

# Multi-way Search Trees

Kuan-Yu Chen (陳冠宇)

2018/10/24 @ TR-212, NTUST

# Review

---

- Red-Black Trees
- Splay Trees
- Huffman Trees

# Multi-way Search Trees.

---

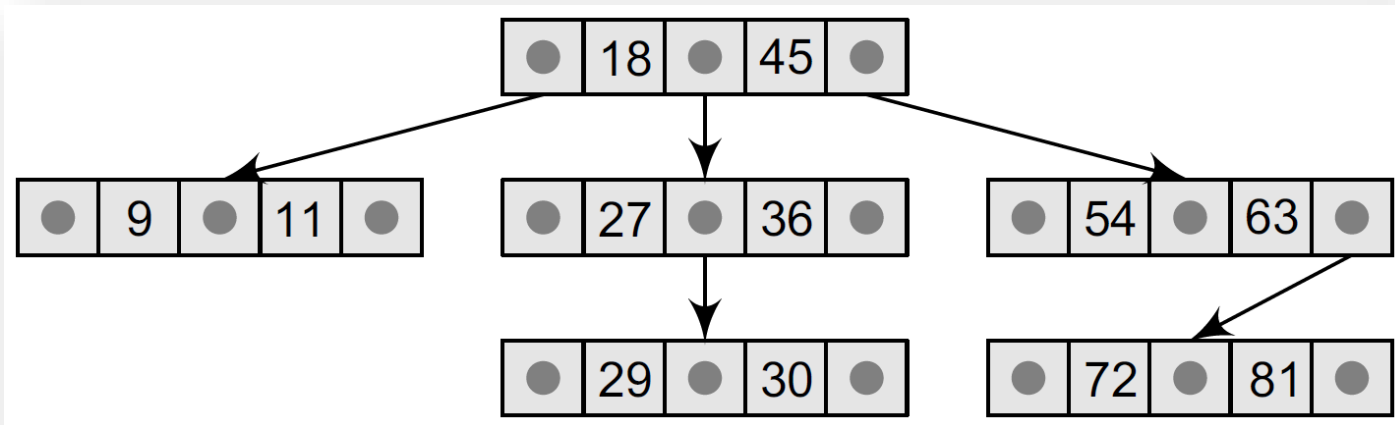
- Every node in a binary search tree contains one value and two pointers, left and right, which point to the node's left and right sub-trees

|                             |                             |                              |
|-----------------------------|-----------------------------|------------------------------|
| Pointer to<br>left sub-tree | Value or Key<br>of the node | Pointer to<br>right sub-tree |
|-----------------------------|-----------------------------|------------------------------|

- An  $M$ -way search tree has  $M - 1$  values per node and  $M$  subtrees
  - $M$  is called the degree of the tree
  - If  $M = 2$ , each node in the  $M$ -way search tree has one value and two sub-trees
    - Binary Search Tree!

# Multi-way Search Trees..

- For a M-way search tree
  - All the key values are stored in ascending order
    - 3-way search tree



- It is not compulsory that every node has exactly  $M-1$  values and  $M$  subtrees
  - The node can have anywhere from 1 to  $M-1$  values
  - The number of sub-trees can vary from 0 (leaf node) to  $i + 1$ 
    - $i$  is the number of key values in the node
    - $M$  is a fixed upper bound,  $(i + 1) \leq M$

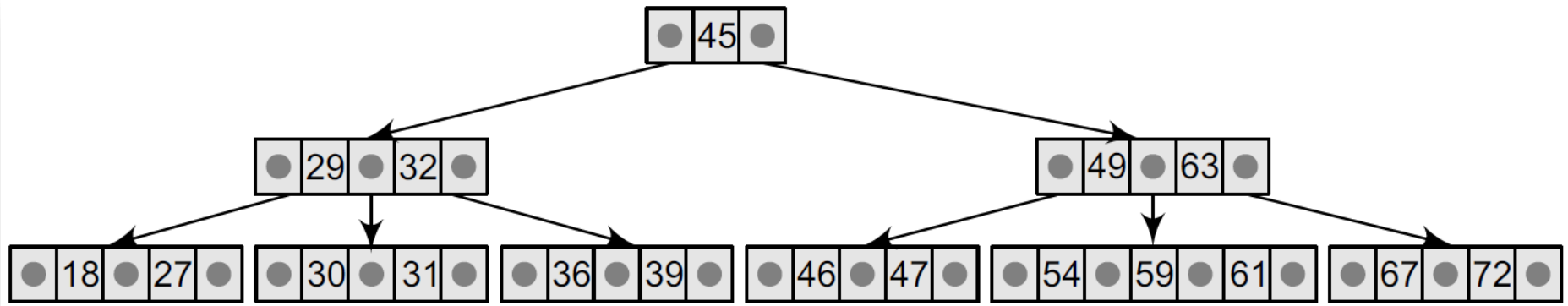
# B Trees.

---

- A B tree is a specialized M-way tree developed by Rudolf Bayer and Ed McCreight in 1970
  - A B tree of order  $m$  can have a maximum of  $m-1$  keys and  $m$  pointers to its sub-trees
- A B tree of order  $m$  is a tree with all the properties of an M-way search tree and has additional properties
  - Every node in the B tree has at most (maximum)  $m$  children
  - Every node in the B tree except the root node and leaf nodes has at least (minimum)  $\left\lceil \frac{m}{2} \right\rceil$  children
    - Degree=4, at least 2 children, at least 1 key
    - Degree=5, at least 3 children, at least 2 key
  - The root node has at least two children
  - All leaf nodes are at the same level

# B Trees..

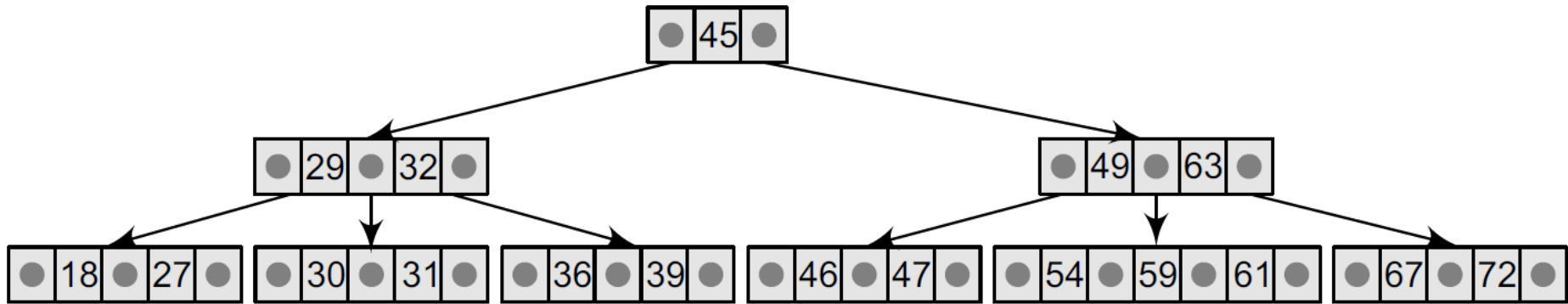
- A B tree of order 4



- While performing insertion and deletion operations in a B tree, the number of child nodes may change
  - The internal nodes may be joined or split to maintain a minimum number of children

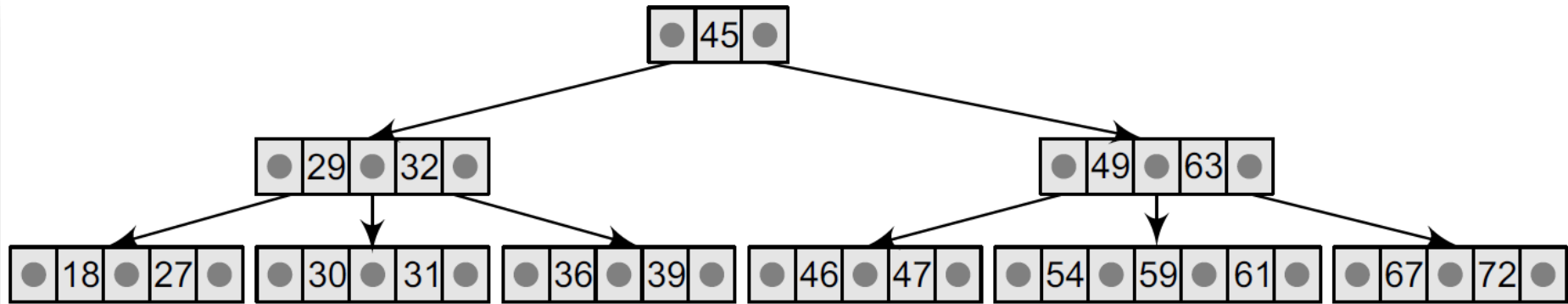
# Searching in a B Tree.

---



- To search for 59
  - The root node has a value 45 which is less than 59
    - Go to right sub-tree
  - The right sub-tree of the root node has two key values, 49 and 63
    - Since  $49 < 59 < 63$ , traverse the right sub-tree of 49, or the left sub-tree of 63
  - This sub-tree has three values, 54, 59, and 61
    - Terminal

# Searching in a B Tree..



- To search for 9
  - Traverse the left sub-tree of the root node
  - The left sub-tree has two key values, 29 and 32
    - Traverse the left sub-tree of 29
  - The sub-tree has two key values, 18 and 27
    - There is no left sub-tree of 18
    - The value 9 is not stored in the tree



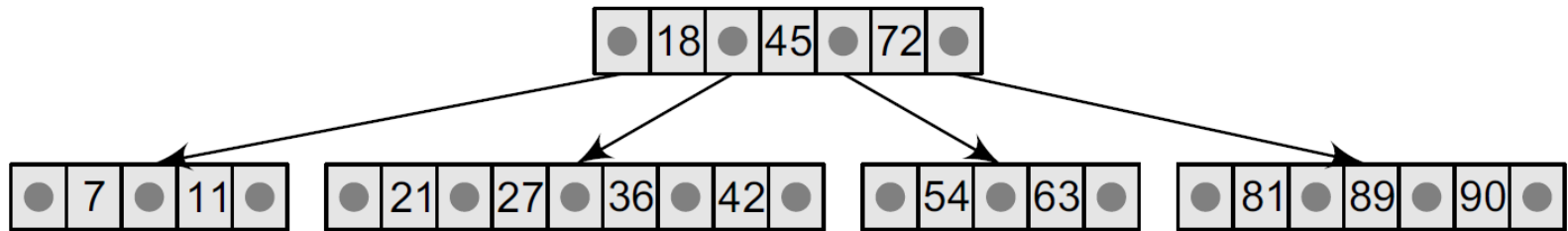
# Inserting a New Element

---

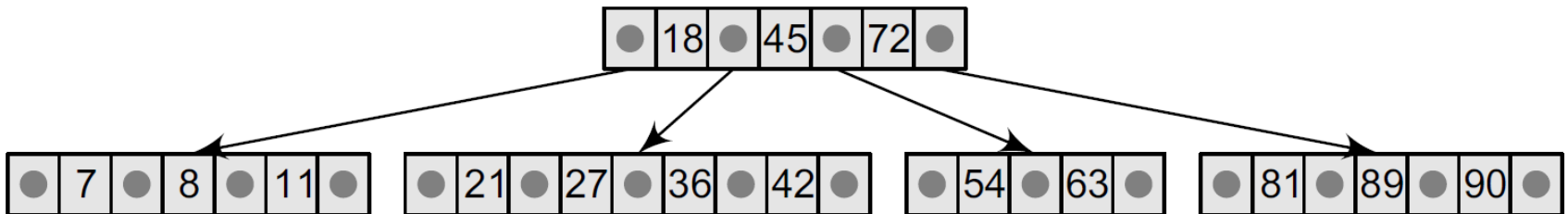
- In a B tree, all insertions are done at the leaf node level
  1. Search the B tree to find the leaf node where the new key value should be inserted
  2. If the leaf node is not full
    - Insert the new element in the node keeping the node's elements ordered
  3. If the leaf node is full
    - Insert the new value in order into the existing set of keys
    - Split the node at its median into two nodes
      - The split nodes are half full
    - Push the median element up to its parent's node
      - If the parent's node is already full
      - Split the parent node by the same steps

# Example.

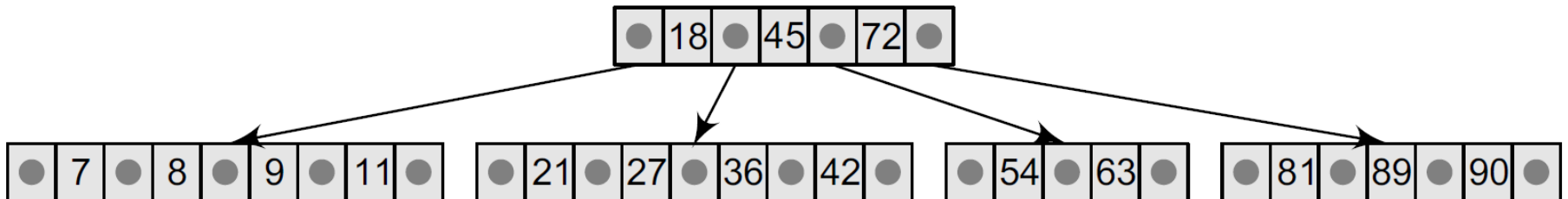
- Given a B tree of order 5, please insert 8, 9, 39, and 4 into it



**Step 1: Insert 8**

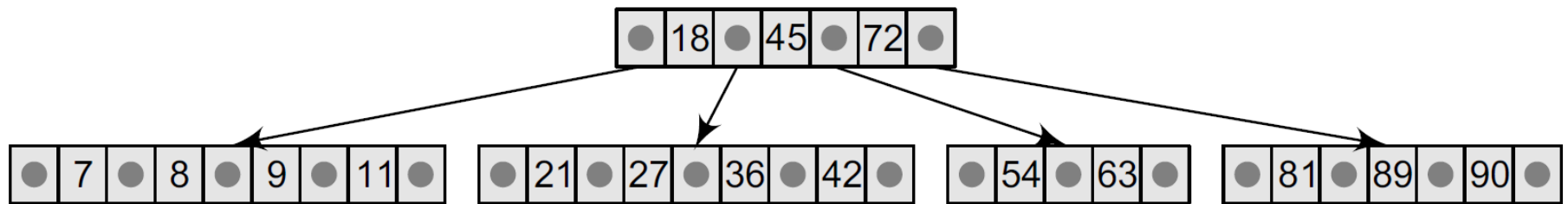


**Step 2: Insert 9**

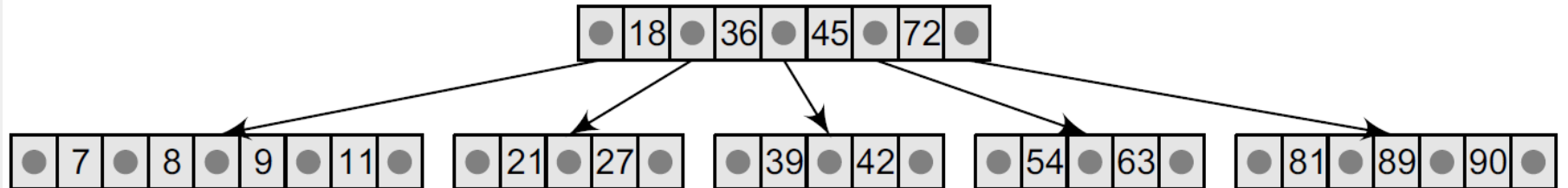


# Example..

- Given a B tree of order 5, please insert 8, 9, 39, and 4 into it

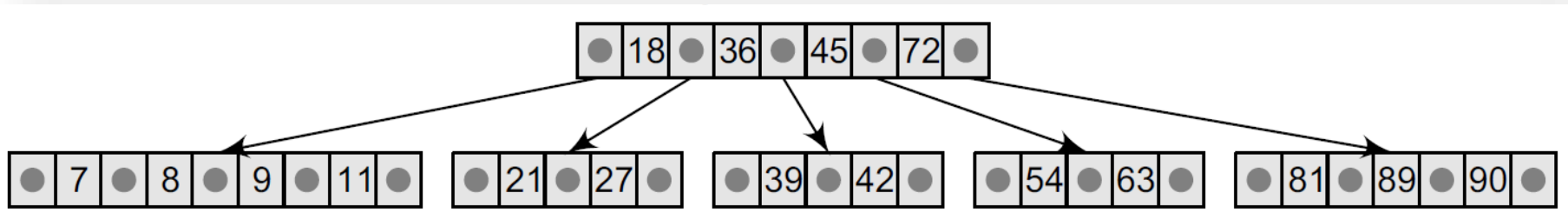


## Step 3: Insert 39

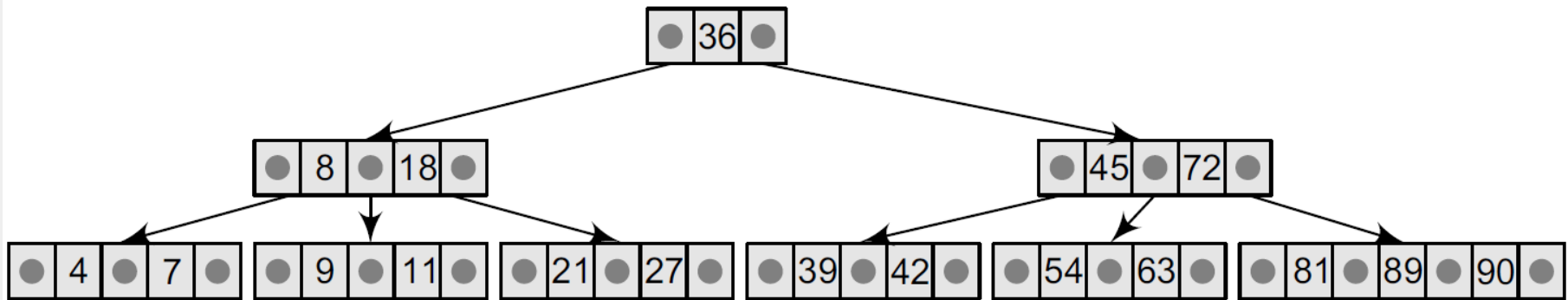


# Example...

- Given a B tree of order 5, please insert 8, 9, 39, and 4 into it



Step 4: Insert 4



# Deleting a New Element.

---

- There are two cases of deletion
  - A leaf node has to be deleted
  - An internal node has to be deleted
    - Promote the successor or predecessor of the key **in the leaf node** to be deleted to occupy the position of the deleted key
      - The processing will be done as if a value from the leaf node has been deleted

# Deleting a New Element..

---

- A leaf node has to be deleted
  - Locate the leaf node which has to be deleted
  - If the leaf node contains more than the minimum number of key values, then delete the value
  - If the leaf node does not contain the minimum number elements, then fill the node by taking an element either from the left or from the right sibling
    - If the left sibling has more than the minimum number of key values
      - ❑ push its largest key into its parent's node
      - ❑ pull down the suitable (intervening) element from the parent node to replace the deleted element
    - If the right sibling has more than the minimum number of key values
    - If both left and right siblings contain only the minimum number of elements

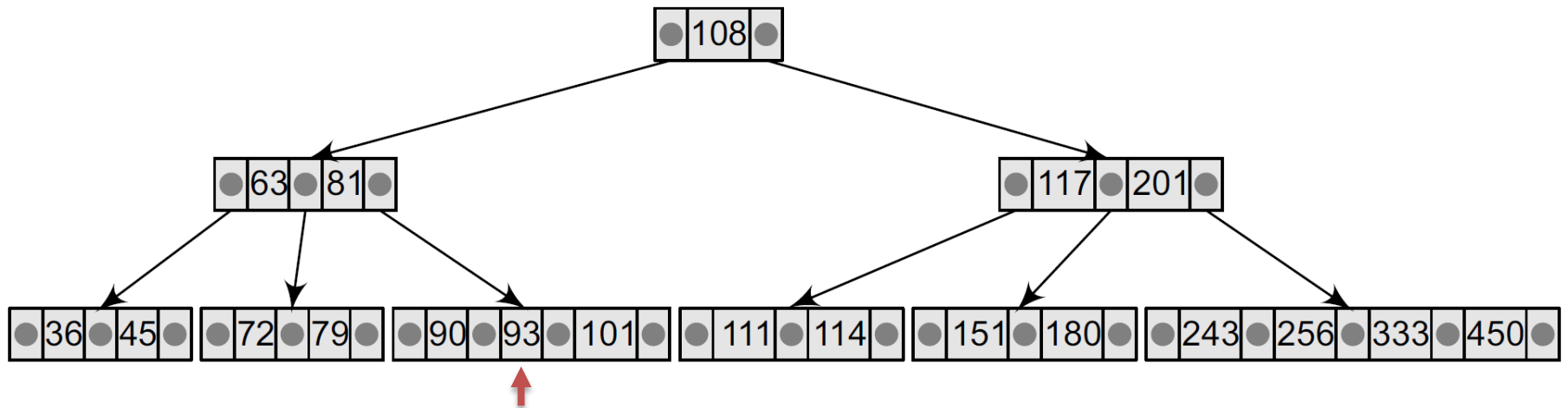
# Deleting a New Element...

---

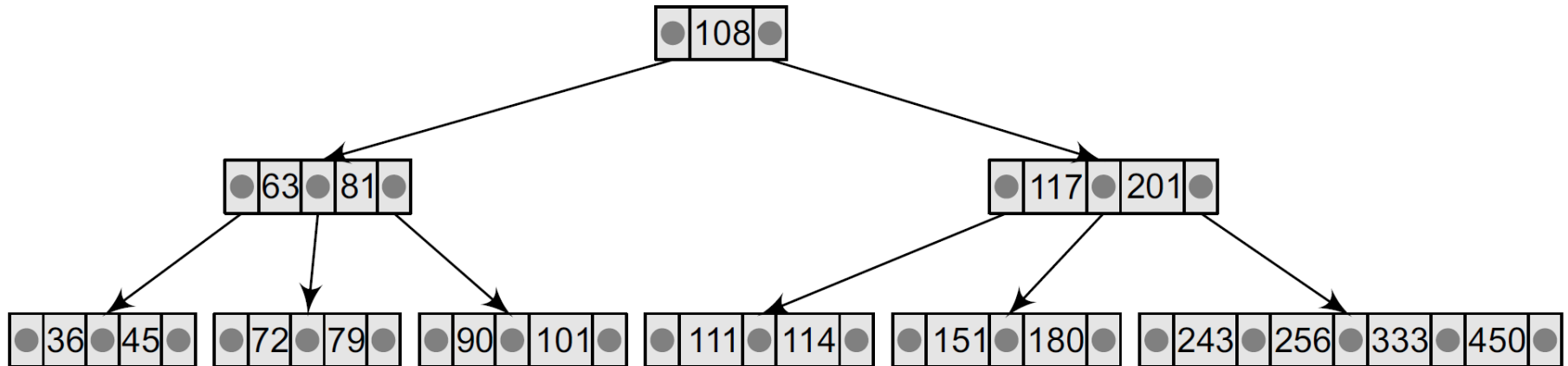
- If the left sibling has more than the minimum number of key values
- If the right sibling has more than the minimum number of key values
  - ❑ push its smallest key into its parent's node
  - ❑ pull down the suitable (intervening) element from the parent node to replace the deleted element
- If both left and right siblings contain only the minimum number of elements
  - ❑ create a new leaf node by combining the two leaf nodes (target+left or target+right) and the intervening element of the parent node
  - ❑ if the parent node contains less than the minimum number of keys in the node
    - ✓ propagate the process upwards, thereby reducing the height of the B tree

# Example – 1.

- Given a B tree of order 5, please delete values 93, 201, 180, and 72



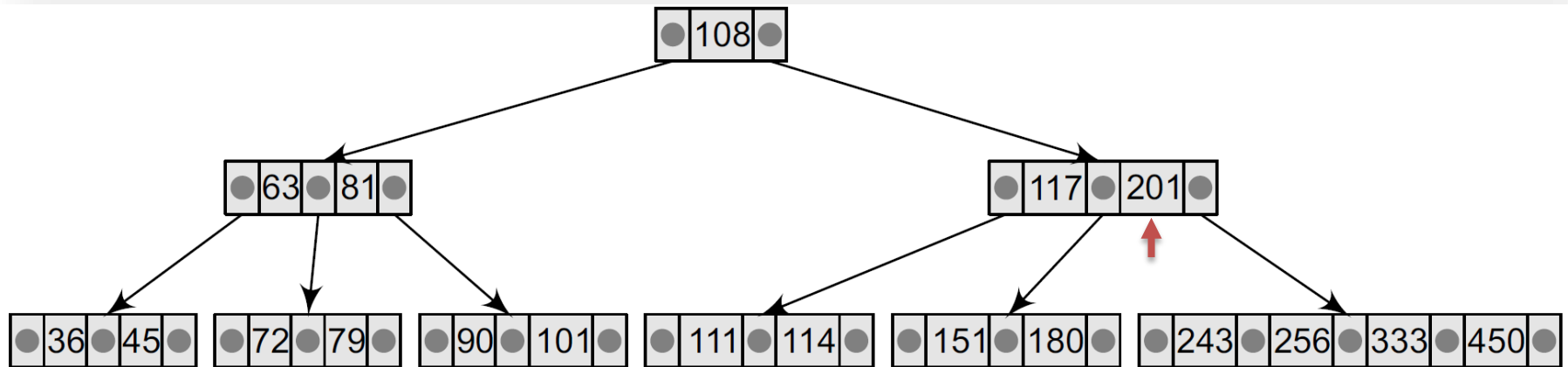
**Step 1: Delete 93**



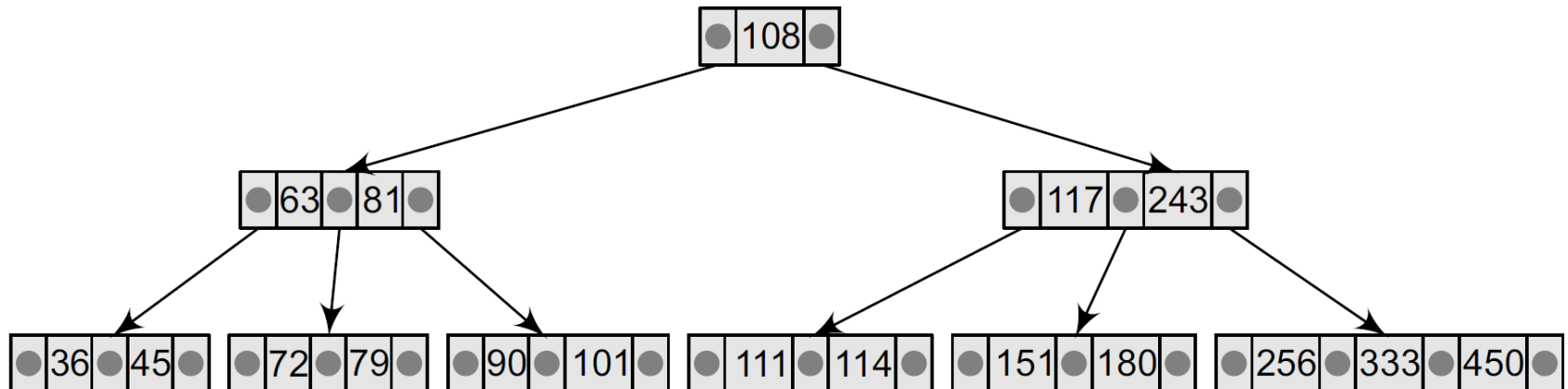


# Example – 1..

- Given a B tree of order 5, please delete values 93, 201, 180, and 72

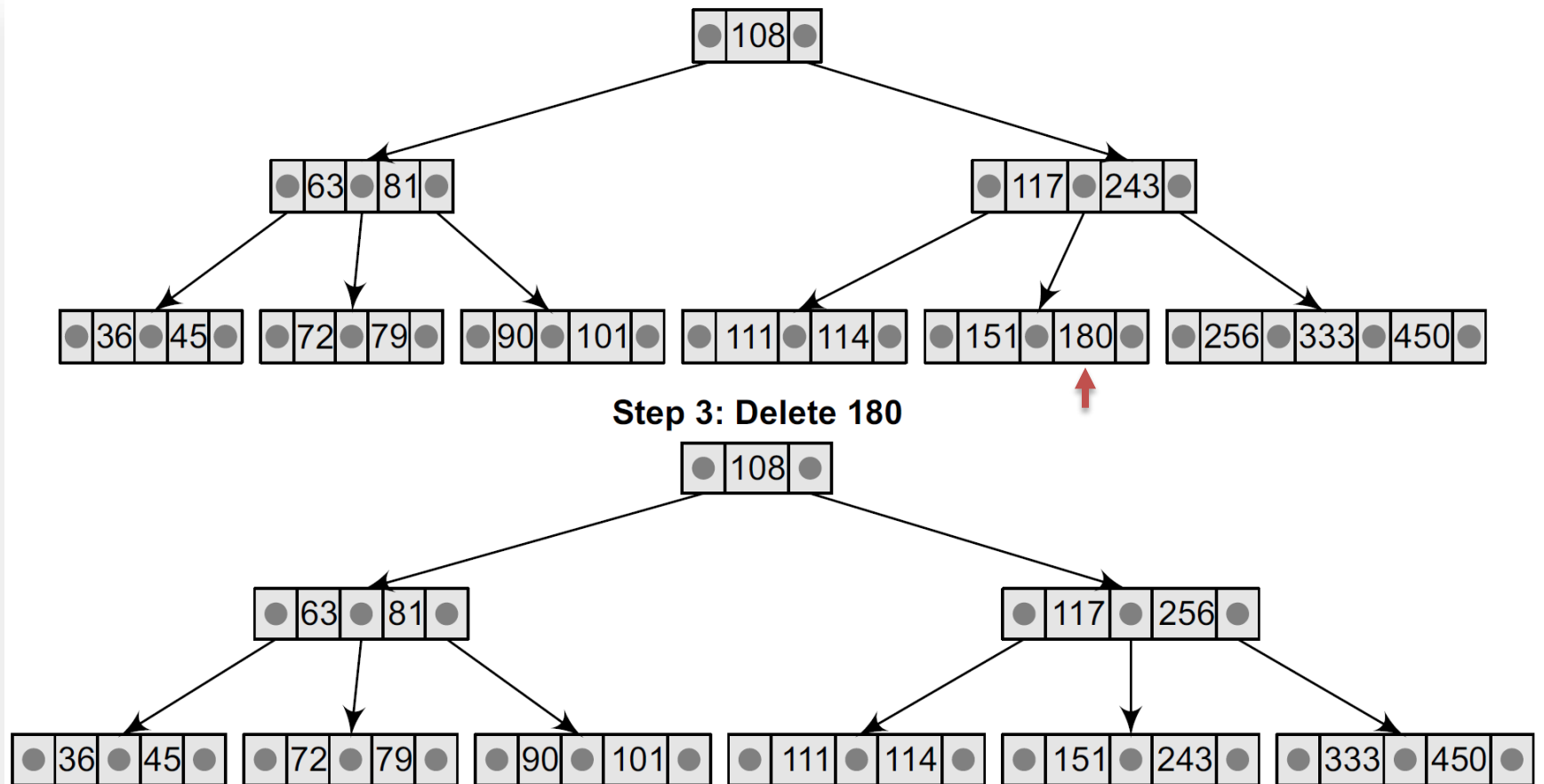


Step 2: Delete 201



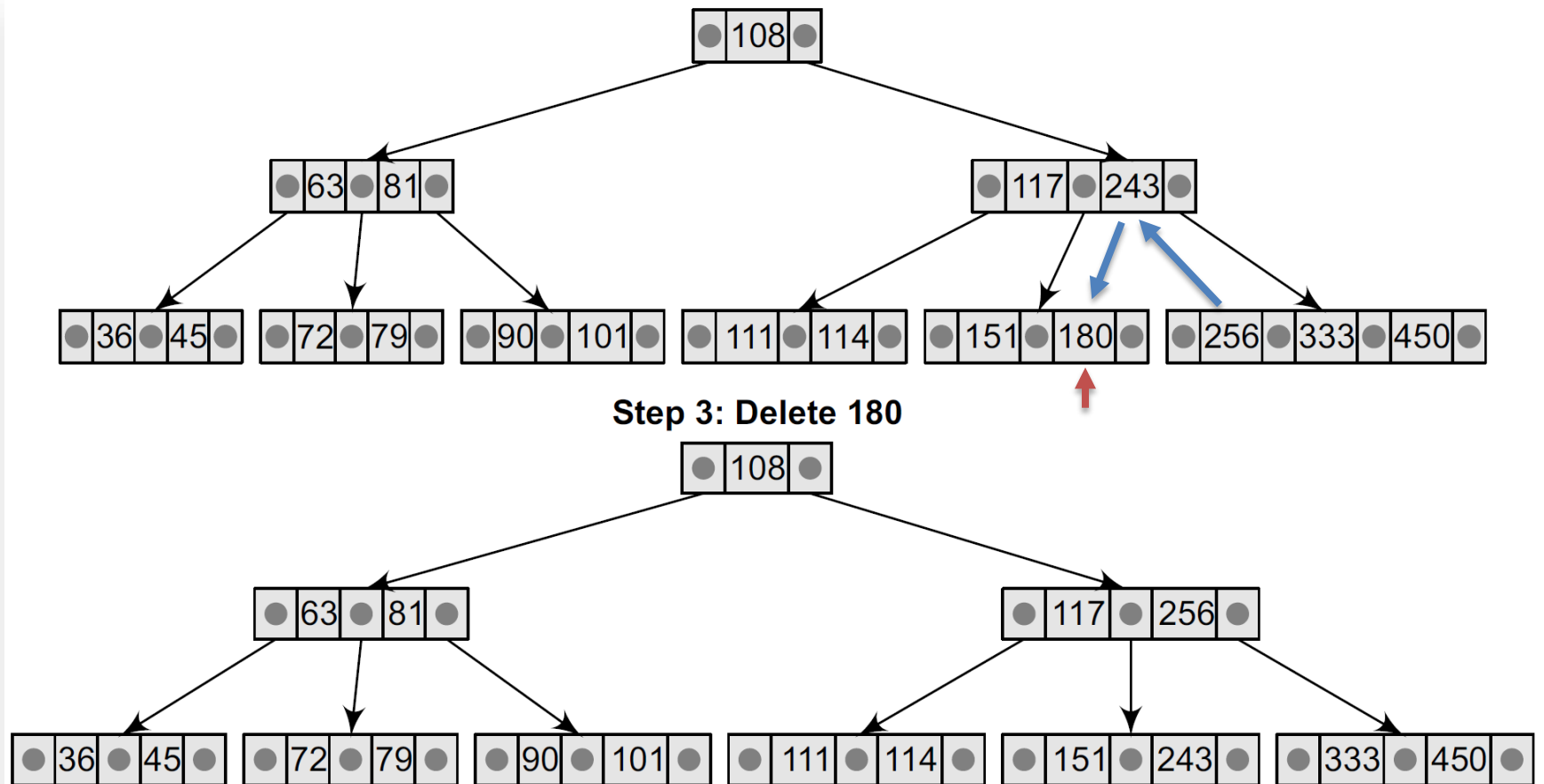
# Example – 1...

- Given a B tree of order 5, please delete values 93, 201, 180, and 72



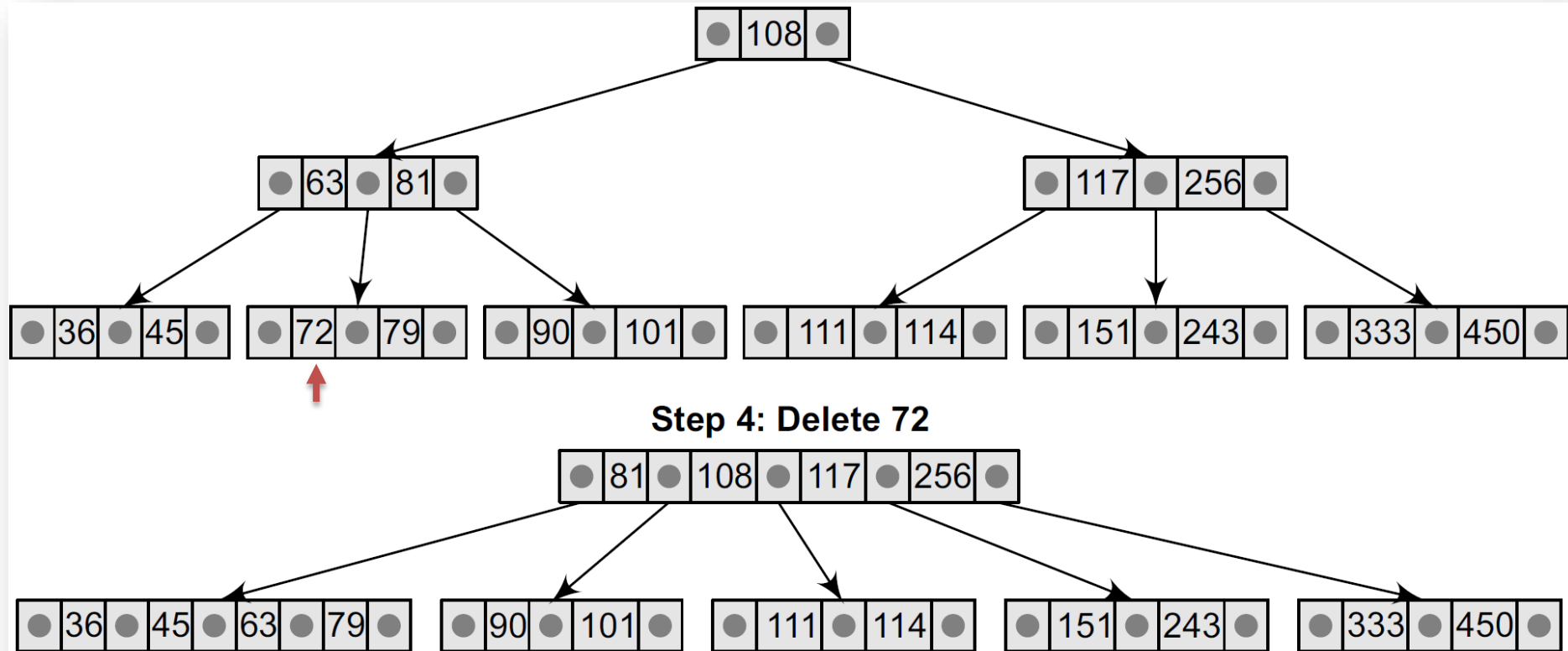
# Example – 1...

- Given a B tree of order 5, please delete values 93, 201, 180, and 72



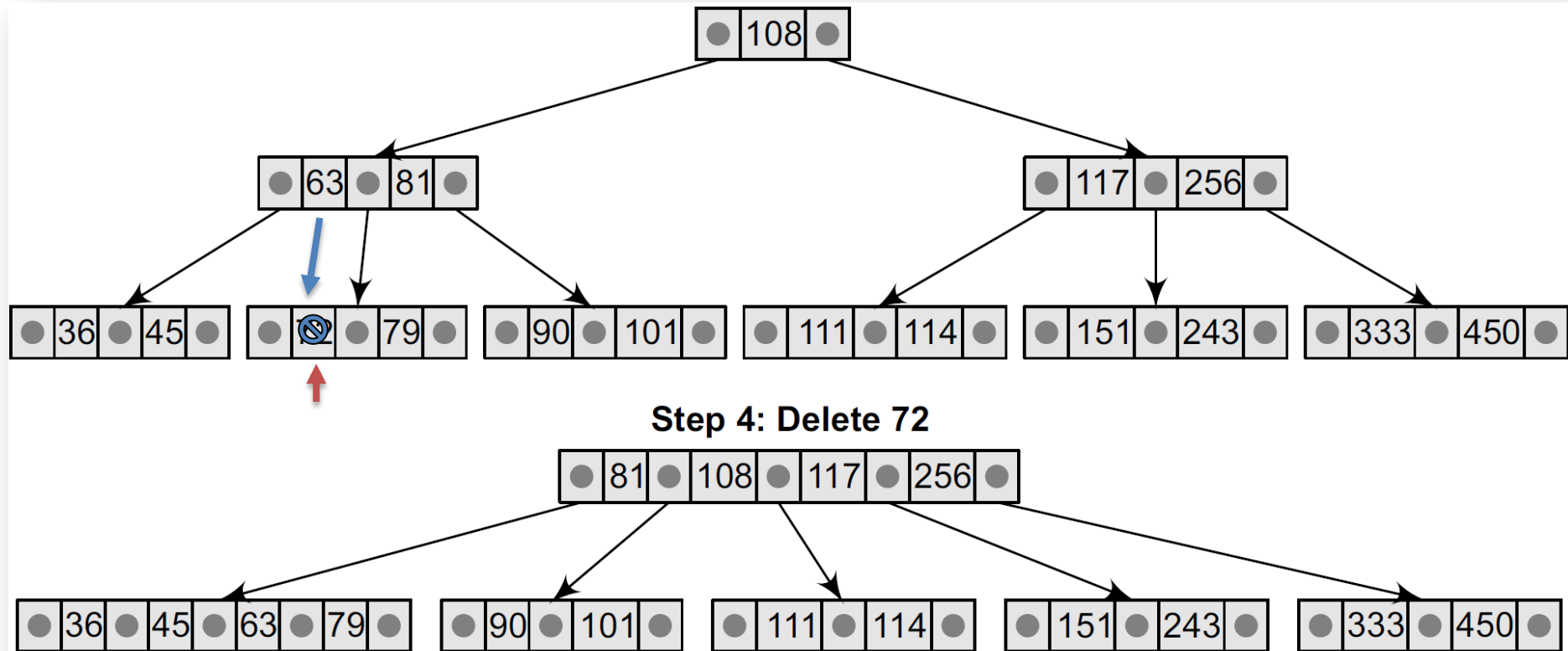
# Example – 1....

- Given a B tree of order 5, please delete values 93, 201, 180, and 72



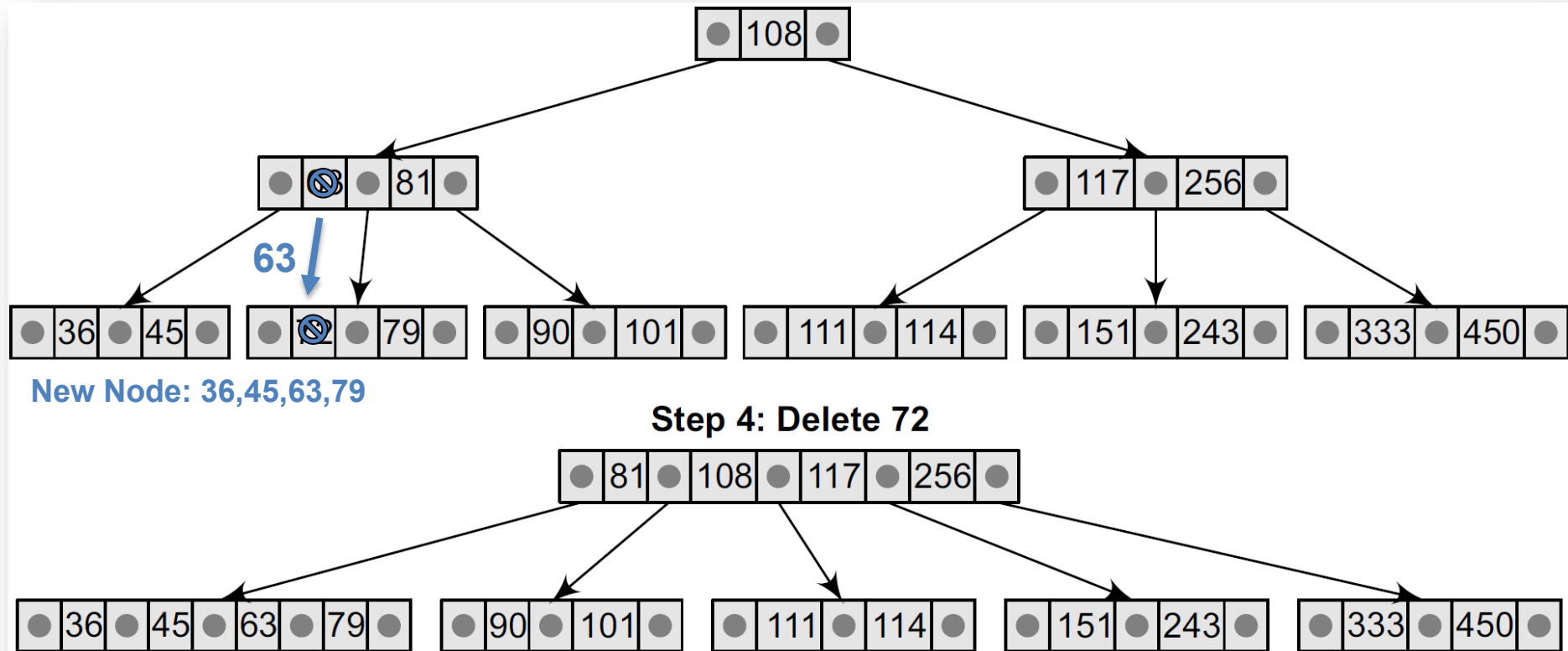
# Example – 1....

- Given a B tree of order 5, please delete values 93, 201, 180, and 72



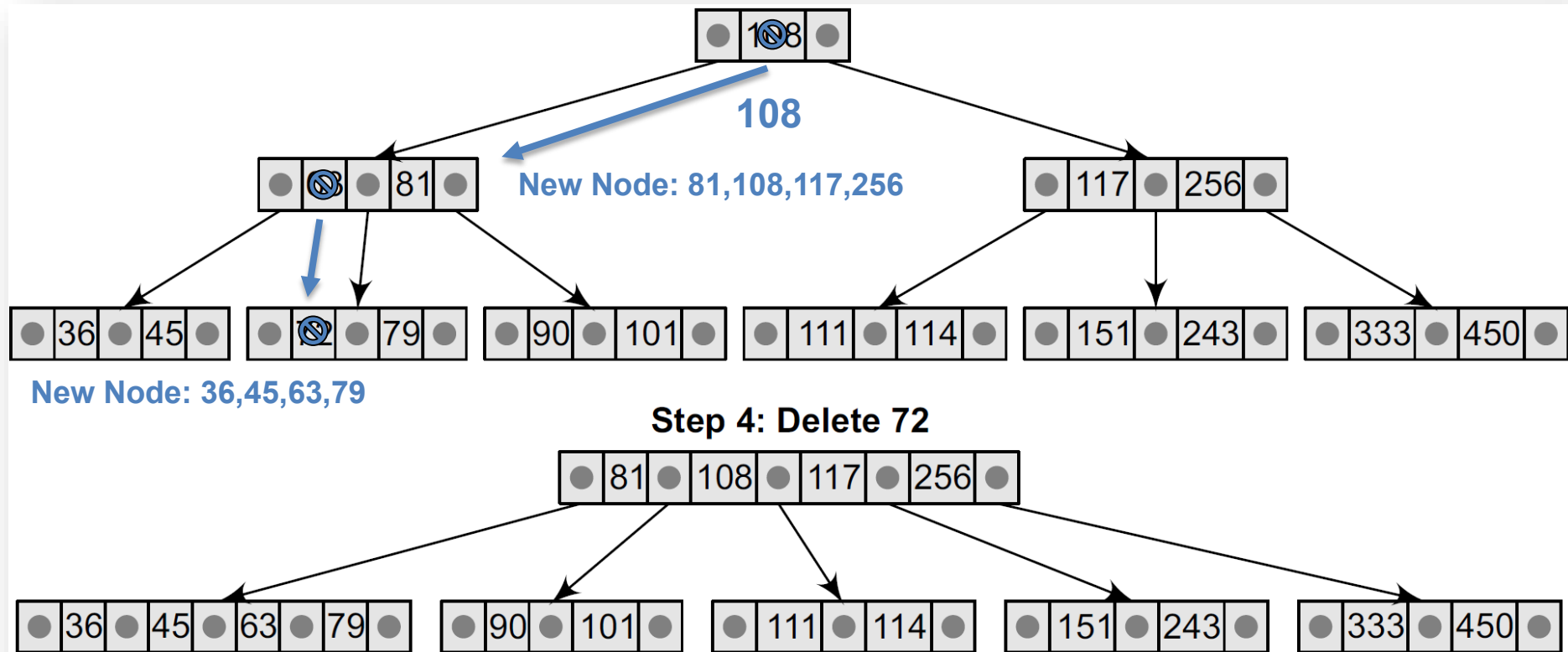
# Example – 1....

- Given a B tree of order 5, please delete values 93, 201, 180, and 72



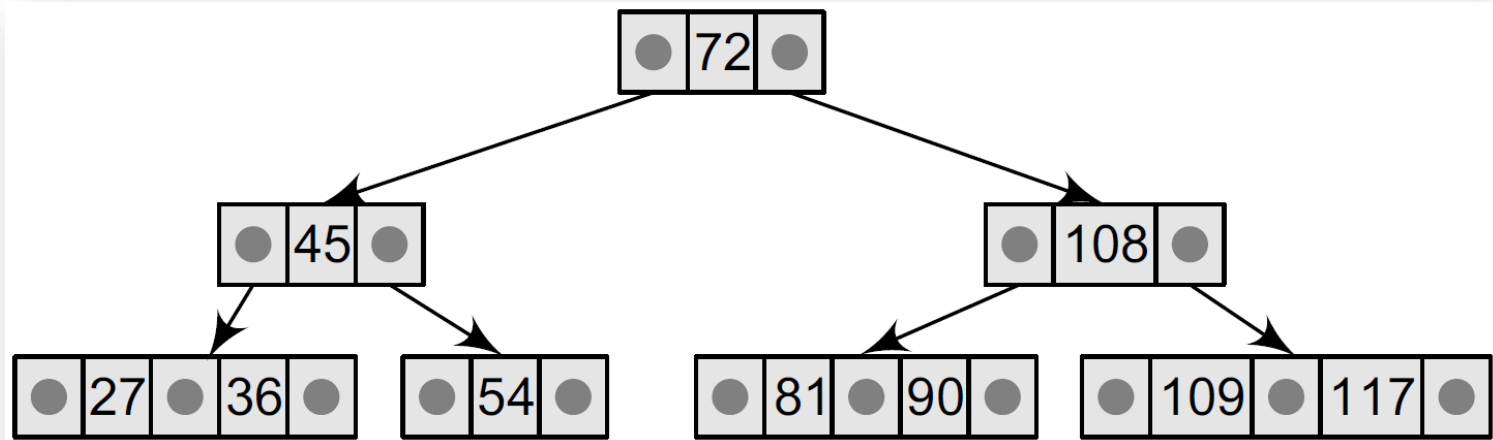
# Example – 1....

- Given a B tree of order 5, please delete values 93, 201, 180, and 72

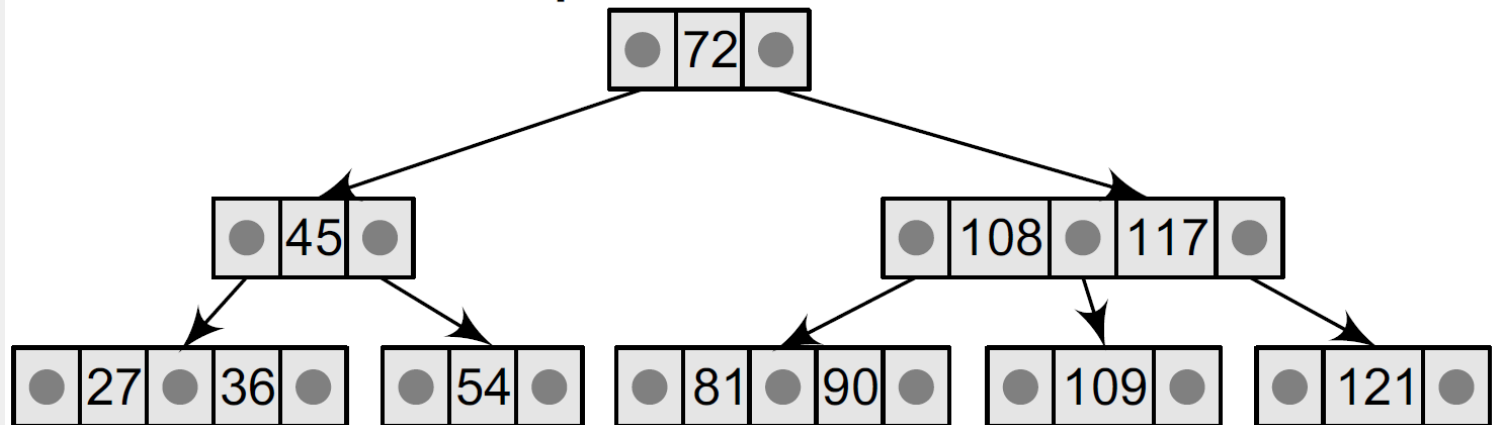


## Example – 2.

- Given a B tree of order 3, please insert 121, 87 and then delete 36, 109



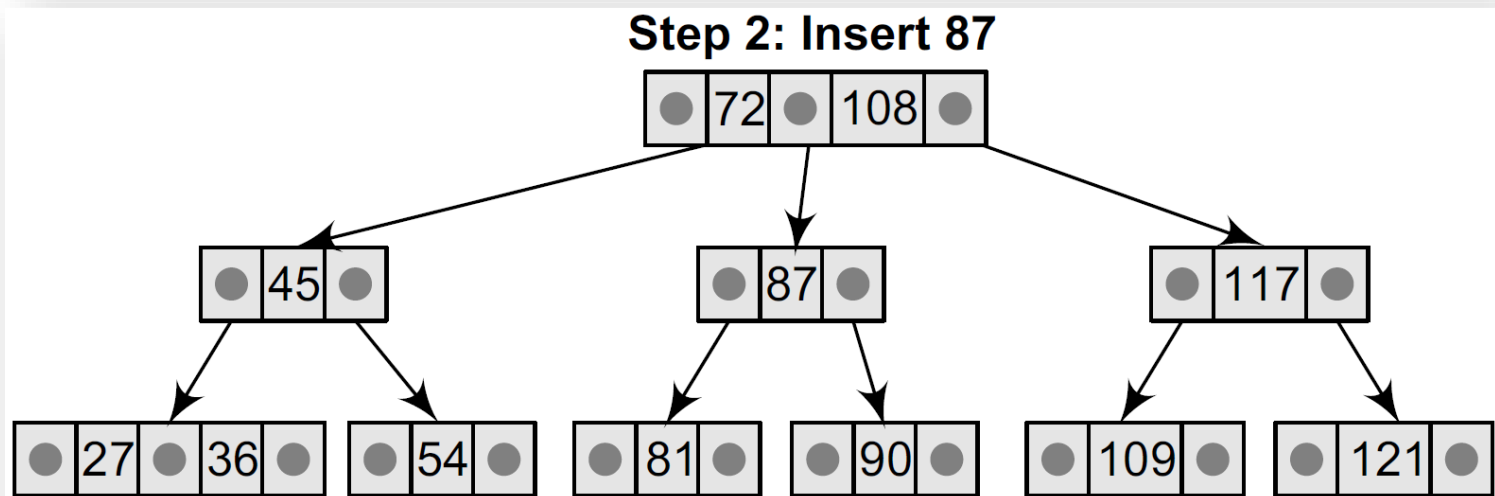
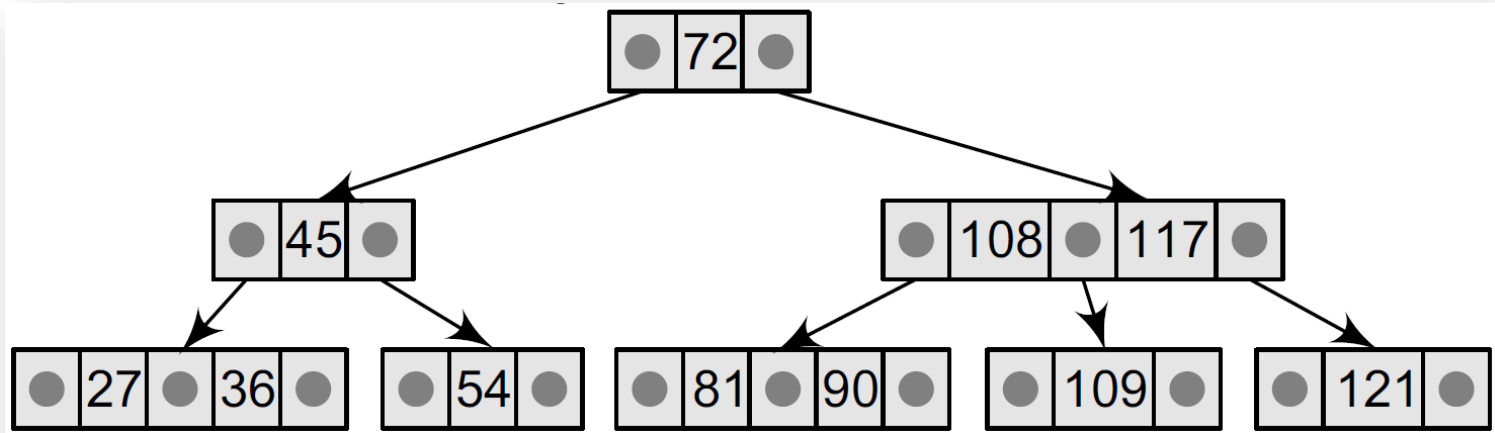
**Step 1: Insert 121**





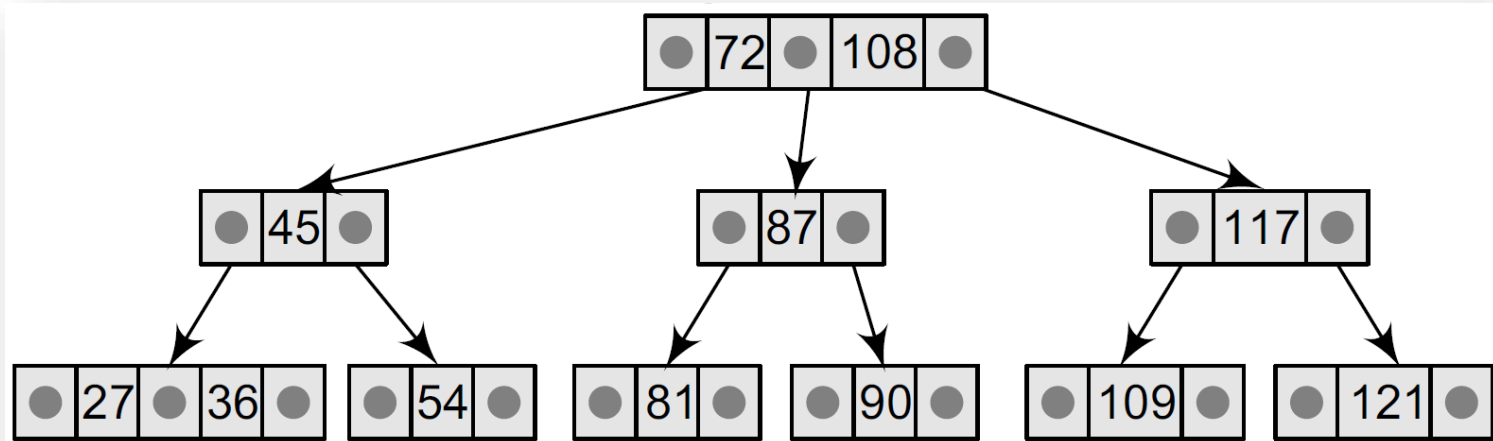
## Example – 2..

- Given a B tree of order 3, please insert 121, 87 and then delete 36, 109

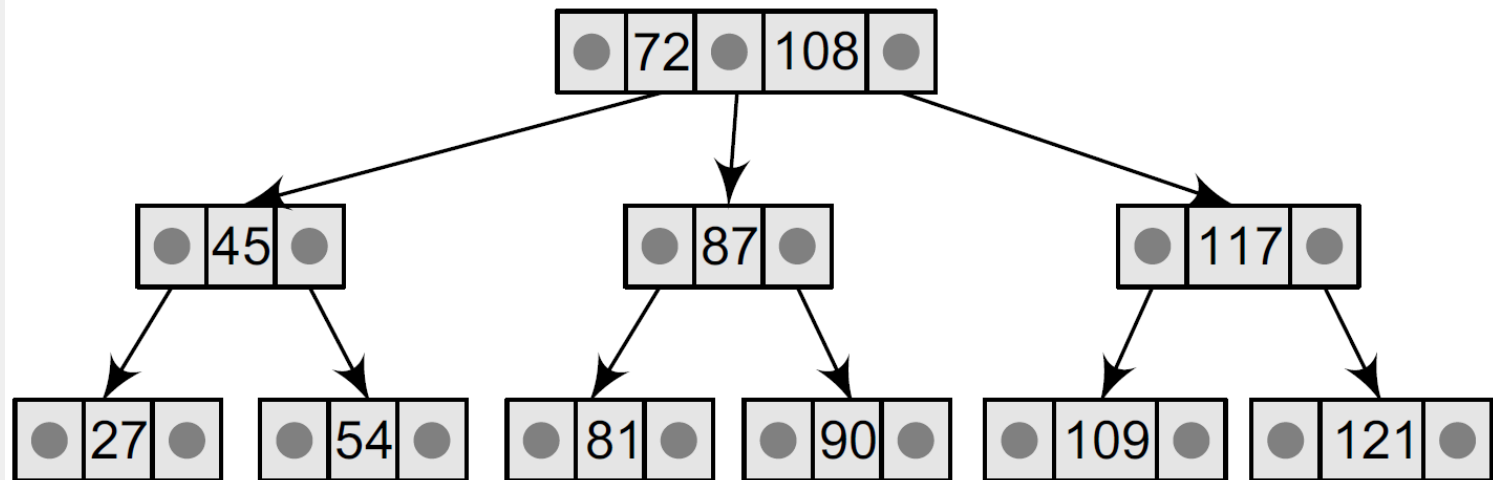


## Example – 2...

- Given a B tree of order 3, please insert 121, 87 and then delete 36, 109

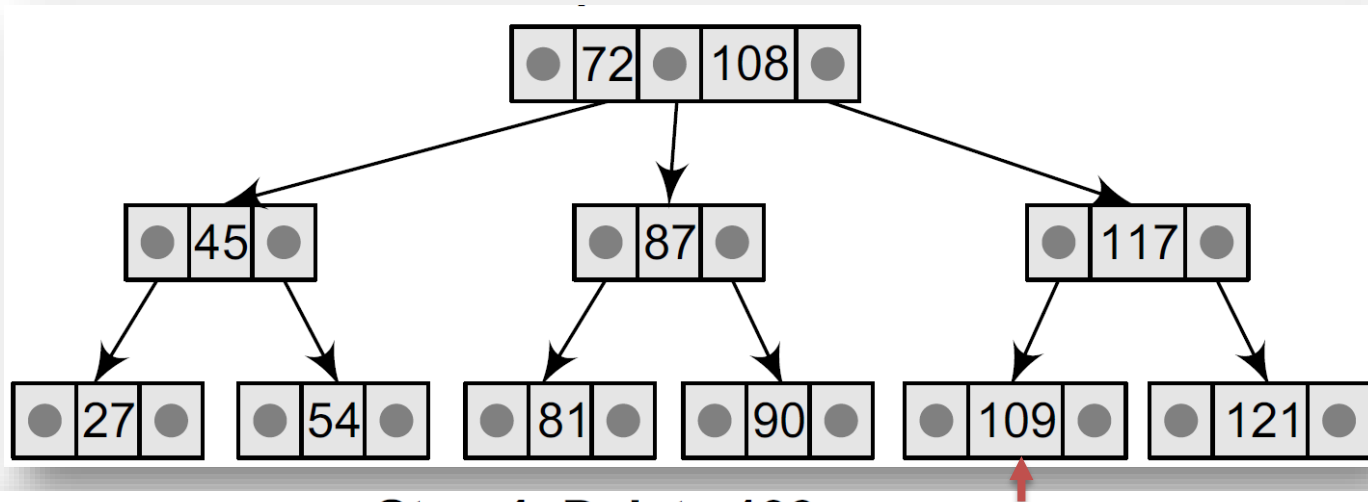


**Step 3: Delete 36**

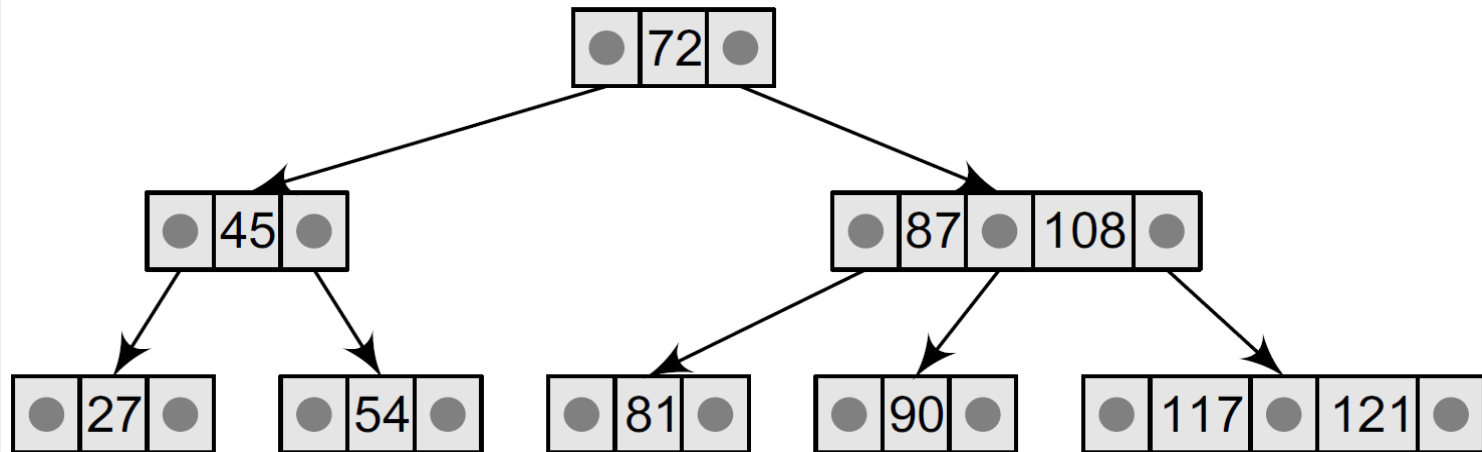


## Example – 2....

- Given a B tree of order 3, please insert 121, 87 and then delete 36, 109

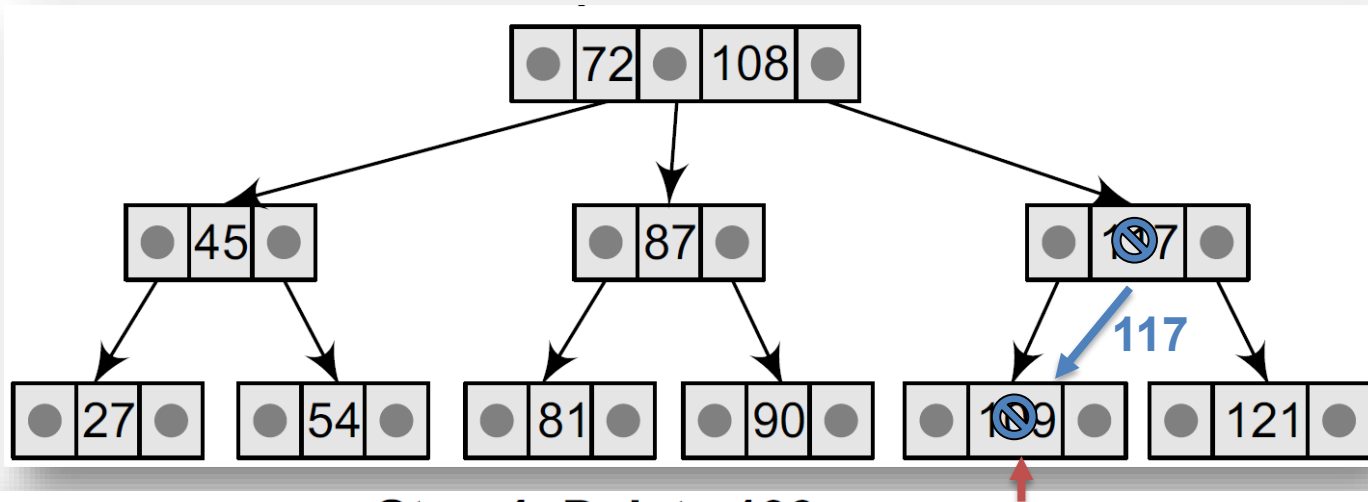


**Step 4: Delete 109**



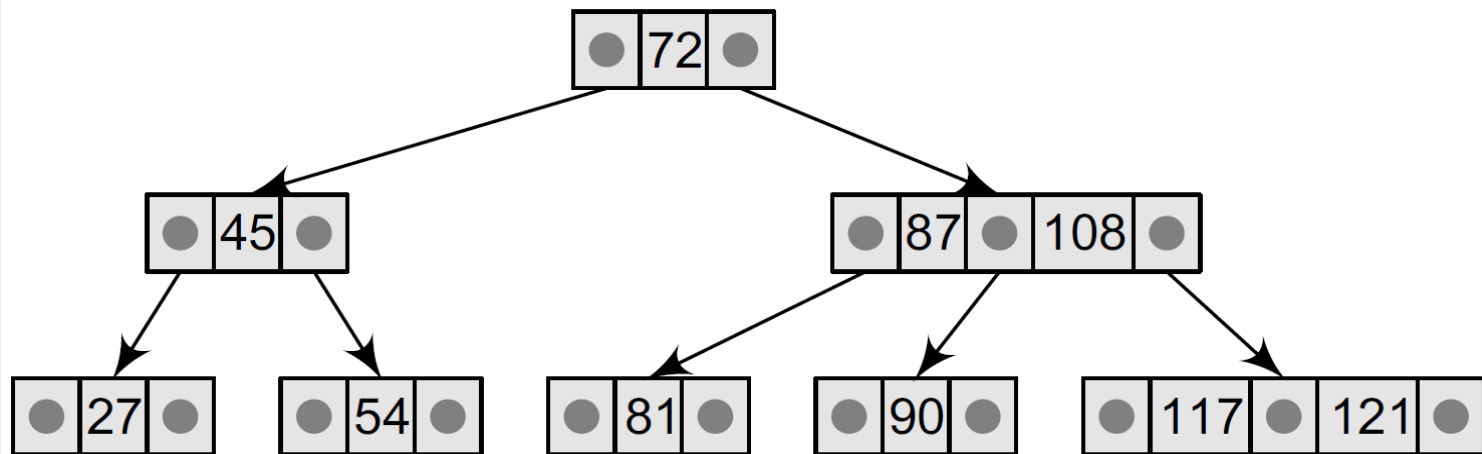
## Example – 2....

- Given a B tree of order 3, please insert 121, 87 and then delete 36, 109



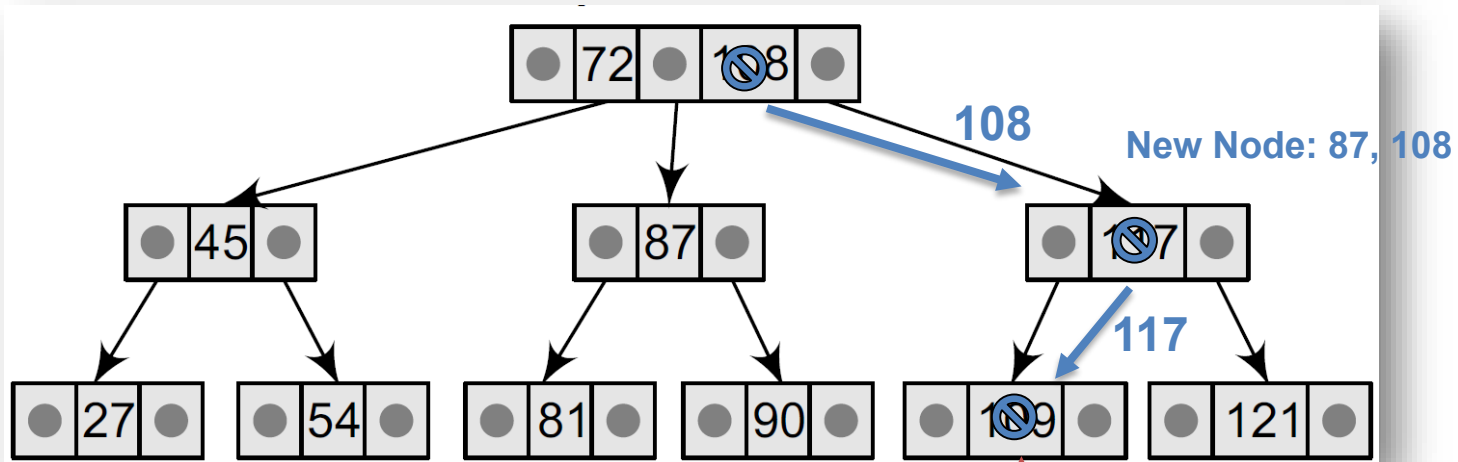
**Step 4: Delete 109**

New Node: 117, 121

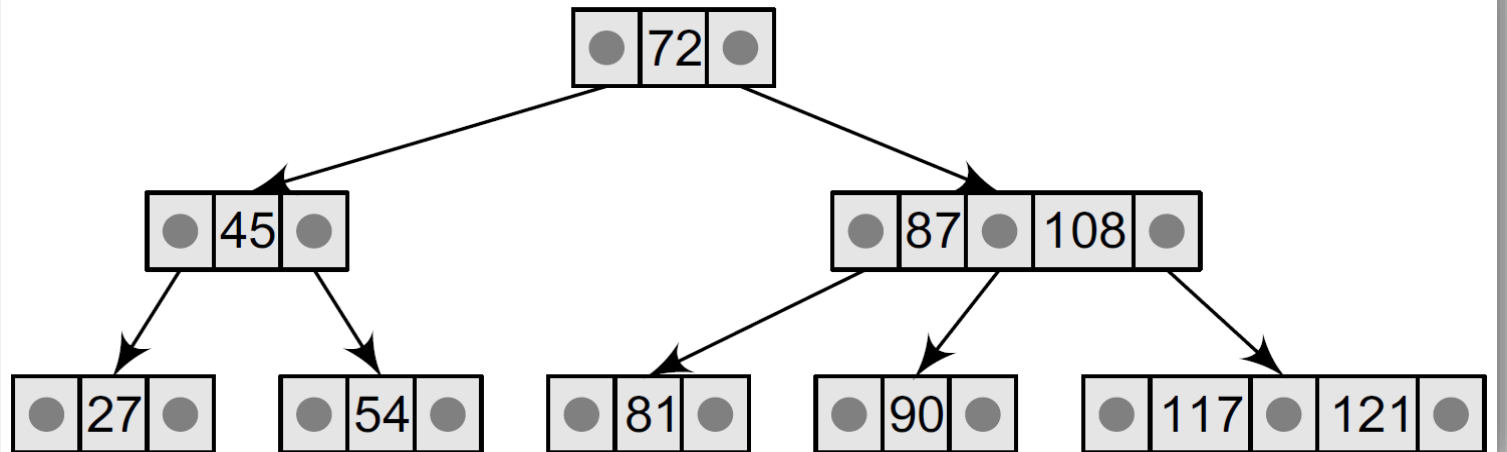


## Example – 2....

- Given a B tree of order 3, please insert 121, 87 and then delete 36, 109

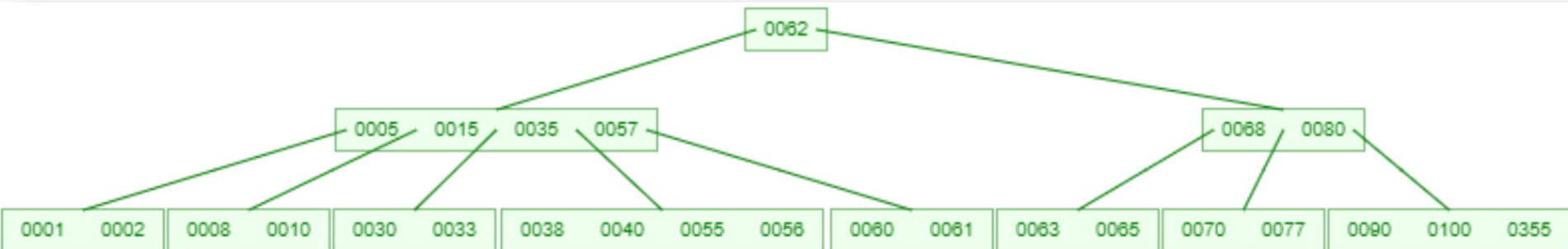
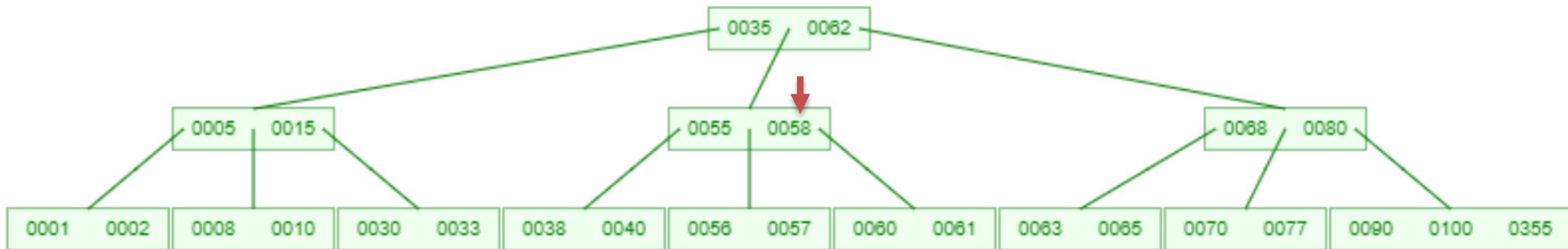


**Step 4: Delete 109**



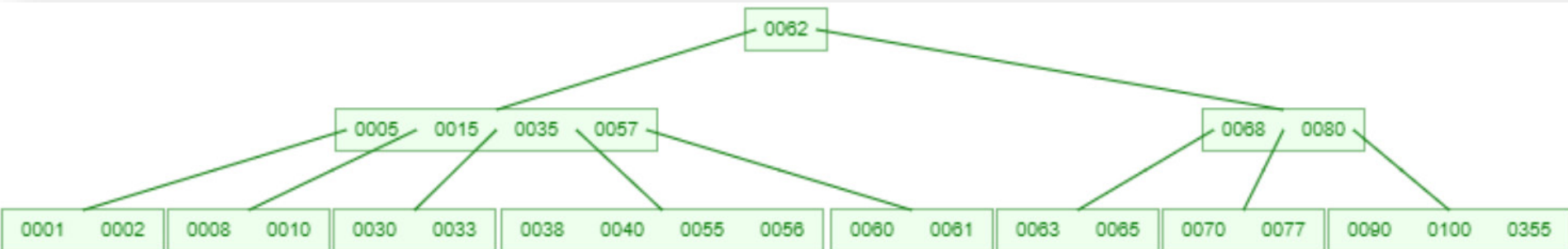
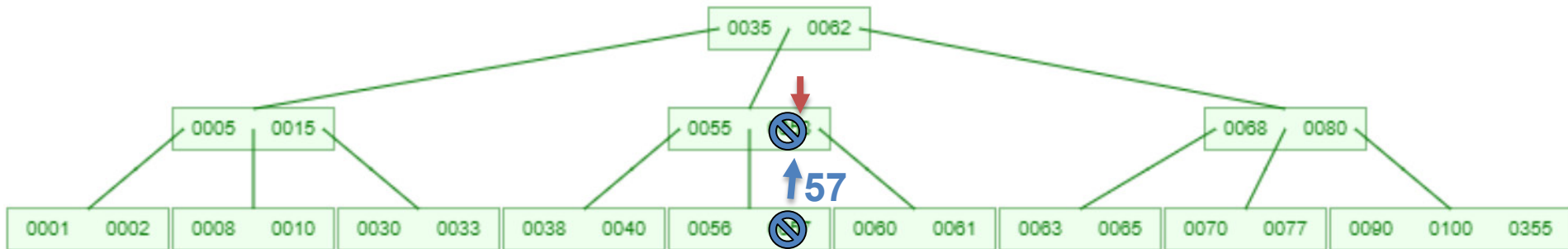
## Example – 3.

- Given a B tree of order 5, please delete 58, 65



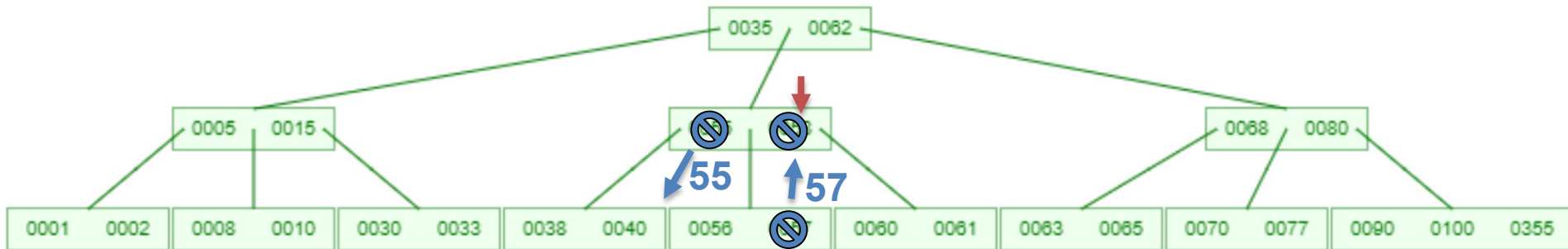
## Example – 3.

- Given a B tree of order 5, please delete 58, 65

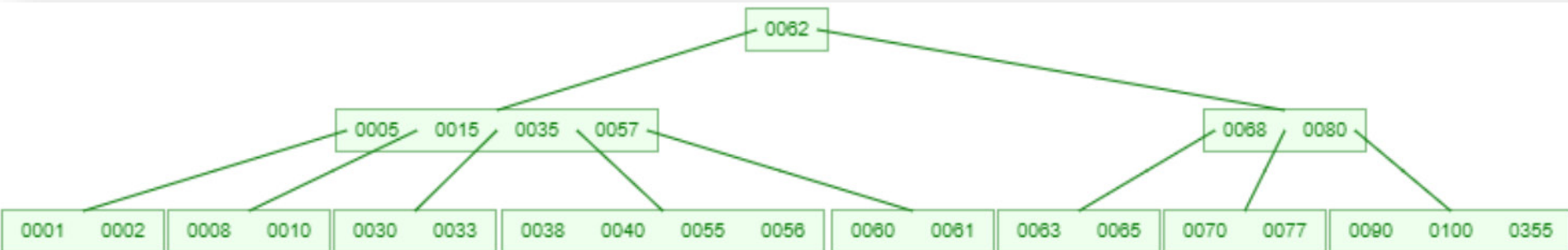


## Example – 3.

- Given a B tree of order 5, please delete 58, 65



New Node: 38, 40, 55, 56

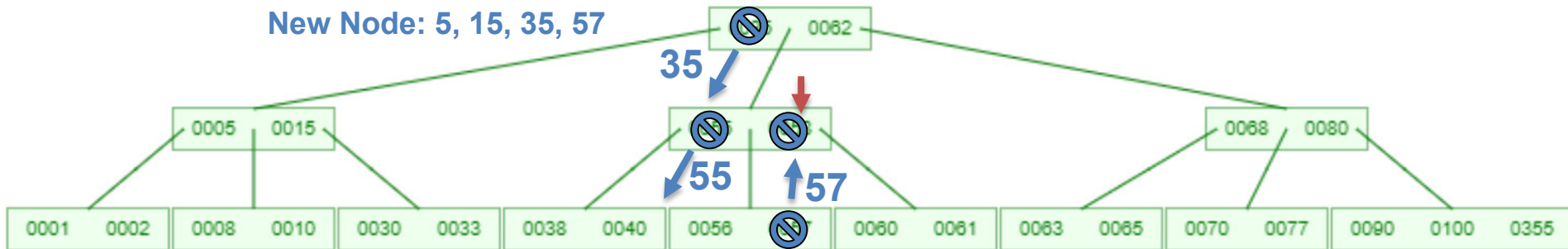




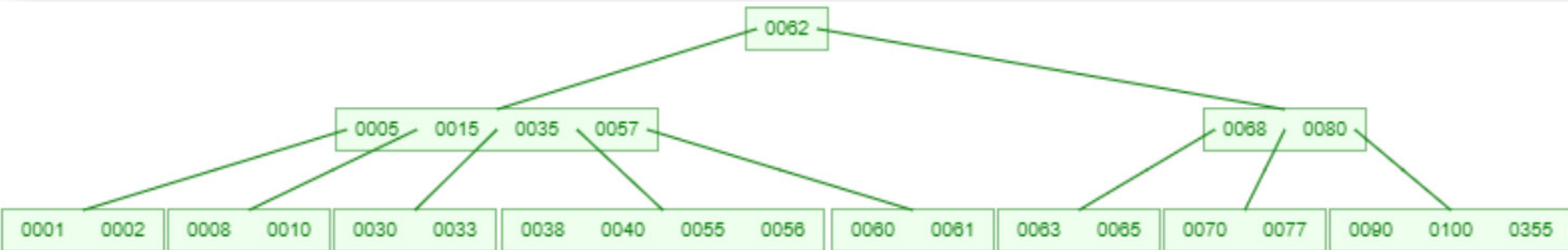
# Example – 3.

- Given a B tree of order 5, please delete 58, 65

New Node: 5, 15, 35, 57

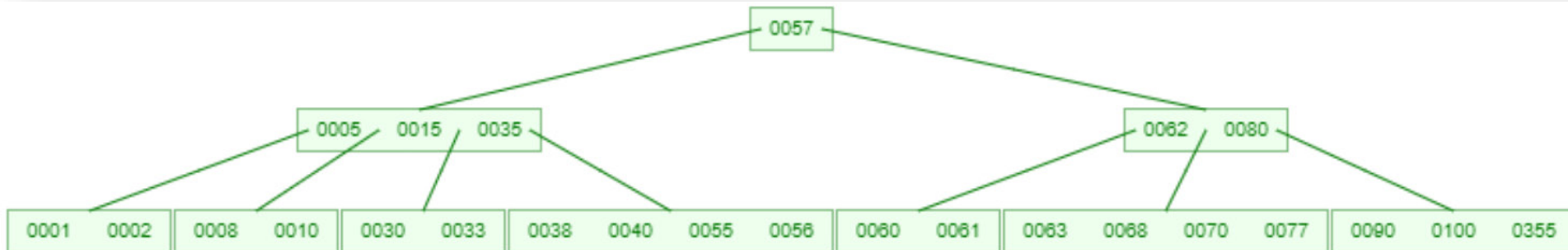
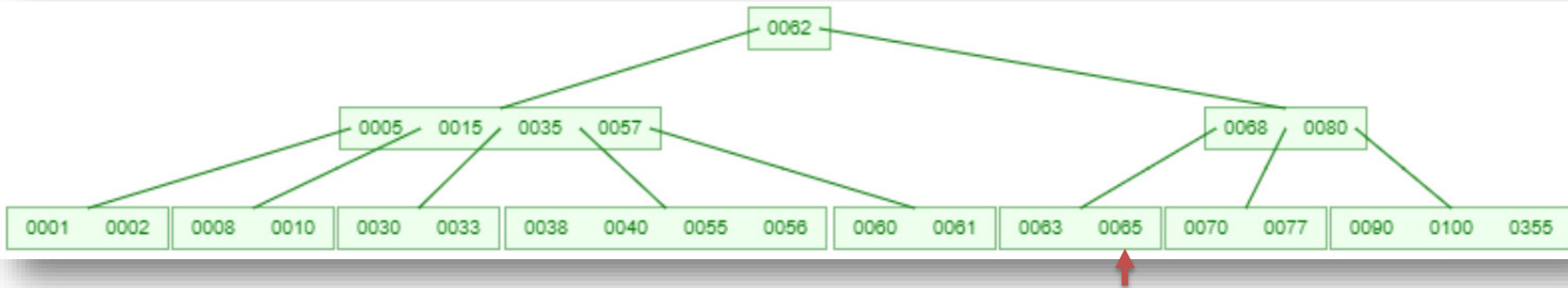


New Node: 38, 40, 55, 56



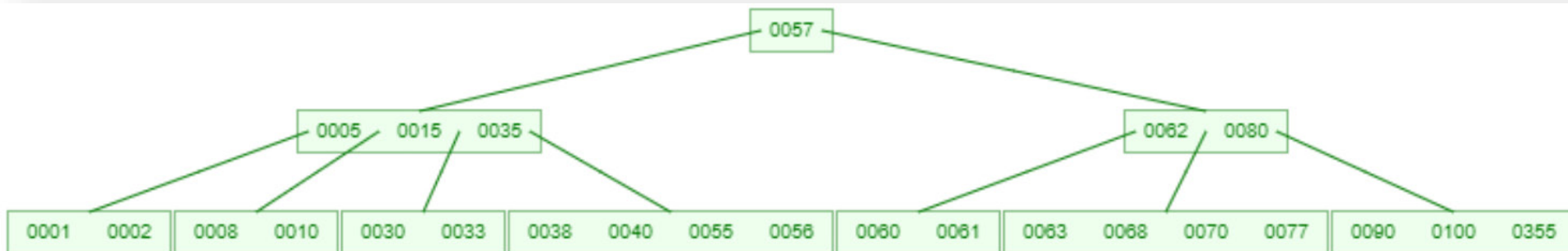
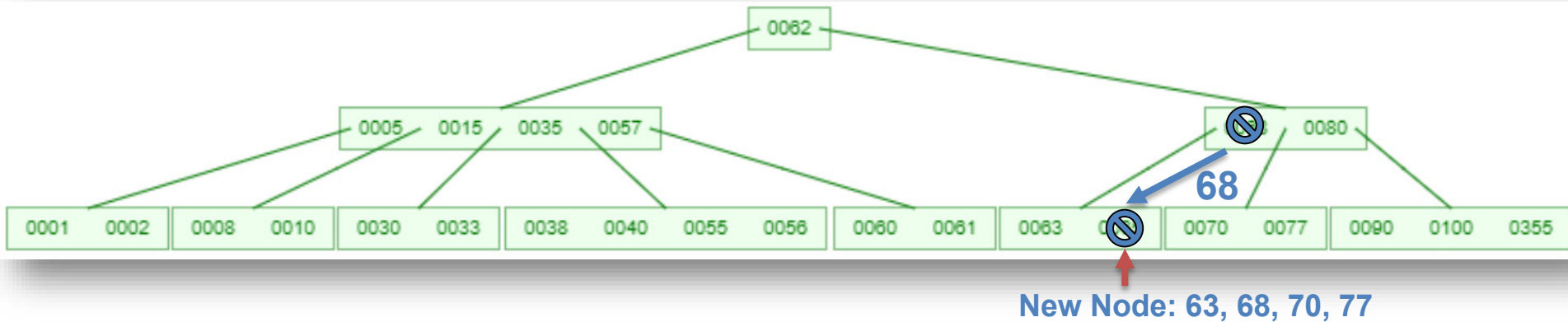
## Example – 3..

- Given a B tree of order 5, please delete 58, 65



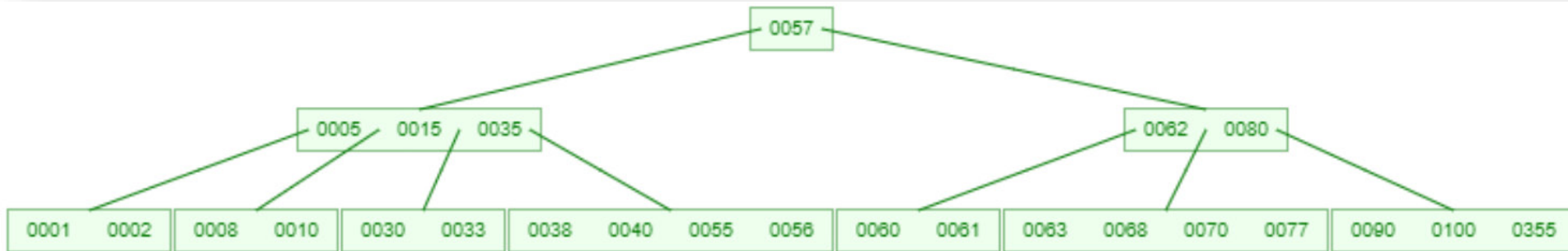
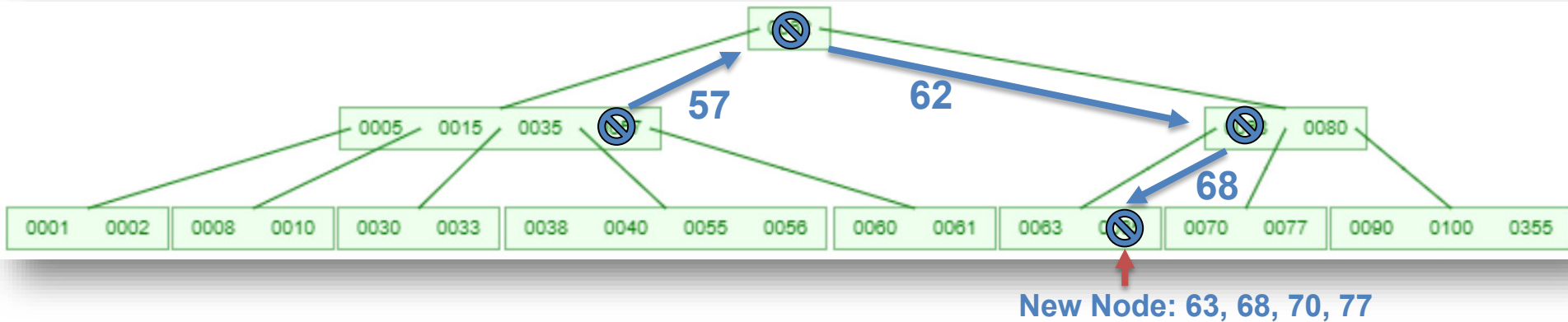
# Example – 3..

- Given a B tree of order 5, please delete 58, 65



# Example – 3..

- Given a B tree of order 5, please delete 58, 65



# Check the Demo!

---

- <https://www.cs.usfca.edu/~galles/visualization/BTree.html>

# Questions?

---



[kychen@mail.ntust.edu.tw](mailto:kychen@mail.ntust.edu.tw)