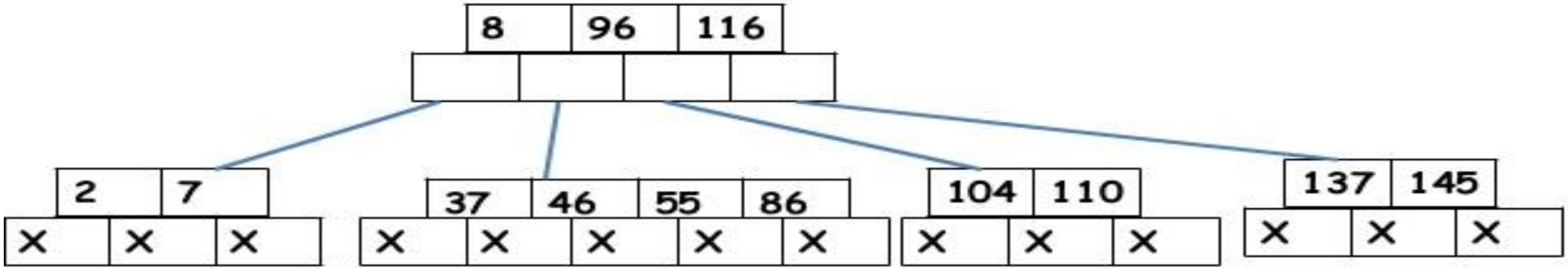
B-Trees Operation

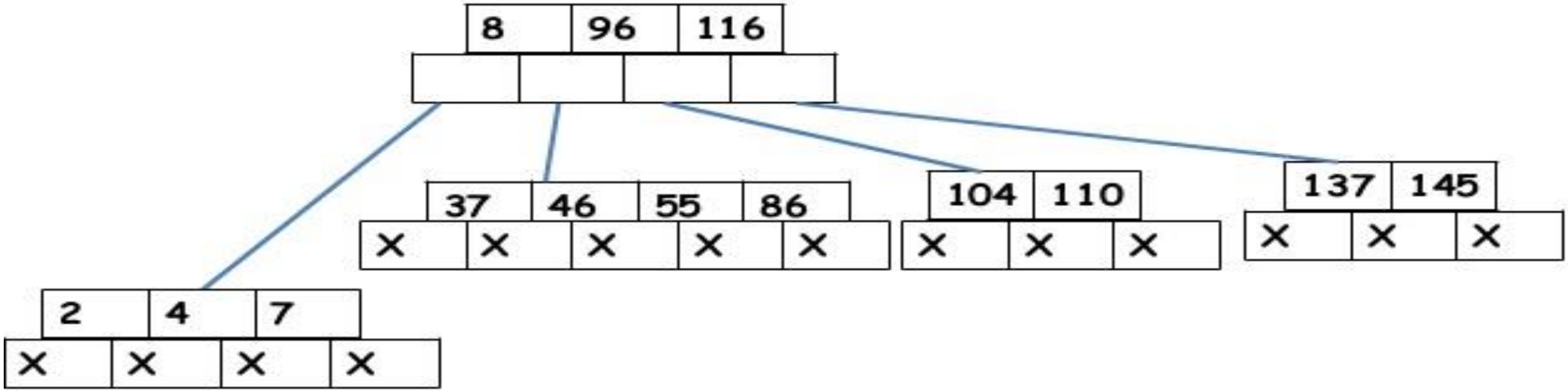
By: Aditya Tiwari
Assistant Professor
CSVTU, Bhilai

5-Way Search Tree



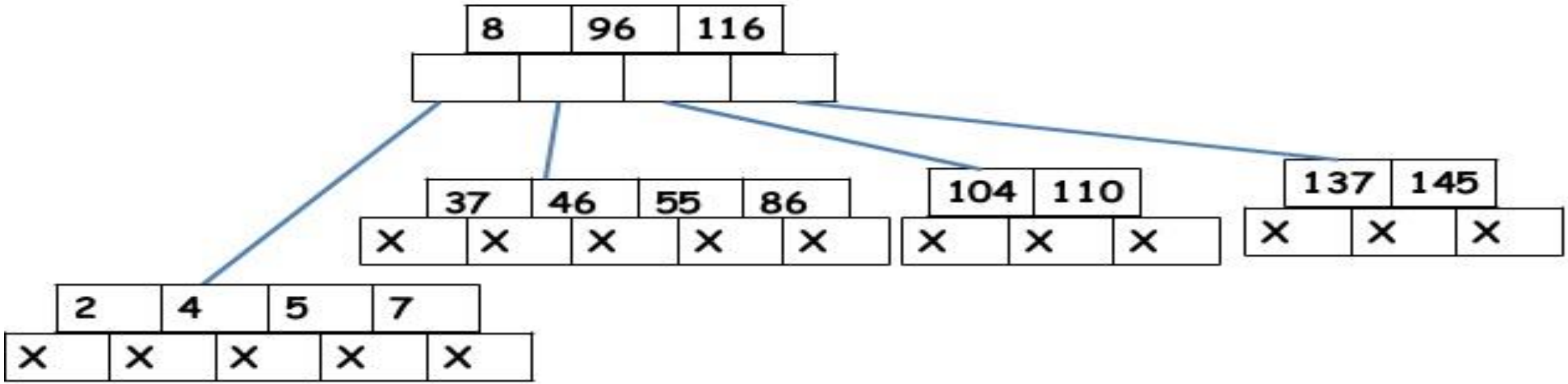
Insert 4, 5, 58, 6 in the order

5-Way Search Tree



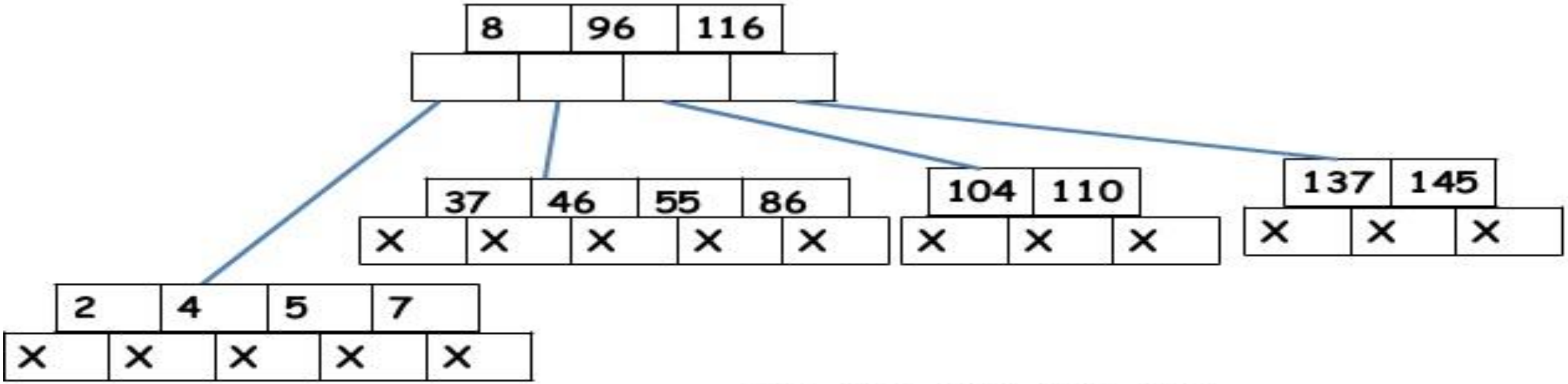
Search tree after inserting 4

5-Way Search Tree



Search tree after inserting 4, 5

5-Way Search Tree

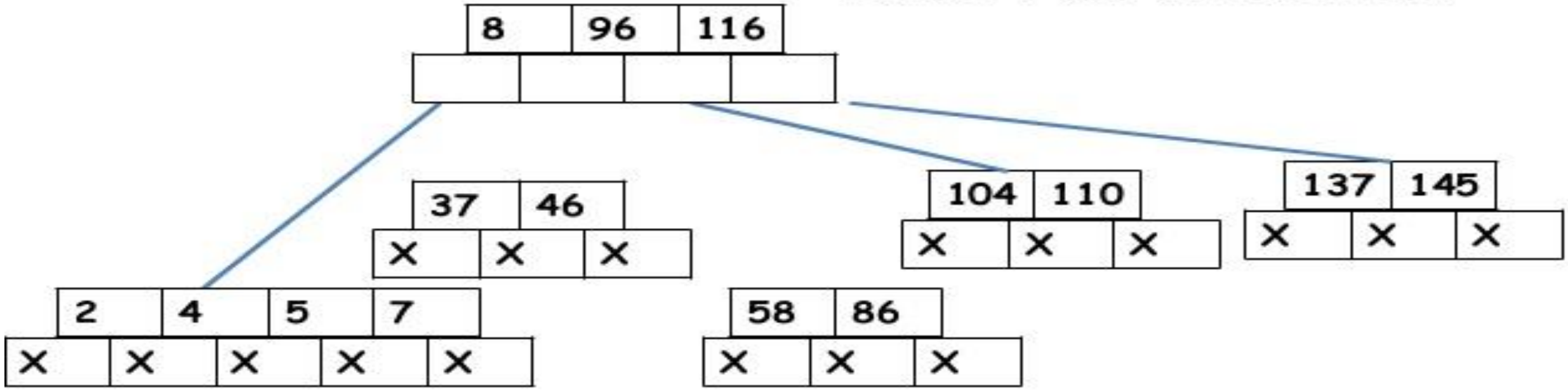


37,46,55,58,86

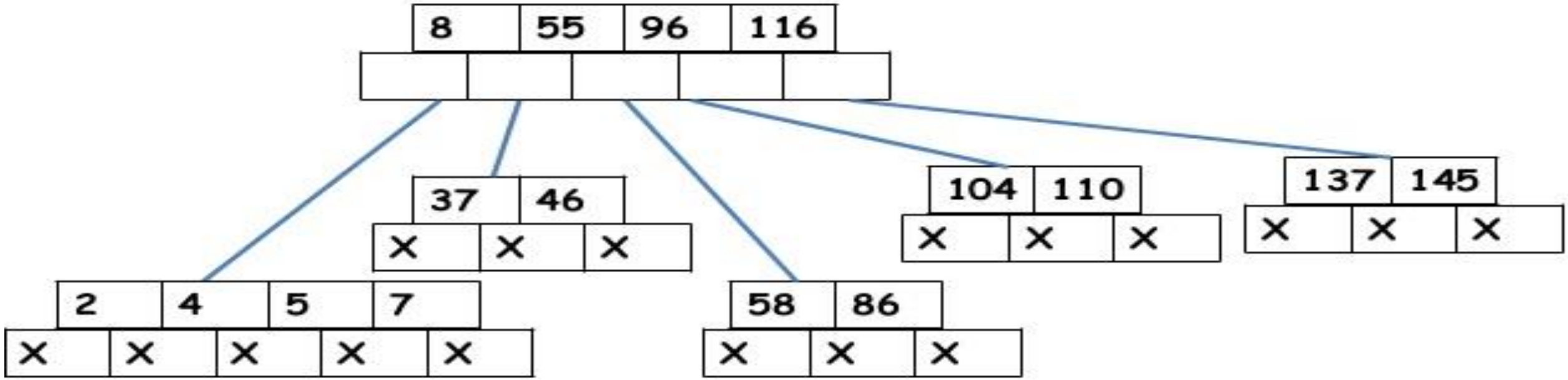
Split the node at its median into two node, pushing the median element up by one level

5-Way Search Tree

Insert 55 in the root

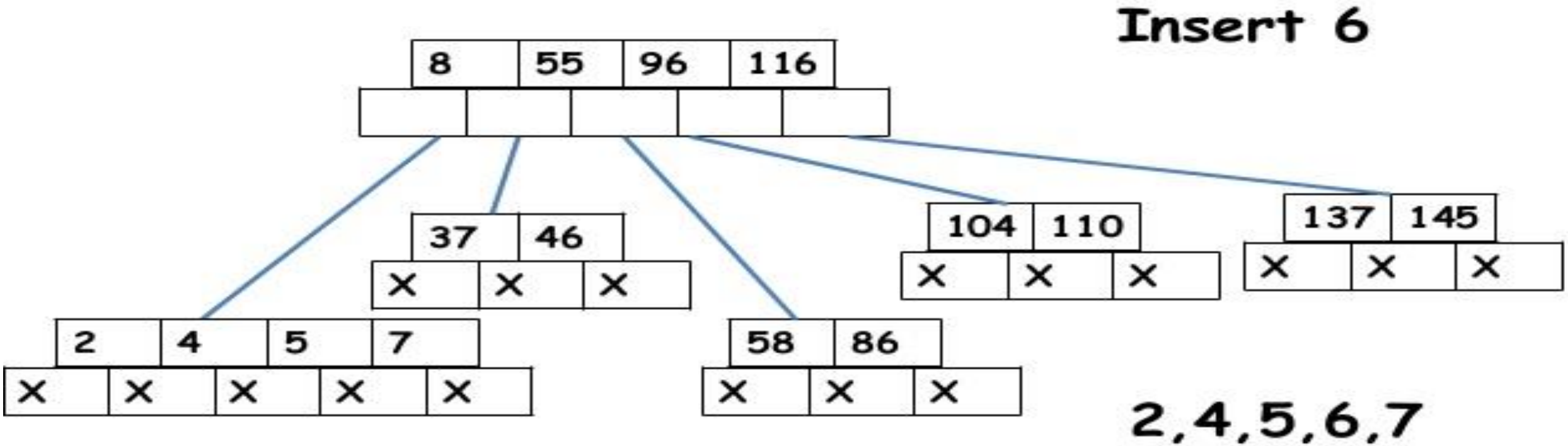


5-Way Search Tree



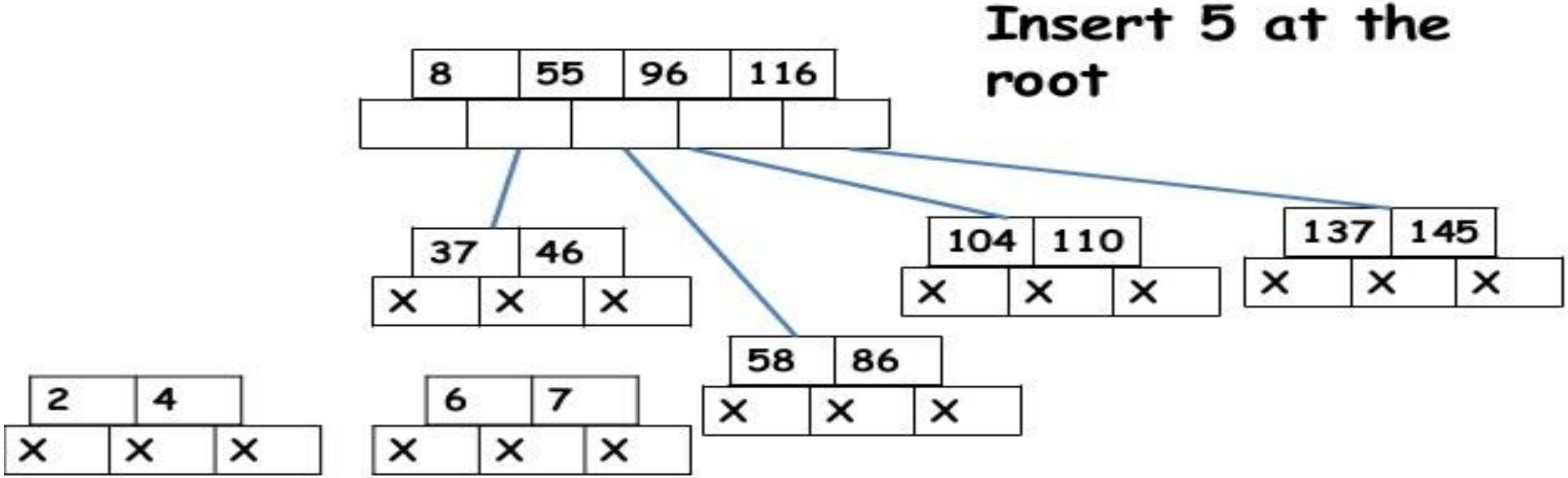
Search tree after inserting 4, 5, 58

5-Way Search Tree

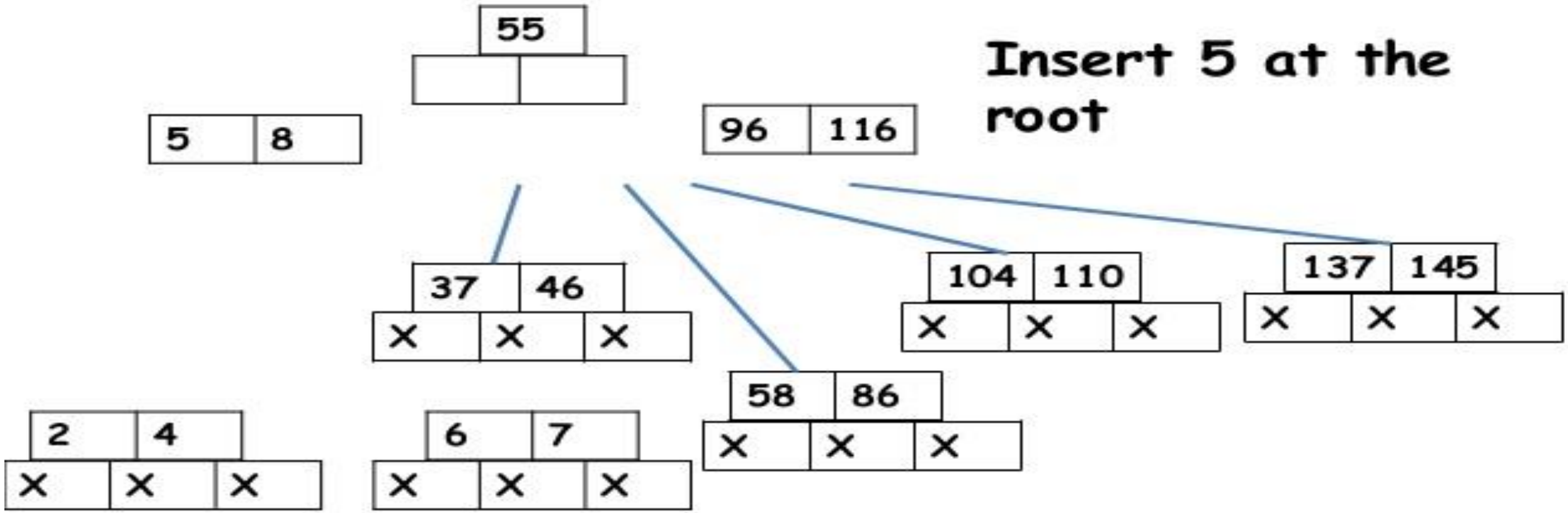


Split the node at its median into two node, pushing the median element up by one level

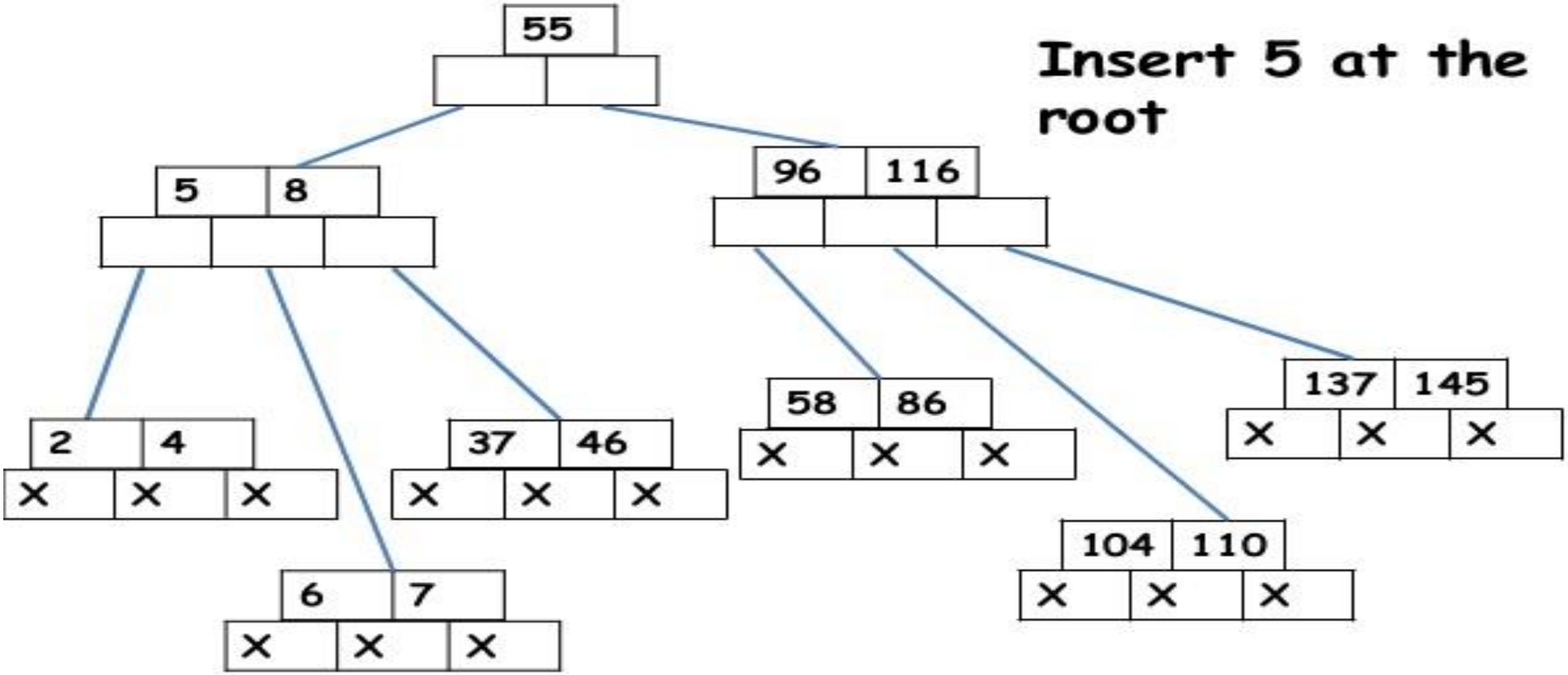
5-Way Search Tree



5-Way Search Tree



5-Way Search 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.

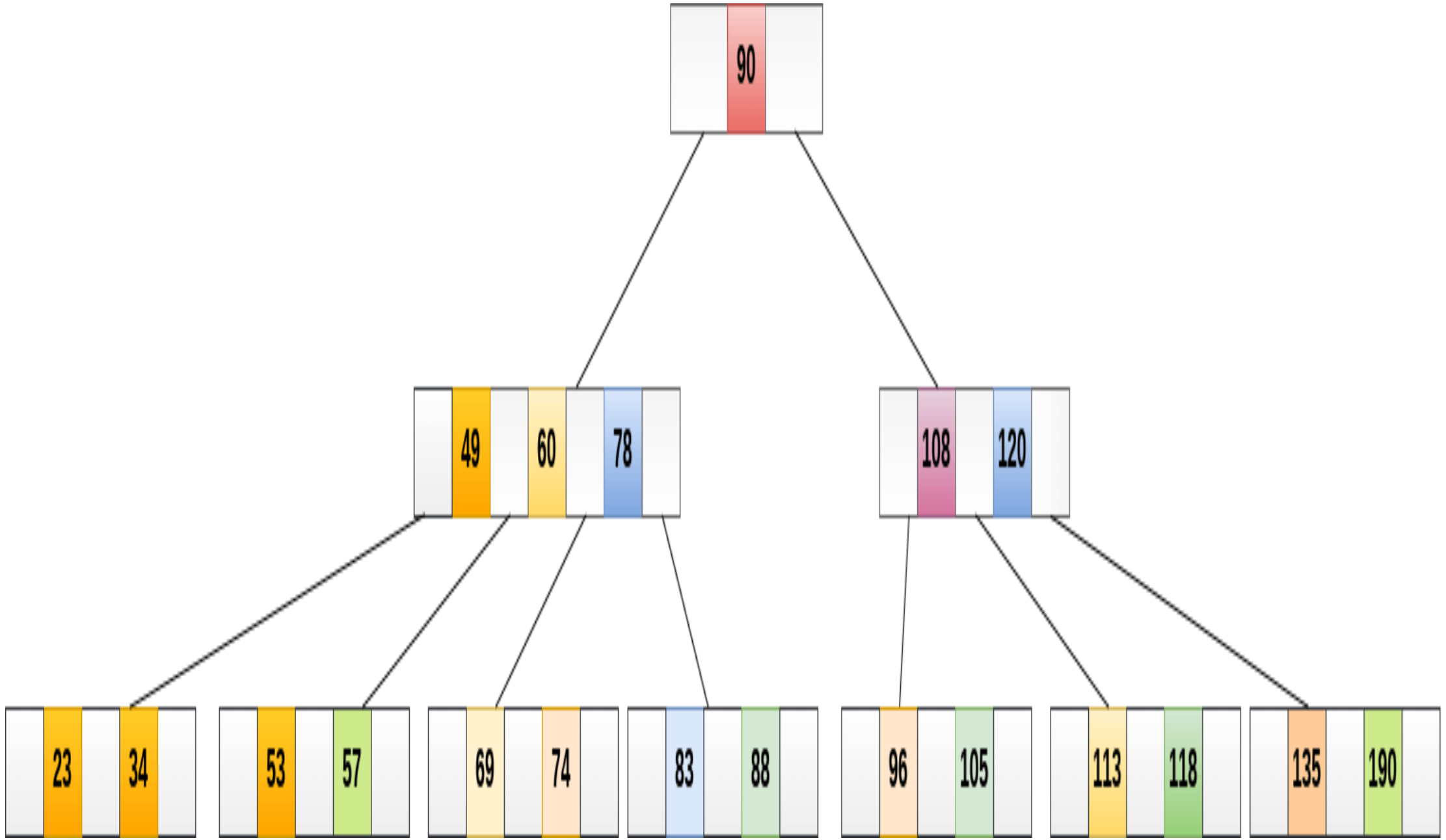
1. Locate the leaf node.
2. If there are more than m/2 keys in the leaf node then delete the desired key from the node.
3. If the leaf node doesn't contain m/2 keys then complete the keys by taking the element from right or left sibling.

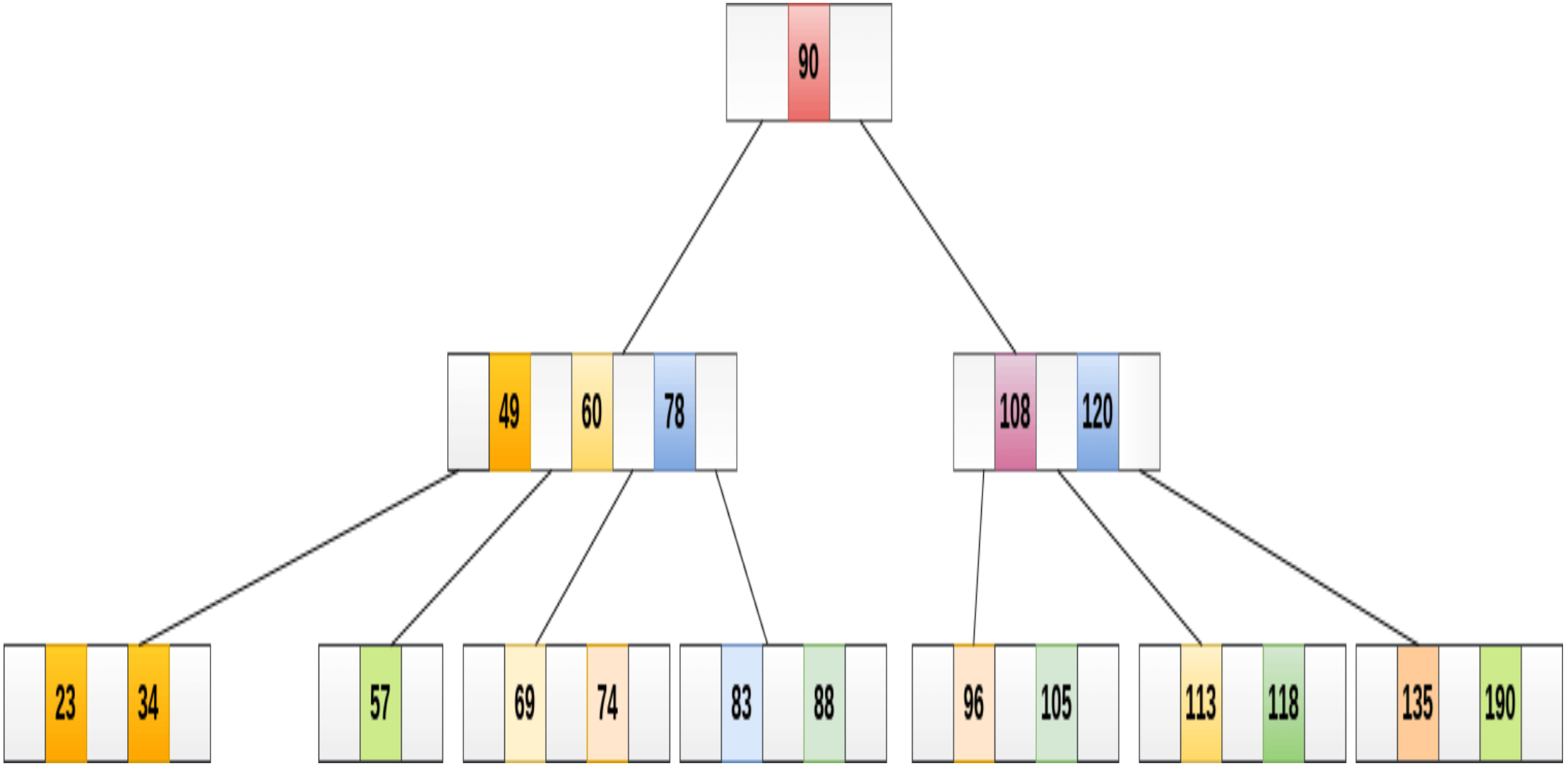
- o If the left sibling contains more than m/2 elements then push its largest element up to its parent and move the intervening element down to the node where the key is deleted.
- o If the right sibling contains more than m/2 elements then push its smallest element up to the parent and move intervening 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 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.

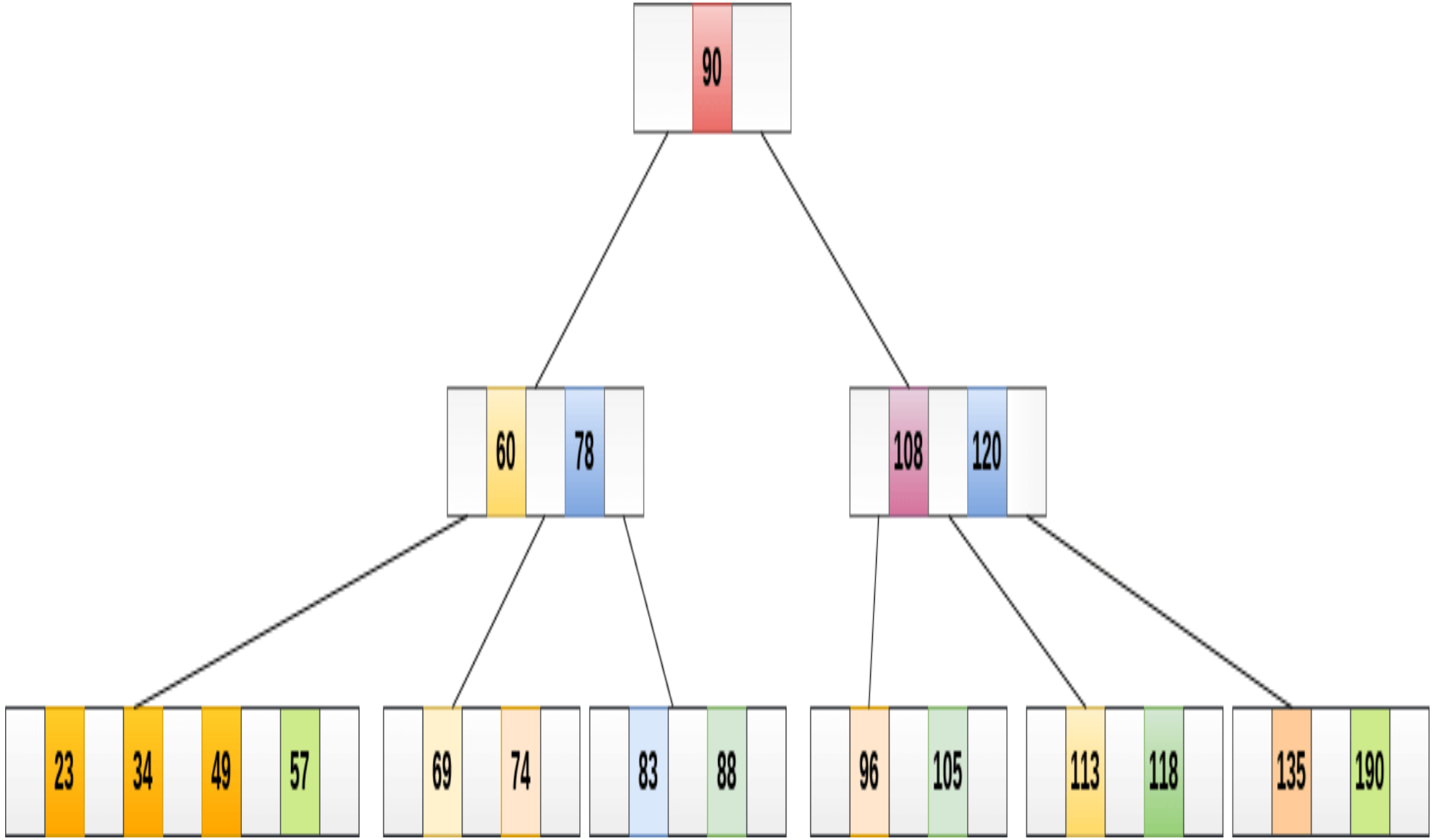
- Example 1
- Delete the node 53 from the B Tree of order 5 shown in the following figure.


90
49
60
78
108
120
23
34
53
57
69
74
83
88
96
105
113
118
135
190



- 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.


90
60
78
108
120
23
34
49
57
69
74
83
88
96
105
113
118
135
190

Deletion in a B-Tree

It is desirable that a key in leaf node be removed.

When a key in an internal node to be deleted, then we promote a successor or a predecessor of the key to be deleted ,to occupy the position of the deleted key and such a key is bound to occur in a leaf node.

Deletion in a B-Tree

Removing a key from leaf node:

If the node contain more than the minimum number of elements, then the key can be easily removed.

If the leaf node contain just the minimum number of elements, then look for an element either from the left sibling node or right sibling node to fill the vacancy.

Deletion in a B-Tree

If the left sibling has more than minimum number of keys, pull the largest key up into the parent node and move down the intervening entry from the parent node to the leaf node where key is deleted.

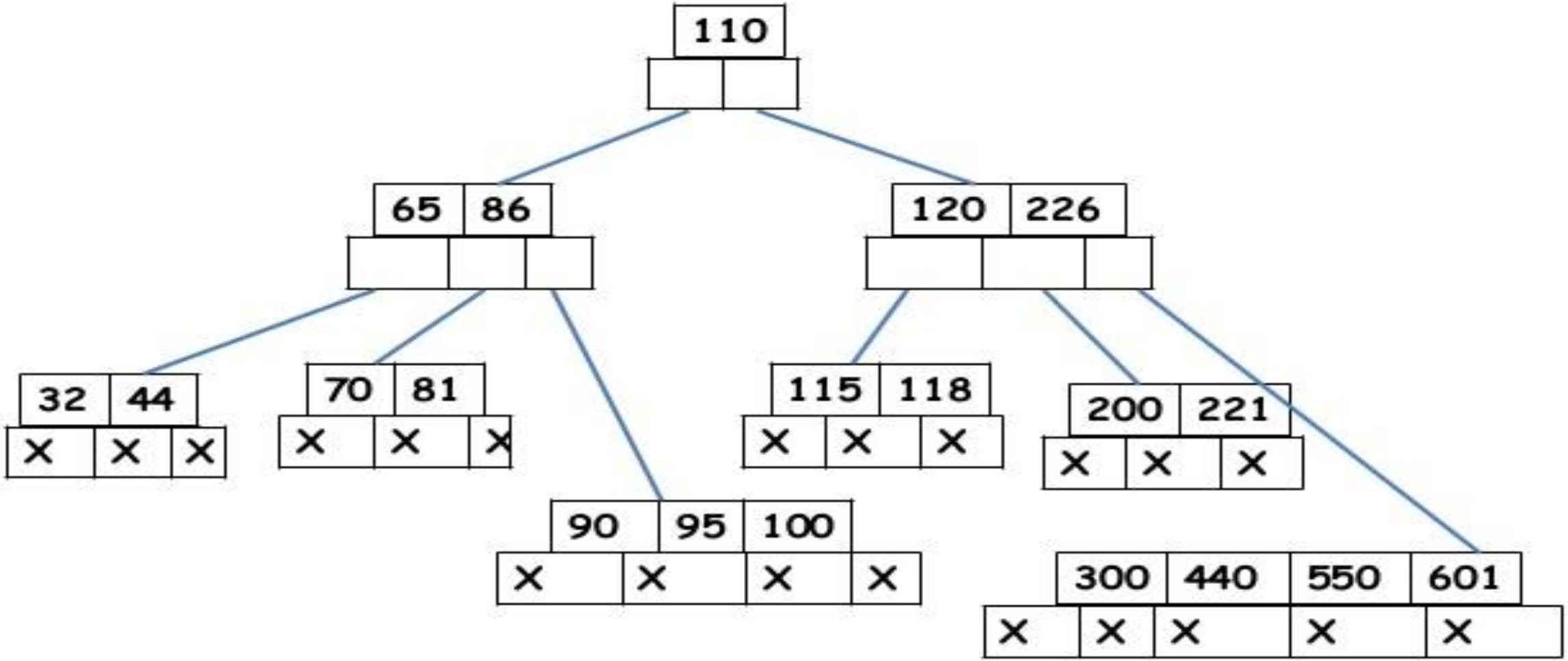
Otherwise, pull the smallest key of the right sibling node to the parent node and move down the intervening parent element to the leaf node.

Deletion in a B-Tree

If both the sibling node has minimum number of entries, then create a new leaf node out of the two leaf nodes and the intervening element of the parent node, ensuring the total number does not exceed the maximum limit for a node.

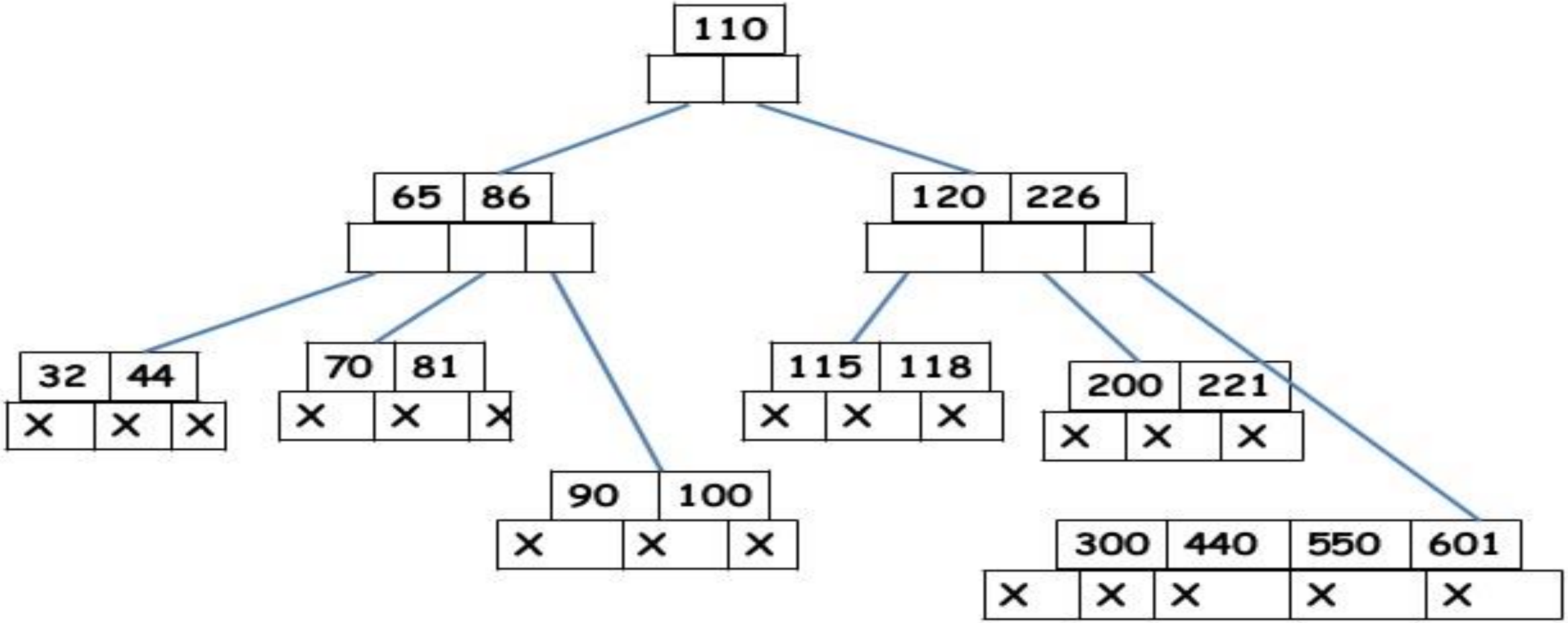
If while borrowing the intervening element from the parent node, it leaves the number of keys in the parent node to be below the minimum number, then we propagate the process upwards ultimately resulting in a reduction of the height of B-tree

B-tree of Order 5



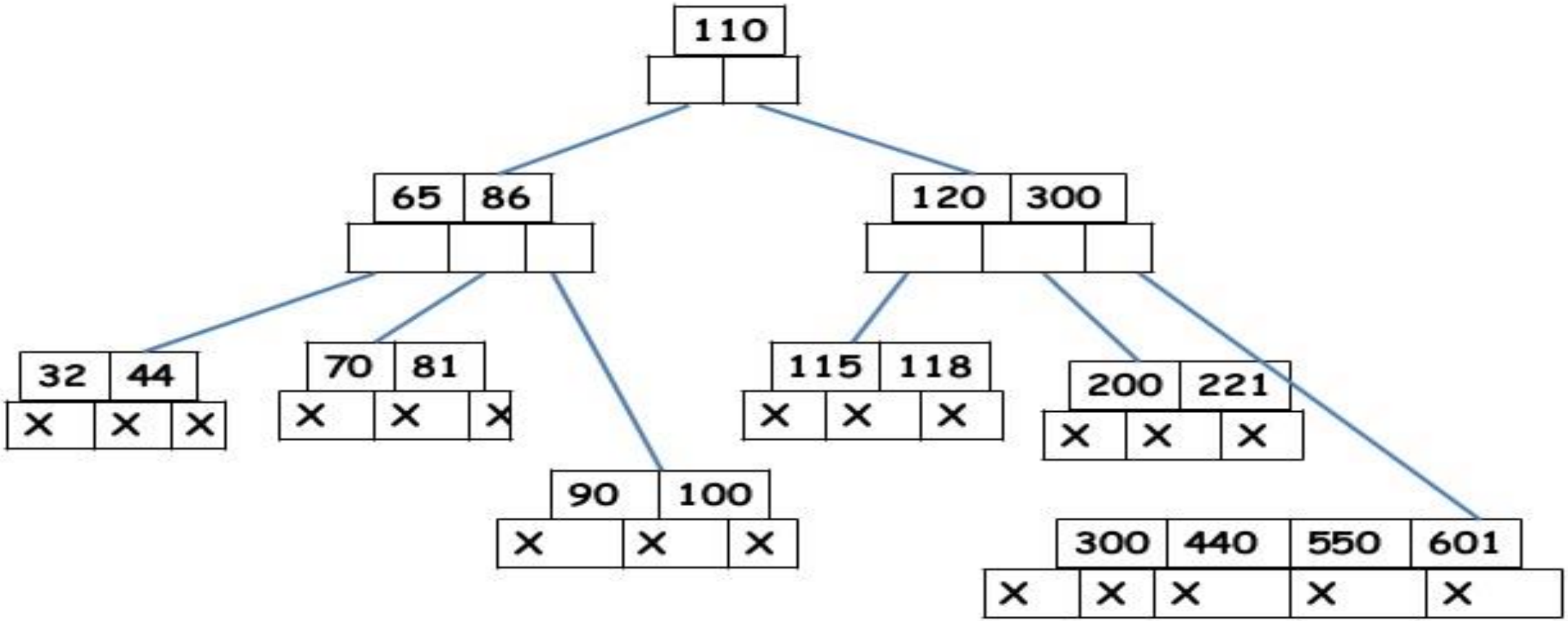
Delete 95, 226, 221, 70

B-tree of Order 5

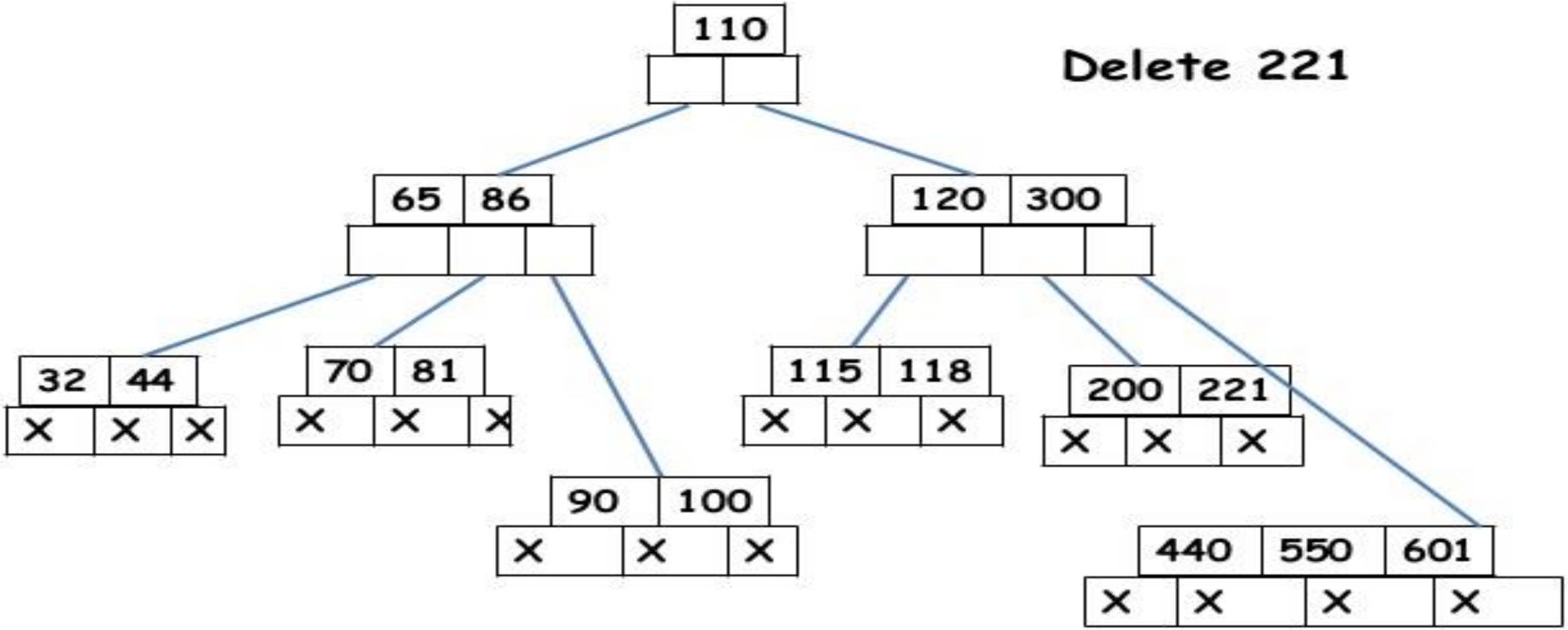


B-tree after deleting 95

B-tree of Order 5



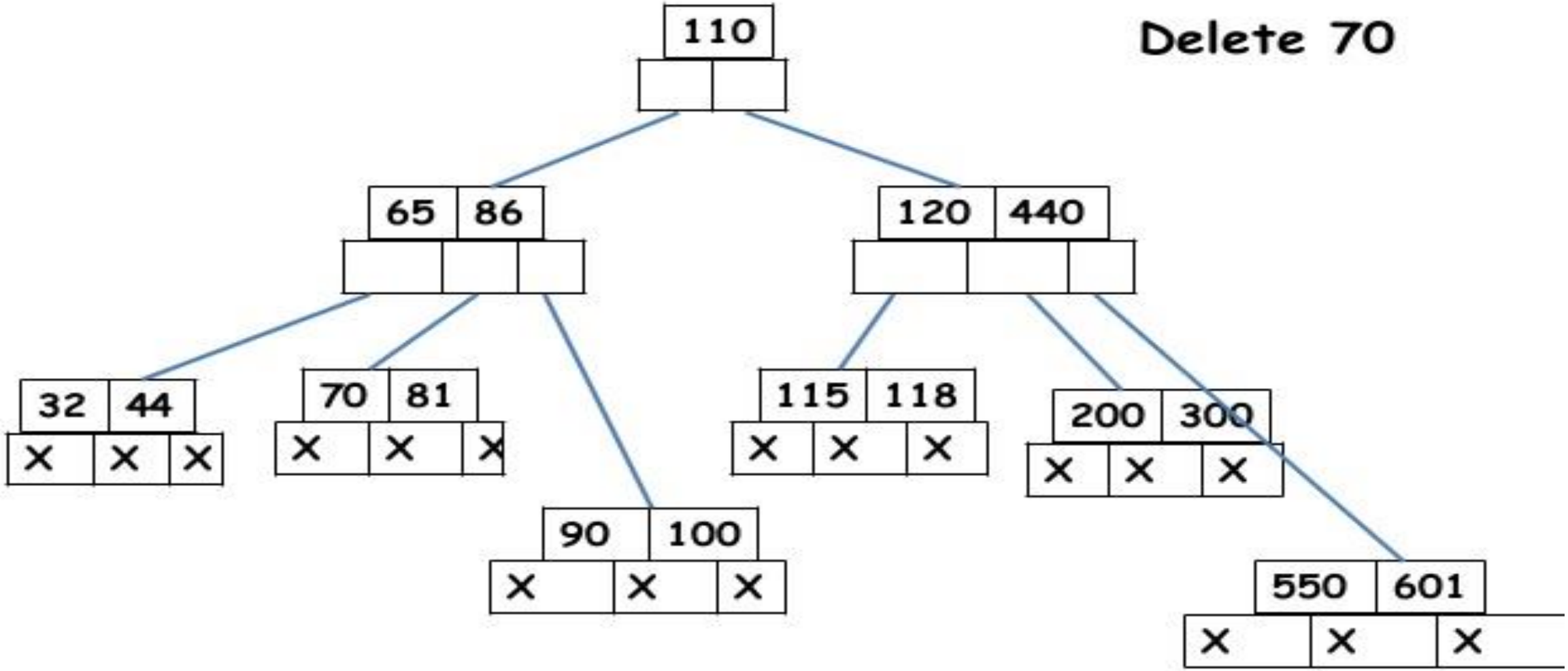
B-tree of Order 5



Delete 221

B-tree after deleting 95, 226

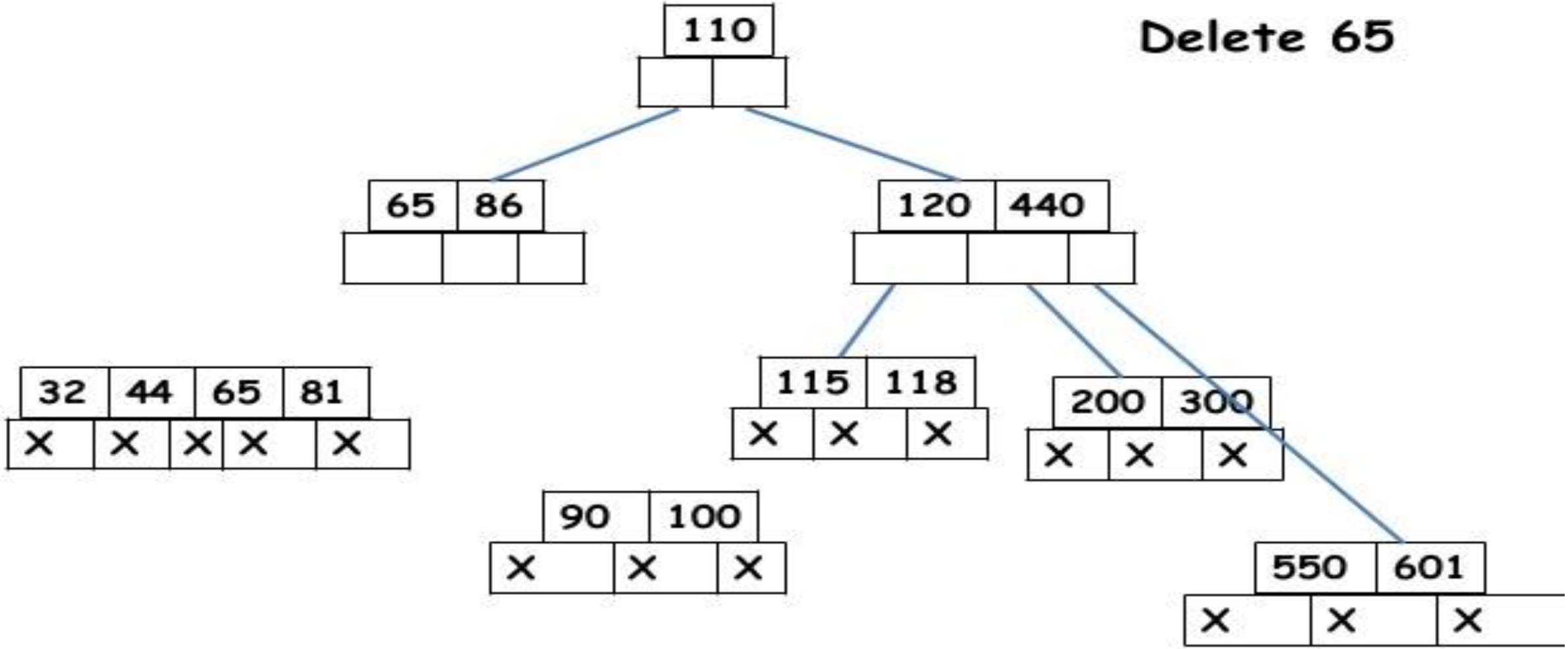
B-tree of Order 5



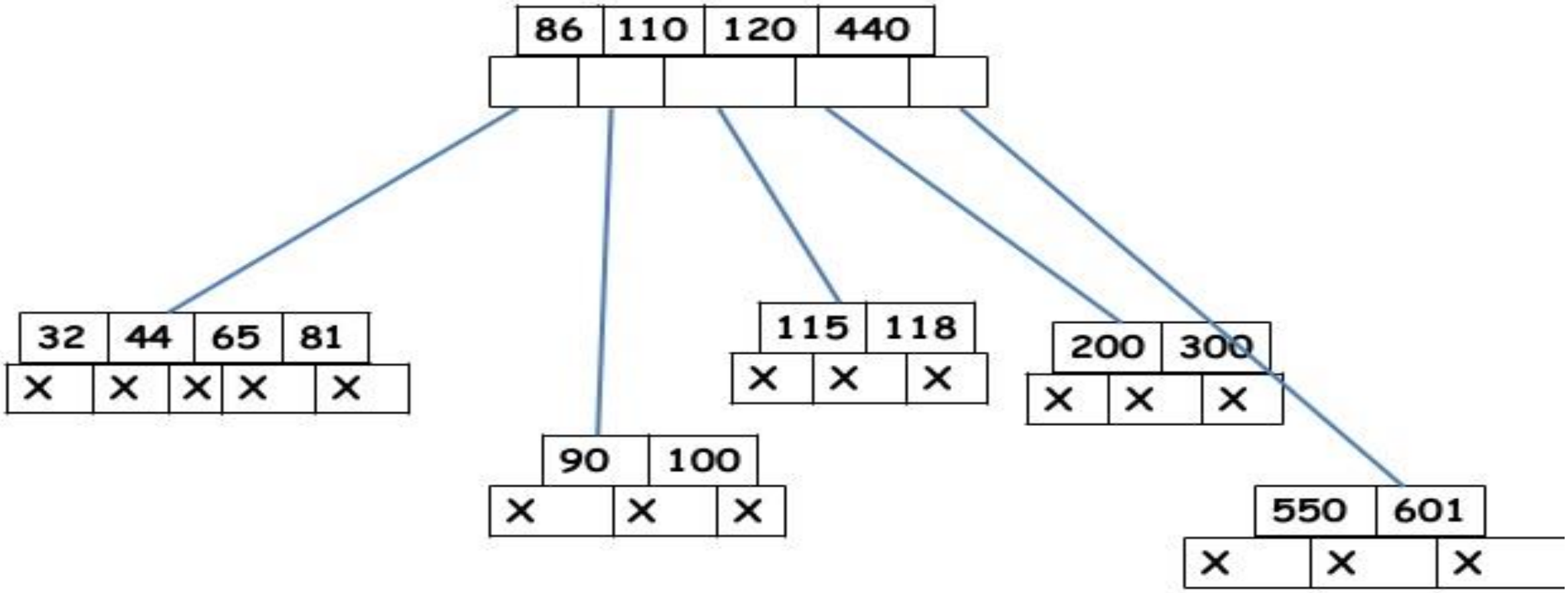
Delete 70

B-tree after deleting 95, 226, 221

B-tree of Order 5



B-tree of Order 5



B-tree after deleting 95, 226, 221, 70

Application of B tree

- B tree is used to index the data and provides fast access to the actual data stored on the disks since, the access to value stored in a large database that is stored on a disk is a very time consuming process.
- Searching an un-indexed and unsorted database containing n key values needs O(n) running time in worst case. However, if we use B Tree to index this database, it will be searched in O(log n) time in worst case.

- Thankyou