

## **Operations on B Tree**

### **Deletion:**

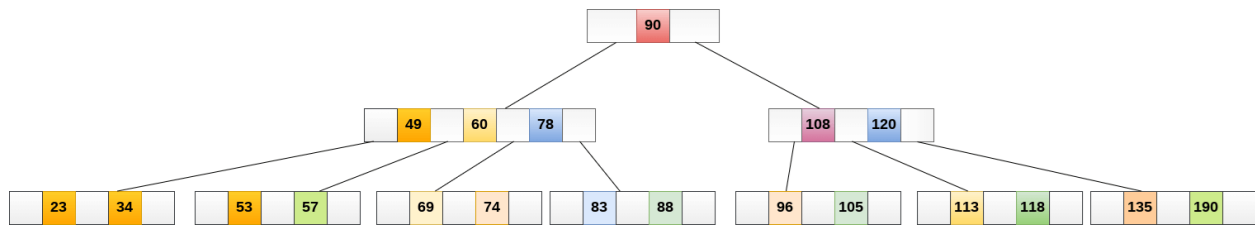
Deletion is also performed at the leaf nodes. The node which is to be deleted can either be a leaf node or an internal node. Following algorithm needs to be followed in order to delete a node from a B tree.

1. Locate the leaf node.
2. If there are more than  $m/2$  keys in the leaf node then delete the desired key from the node.
3. If the leaf node doesn't contain  $m/2$  keys, then complete the keys by taking the element from right or left sibling.
  - If the left sibling contains more than  $m/2$  elements, then push its largest element up to its parent and move the intervening element down to the node where the key is deleted.
  - If the right sibling contains more than  $m/2$  elements, then push its smallest element up to the parent and move intervening element down to the node where the key is deleted.
4. If neither of the sibling contain more than  $m/2$  elements, then create a new leaf node by joining two leaf nodes and the intervening element of the parent node.
5. If parent is left with less than  $m/2$  nodes then, apply the above process on the parent too.

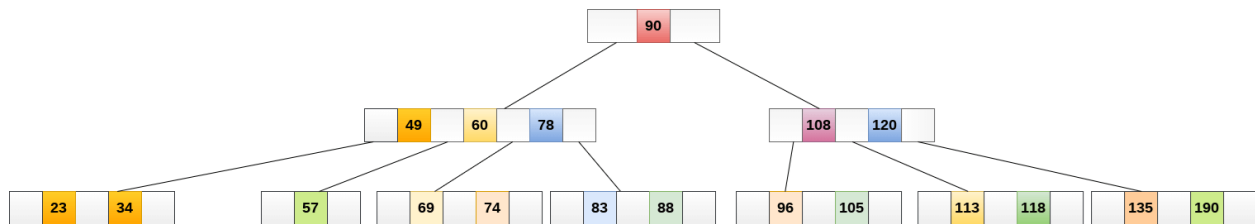
If the node which is to be deleted is an internal node, then replace the node with its in-order successor or predecessor. Since, successor or predecessor will always be on the leaf node hence, the process will be similar as the node is being deleted from the leaf node.

### **Example**

Delete the node 53 from the B Tree of order 5 shown in the following figure.



53 is present in the right child of element 49. Delete it.



Now, 57 is the only element, which is left in the node, the minimum number of elements that must be present in a B tree of order 5, is 2. it is less than that, the elements in its left and right sub-tree are also not sufficient therefore, merge it with the left sibling and intervening element of parent i.e. 49.

The final B tree is shown as follows.

