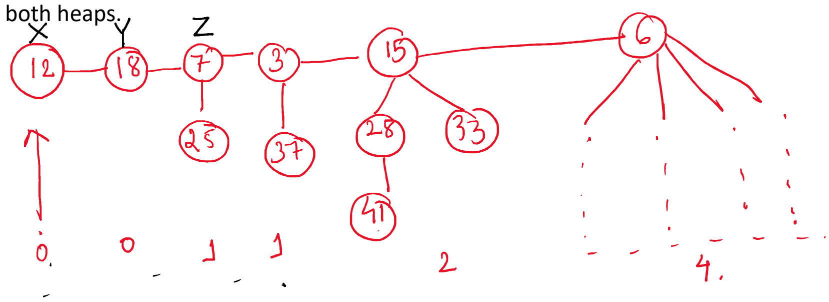
Algo –

1. Take X, Y, Z pointers.
2. Find deg(X), deg(Y), deg(Z).
3. deg(X) ≠ deg(Y) ⟶ Move X, Y, Z in right direction by 1 place.
4. deg(X) = deg(Y) = deg(Z) ⟶ Move X, Y, Z in right direction by 1 place.
5. deg(X) = deg(Y) ≠ deg(Z)
   - ⟶ key(X) <= key(Y) ⟶ X → left = Y
   - ⟶ key(X) > key(Y) ⟶ Y → left = X

eg

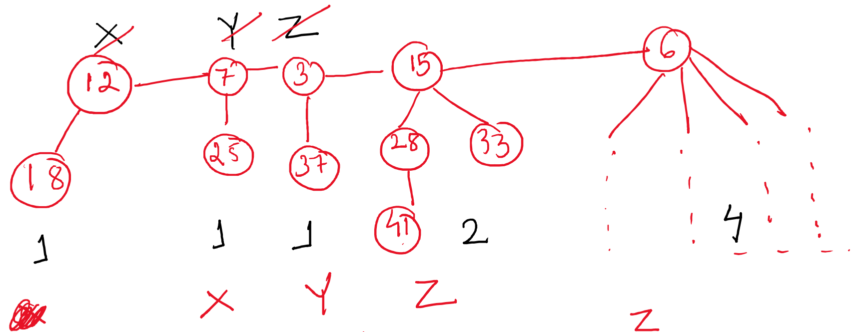


Combine both heaps.

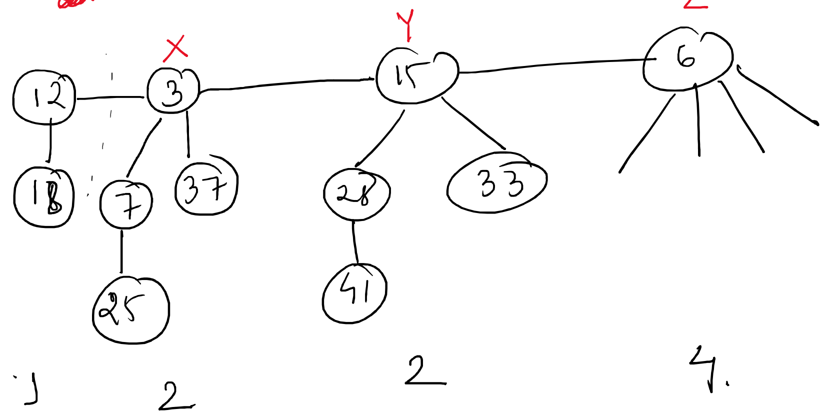


① $deg(X)=0$ $deg(Y)=0$ $deg(Z)=1.$

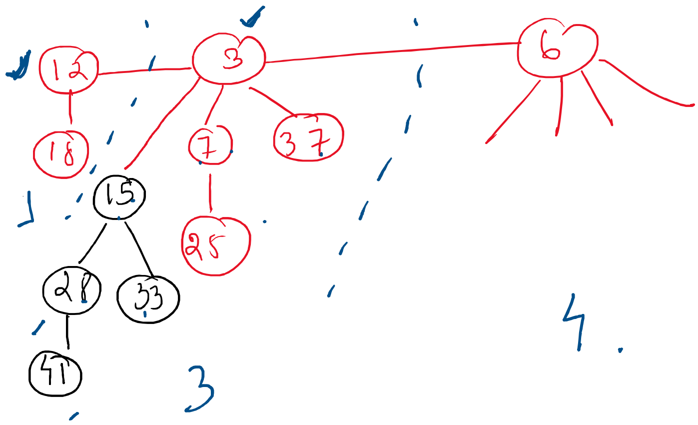
$key(X)=12$ $key(Y)=18$



$deg(X) = 1$
$deg(Y) = 1$
$deg(Z) = 1$



$$\left.\begin{array}{l} deg(X) = 1 \\ deg(Y) = 1 \\ deg(Z) = 2 \end{array}\right\} \begin{array}{l} key(X) = 7 \\ key(Y) = 3 \end{array}$$



$$\left.\begin{array}{l} deg(X) = 2 \\ deg(Y) = 2 \\ deg(Z) = 4 \end{array}\right\} \begin{array}{l} key(X) = 3 \\ key(Y) = 15 \end{array}$$

Topics H/W —

Shell sort

Bucket Sort — Imp.

B-Tree Deletion.