Deleting a Value From a B-Tree

Deletion from a B-tree is more complicated than insertion, because we can delete a key from any node-not just a leaf and when we delete a key from an internal node, we will have to rearrange the node's children.

There are three possible case for deletion in b tree.

Case-I target key is in the leaf node

- Target is in the leaf node, more than min keys.
  - Deleting this will not violate the property of B Tree
- Target is in leaf node, it has min key nodes
  - Deleting this will violate the property of B Tree
  - Target node can borrow key from immediate left node, or immediate right node (sibling)
  - The sibling will say yes if it has more than minimum number of keys
  - The key will be borrowed from the parent node, the max value will be transferred to a parent, the max value of the parent node will be transferred to the target node, and remove the target value
- Target is in the leaf node, but no siblings have more than min number of keys
  - Search for key
  - Merge with siblings and the minimum of parent nodes
  - Total keys will be now more than min
  - The target key will be replaced with the minimum of a parent node

Case 2- target key is in an internal node

- Either choose, in- order predecessor or in-order successor
- In case the of in-order predecessor, the maximum key from its left subtree will be selected
- In case of in-order successor, the minimum key from its right subtree will be selected
- If the target key's in-order predecessor has more than the min keys, only then it can replace the target key with the max of the in-order predecessor
- If the target key's in-order predecessor does not have more than min keys, look for in-order successor's minimum key.
- If the target key's in-order predecessor and successor both have less than min keys, then merge the predecessor and successor.

Case 3- target key is in a root node

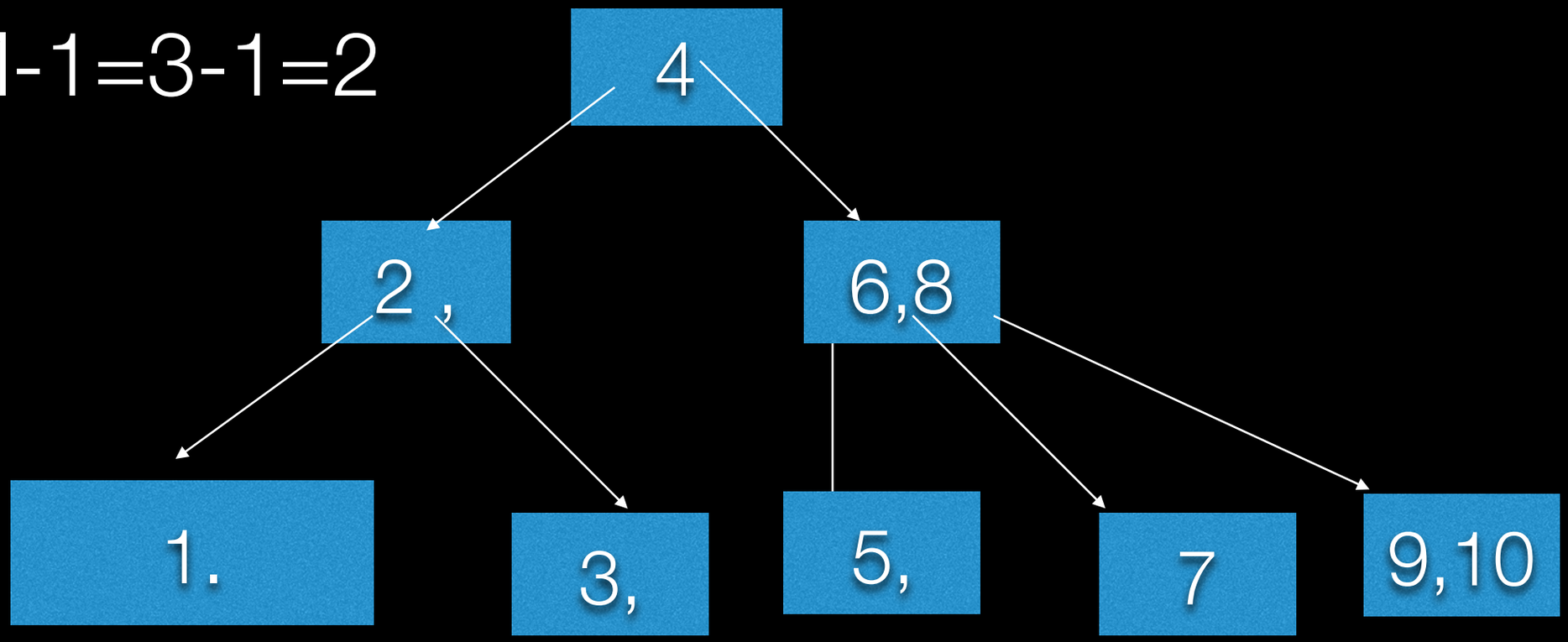
- Replace with the maximum element of the in-order predecessor subtree
- If, after deletion, the target has less than min keys, then the target node will borrow max value from its sibling via sibling's parent.
- The max value of the parent will be taken by a target, but with the nodes of the max value of the sibling.

1,2,3,4,5,6,7,8,9,10

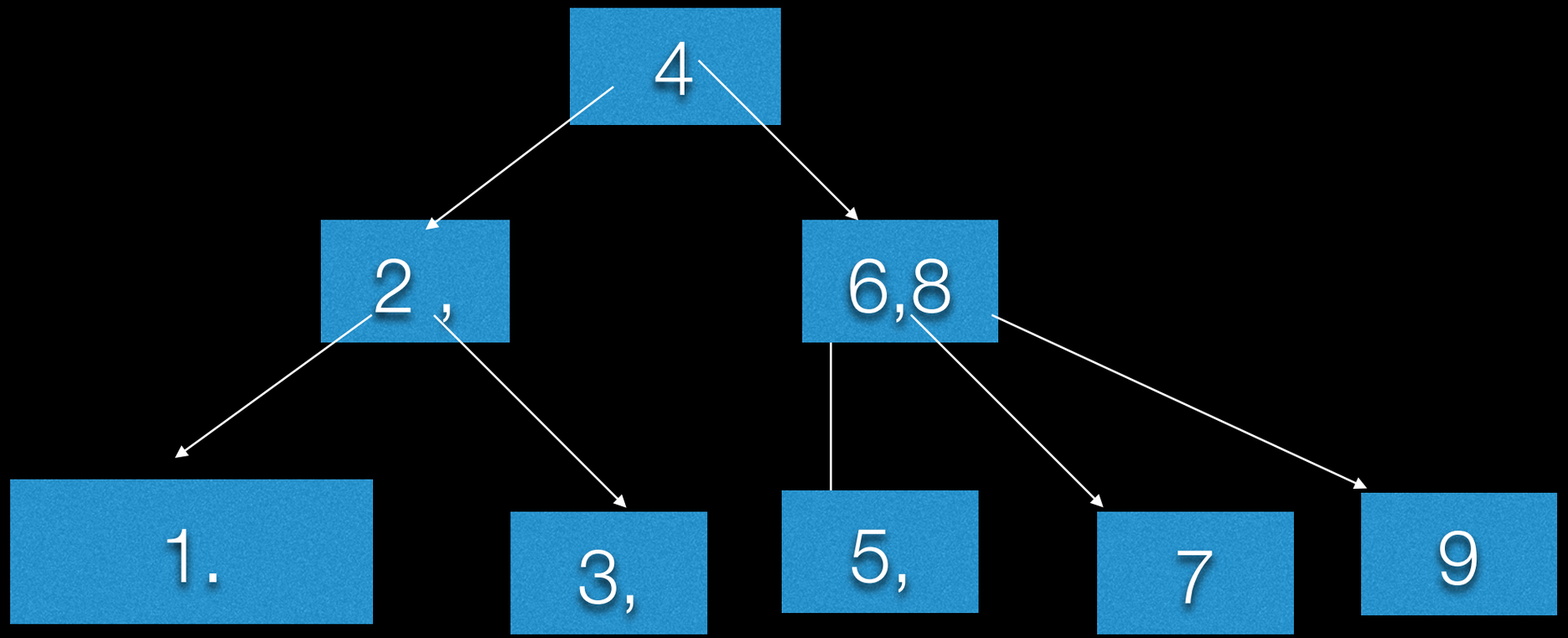
M=3

Max key M-1=3-1=2

IN Case Of Leaf Node

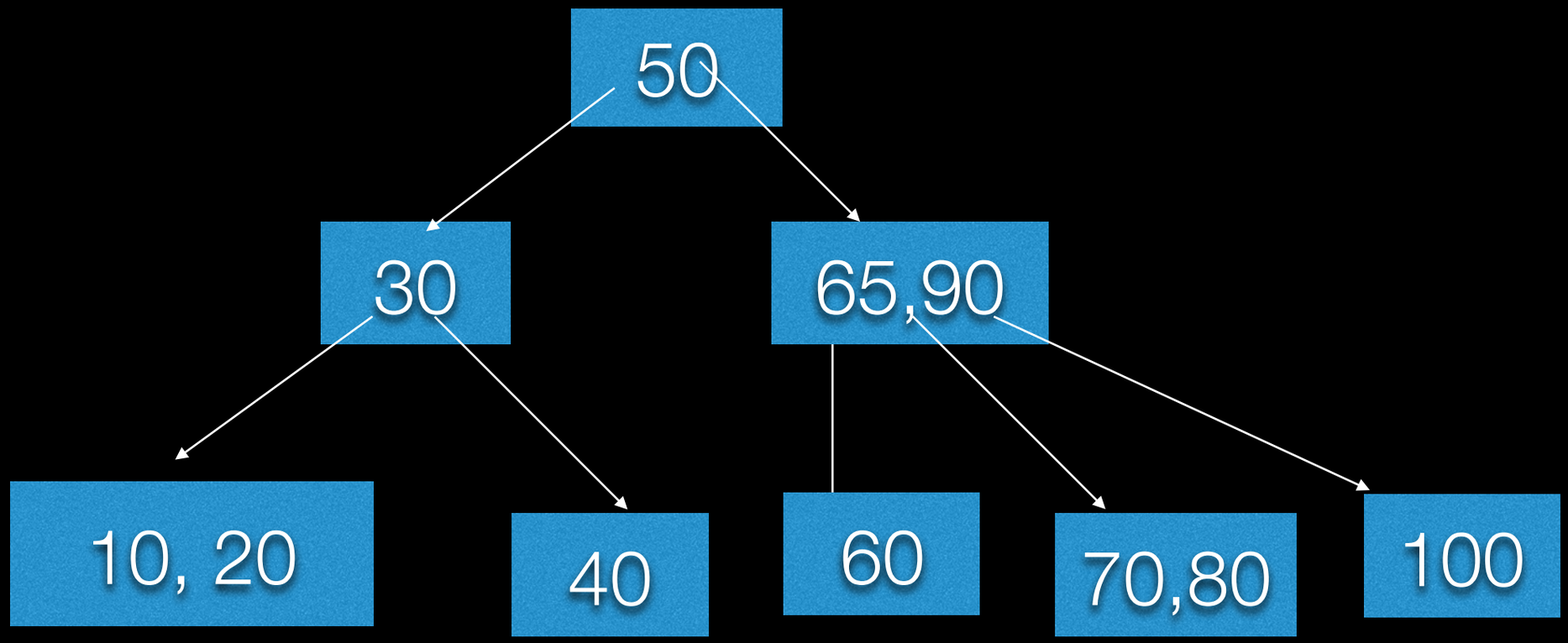


Delete 10


4
2 ,
6,8
1.
3,
5,
7
9

Internal Node Deletion

Delete 65



Replace with in- order predecessor or in-order successor

in- order predecessor of 65 = 60
in-order successor 0f 65 =70

Replace with in-order successor 0f 65 =70