Tutorial 8: Indexing Techniques

CS3402 Database Systems

Question 1

- Construct a B^+-tree for the following set of key values:

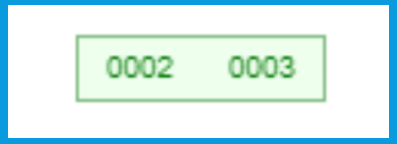
  (2, 3, 5, 7, 11, 17, 19, 23, 29, 31)

  Assuming that the tree is initially empty, values are added in ascending order, and the number of key values in internal nodes and leaf nodes are both 3 (i.e., the maximum number of tree pointers in an internal node is 4 = max. degree).

Question 1 (Answer) (1/5)

- Insert 2
  - The node has an empty space

- Insert 3
  - The node has an empty space

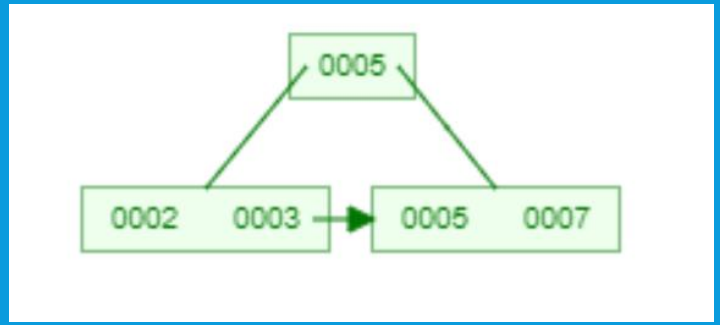


- Insert 5
  - The node has an empty space

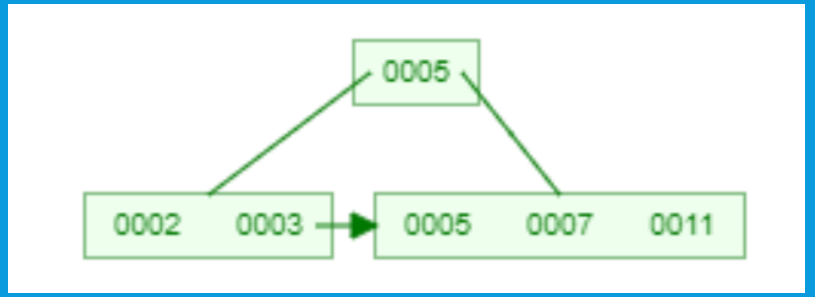
B⁺-tree visualization tool: https://www.cs.usfca.edu/~galles/visualization/BPlusTree.html

Question 1 (Answer) (2/5)

- Insert 7
  - Since the node is already full, split it into two nodes, distributing the keys evenly between the two nodes. Since the node is a leaf, take a copy of the minimum value in the second of these two nodes (i.e., 5) and repeat the insertion algorithm to insert it into the parent node.

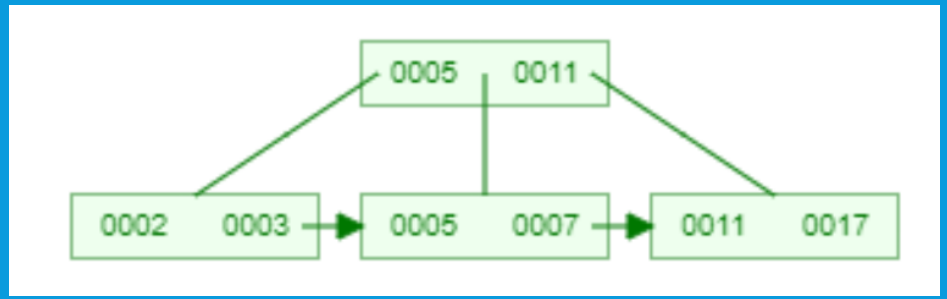


- Insert 11
  - The node has an empty space

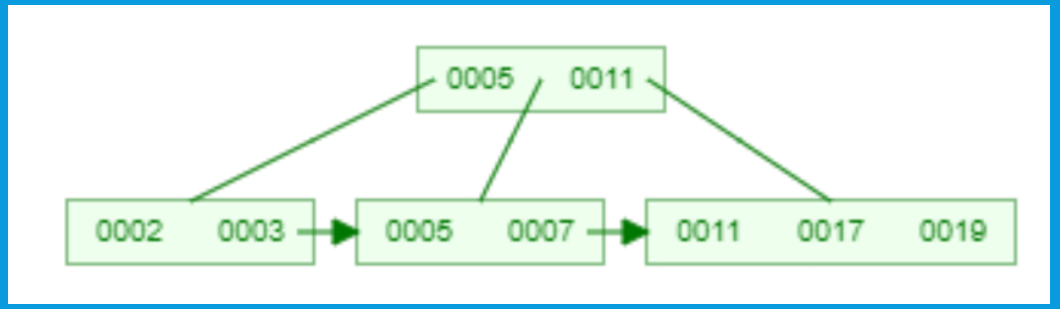


Question 1 (Answer) (3/5)

- Insert 17
  - Since the node is already full, split it into two nodes, distributing the keys evenly between the two nodes. Since the node is a leaf, take a copy of the minimum value in the second of these two nodes (i.e., 11) and repeat the insertion algorithm to insert it into the parent node.

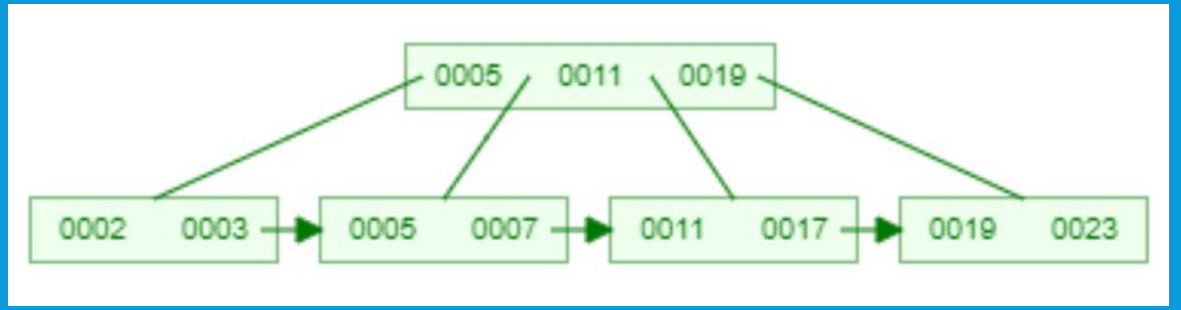


- Insert 19
  - The node has an empty space

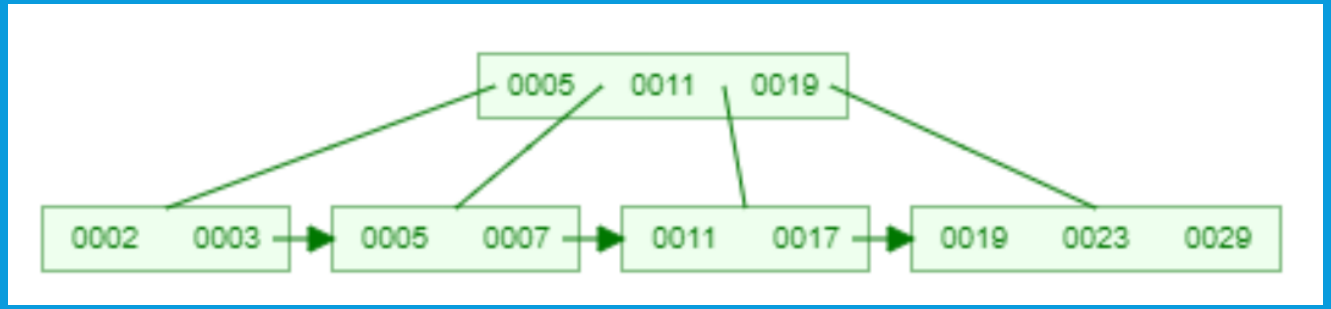


Question 1 (Answer) (4/5)

- Insert 23
  - Since the node is already full, split it into two nodes, distributing the keys evenly between the two nodes. Since the node is a leaf, take a copy of the minimum value in the second of these two nodes (i.e., 19) and repeat the insertion algorithm to insert it into the parent node.

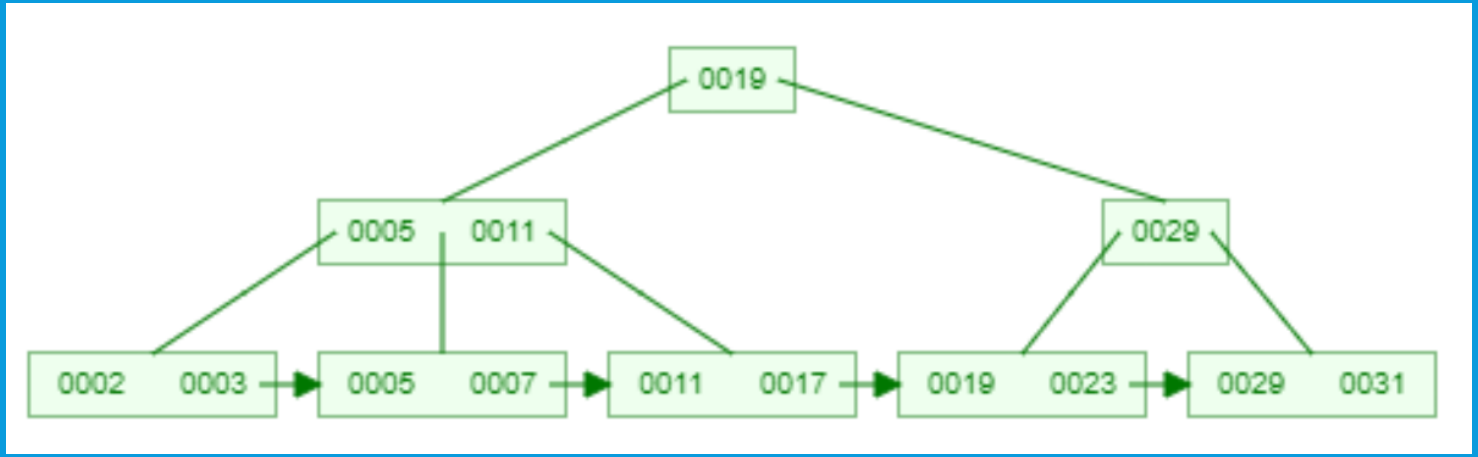


- Insert 29
  - The node has an empty space



Question 1 (Answer) (5/5)

- Insert 31
  - Since the node is already full, split it into two nodes, distributing the keys evenly between the two nodes. Since the node is a leaf, take a copy of the minimum value in the second of these two nodes (i.e., 29) and repeat the insertion algorithm to insert it into the parent node. Since the parent node is a non-leaf, exclude the middle value (i.e., 19) during the split and repeat this insertion algorithm to insert this excluded value into the parent node (i.e., the root).

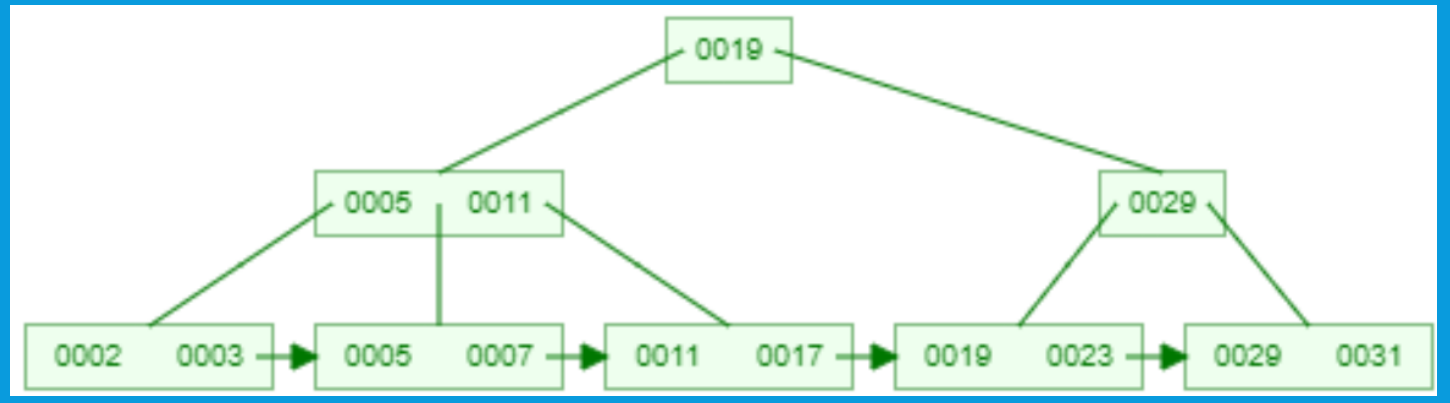


Question 2

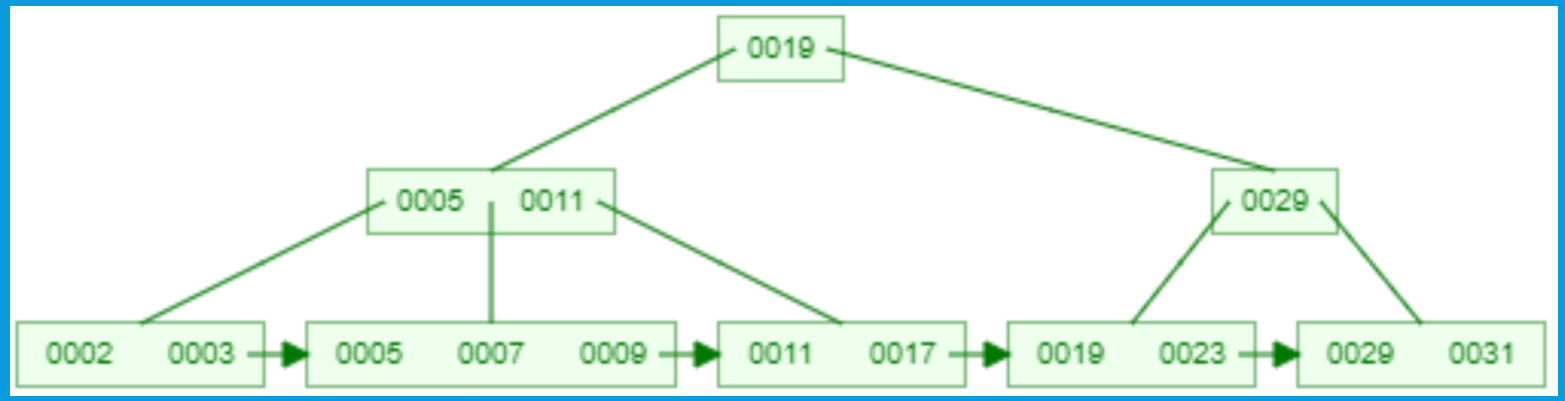
- For the B^+-tree constructed for Question 1, show the form of the tree after each of the following series of operations:
  - Insert 9
  - Insert 10
  - Insert 8
  - Delete 7
  - Delete 8
  - Delete 5
  - Delete 3
  - Delete 11

Question 2 (Answer) (1/8)

- The B+ tree constructed for Question 1

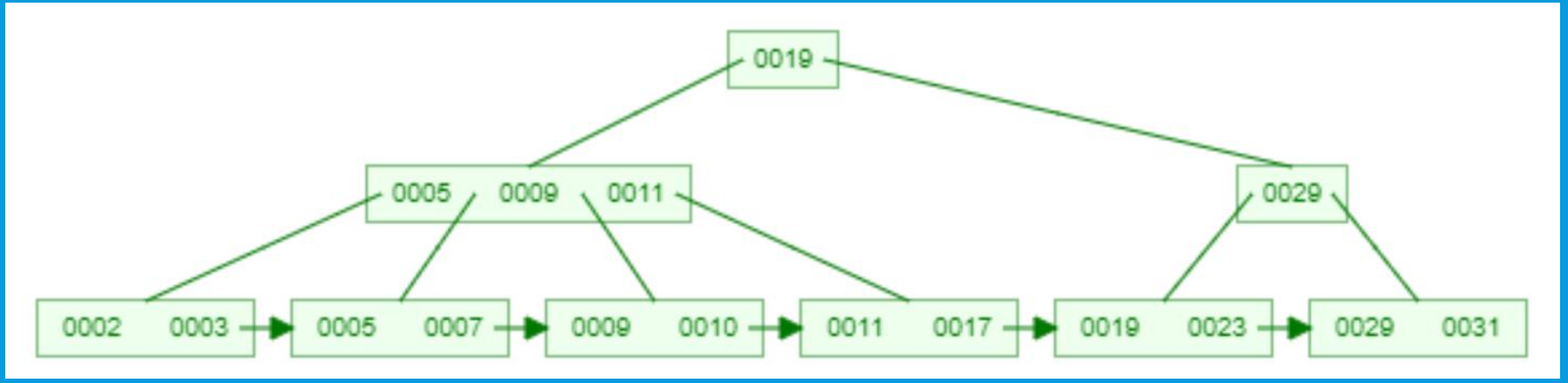


- Insert 9
  - The node has an empty space



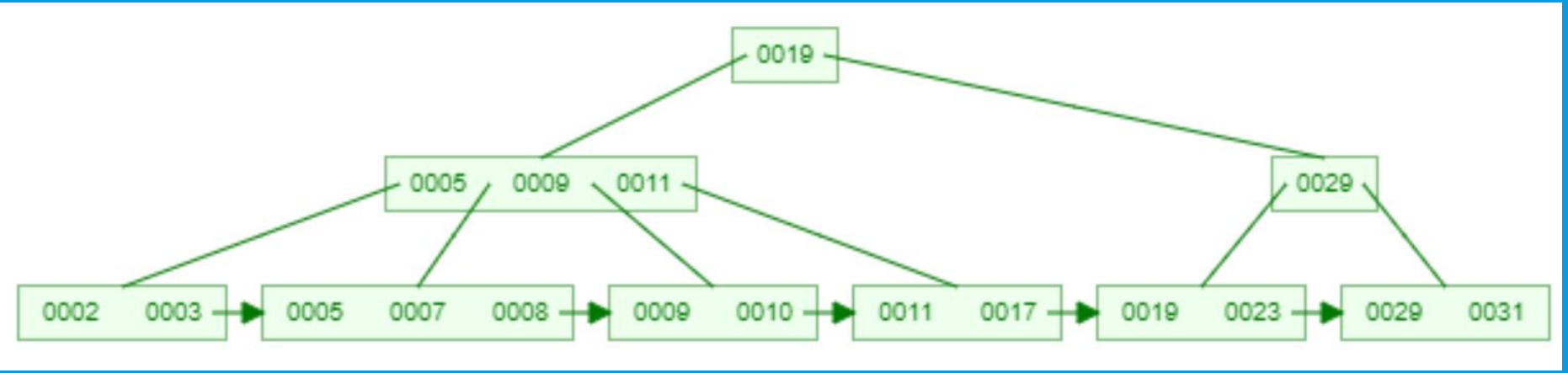
Question 2 (Answer) (2/8)

- Insert 10
  - Since the node is already full, split it into two nodes, distributing the keys evenly between the two nodes. Since the node is a leaf, take a copy of the minimum value in the second of these two nodes (i.e., 9) and repeat the insertion algorithm to insert it into the parent node.



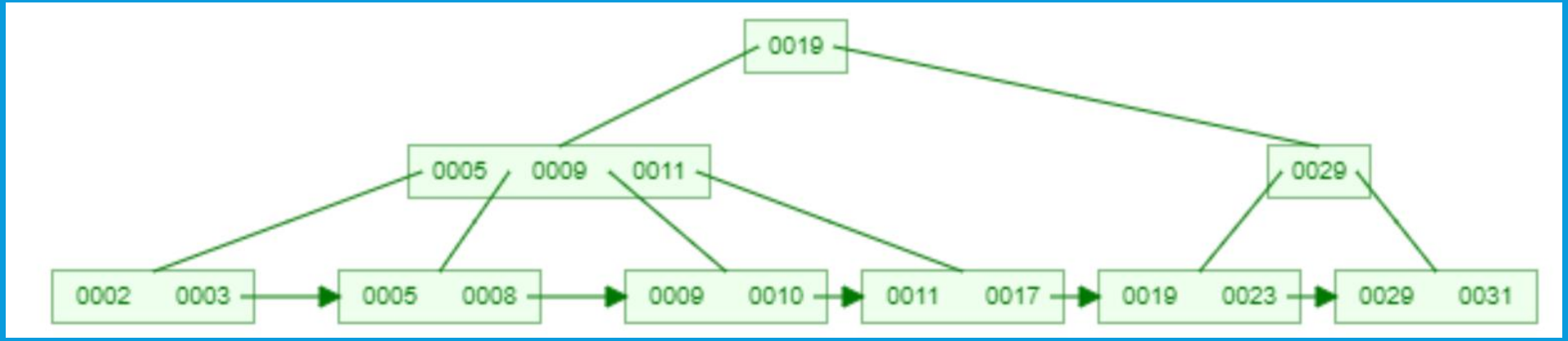
Question 2 (Answer) (3/8)

- Insert 8
  - The node has an empty space



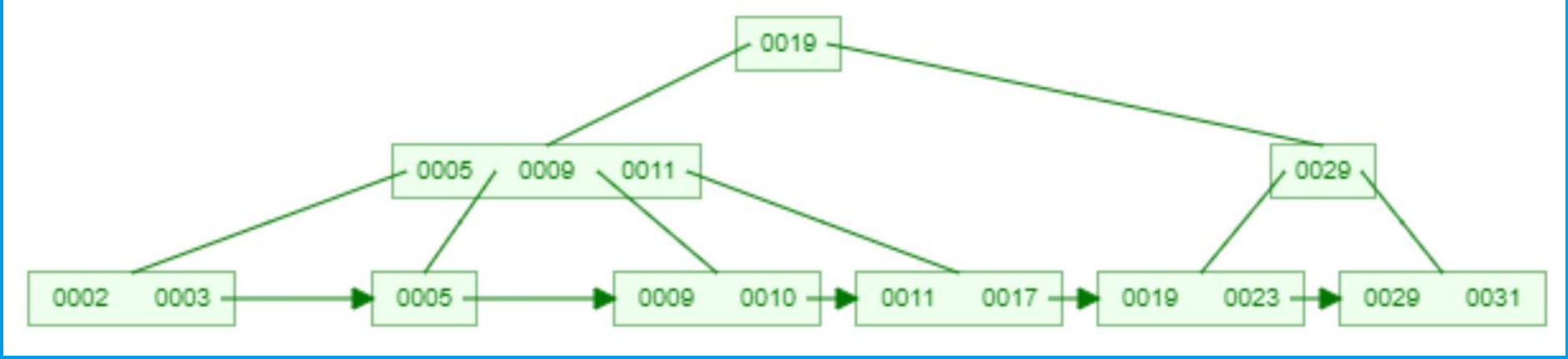
Question 2 (Answer) (4/8)

- Delete 7
  - The node still has half-full keys



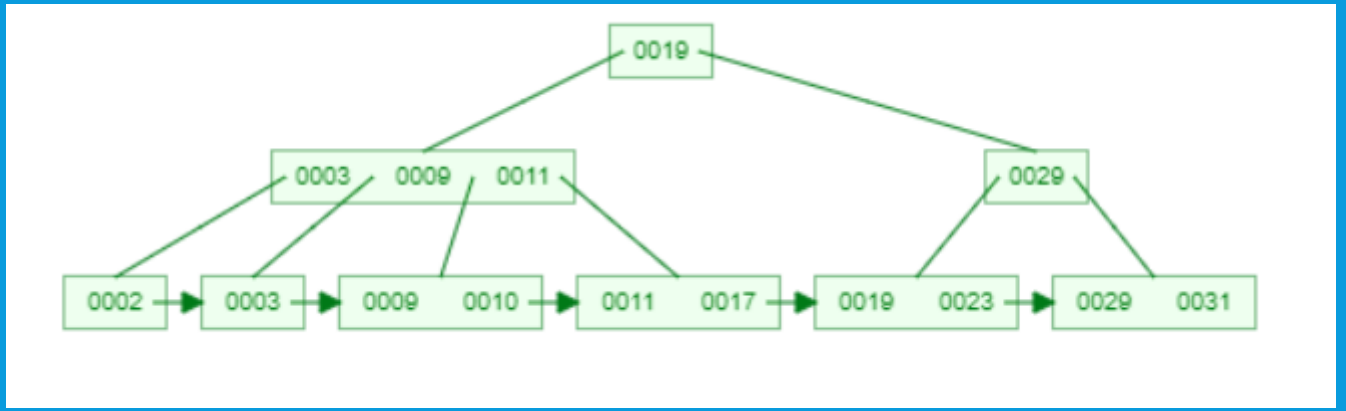
Question 2 (Answer) (5/8)

- Delete 8
  - The node still has half-full keys



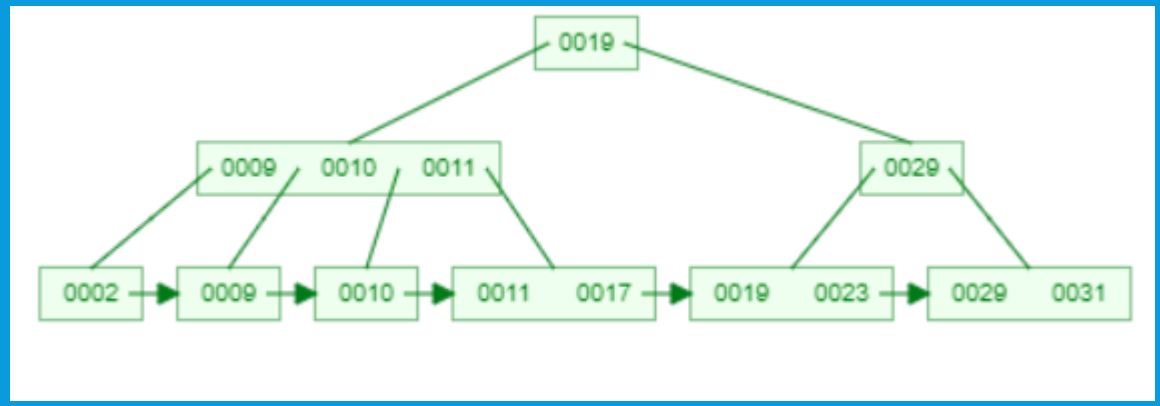
Question 2 (Answer) (6/8)

- Delete 5
  - Distribute the keys between the node and the neighbor. Repair the keys in the levels above.



Question 2 (Answer) (7/8)

- Delete 3
  - Distribute the keys between the node and the neighbor. Repair the keys in the levels above.



Question 2 (Answer) (8/8)

- Delete 11
  - Distribute the keys between the node and the neighbor. Repair the keys in the levels above.