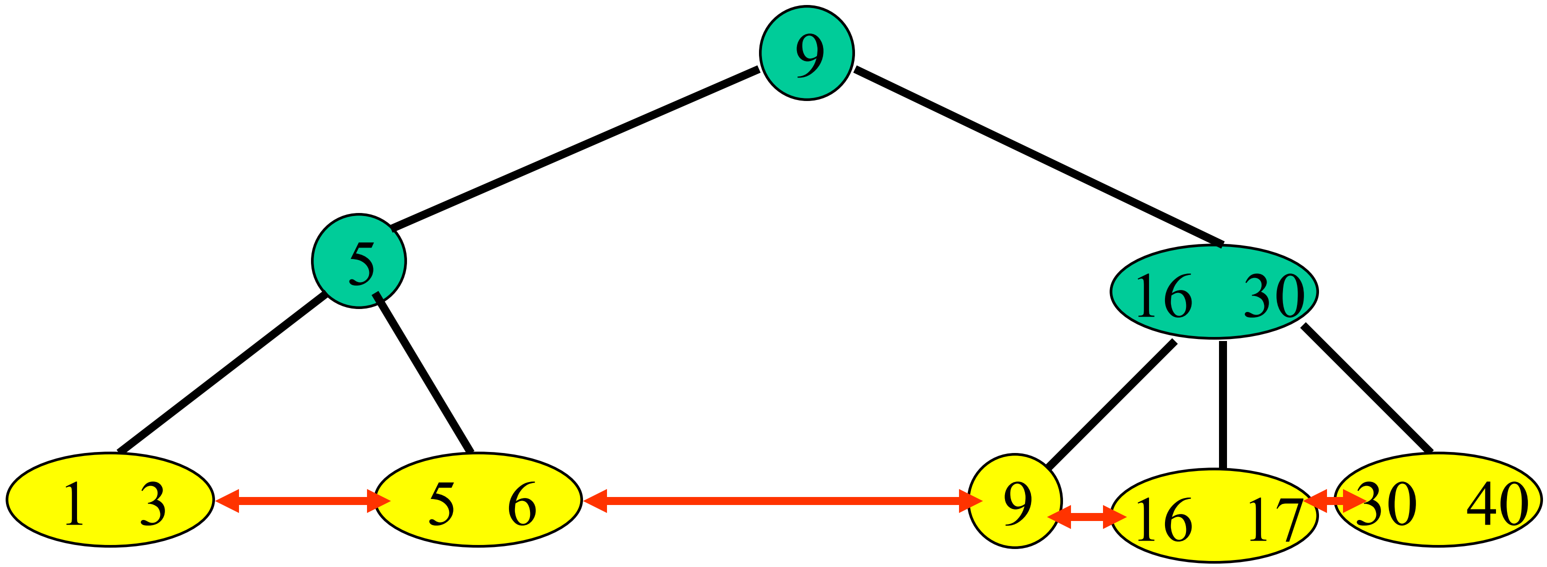
B^+-Trees

- Same structure as B-trees.
- Dictionary pairs are in leaves only. Leaves form a doubly-linked list.
- Remaining nodes have following structure:

$$j\ a_0\ k_1\ a_1\ k_2\ a_2\ \ldots\ k_j\ a_j$$

- j = number of keys in node.
- a_i is a pointer to a subtree.
- k_i <= smallest key in subtree a_i and > largest in a_{i-1}.

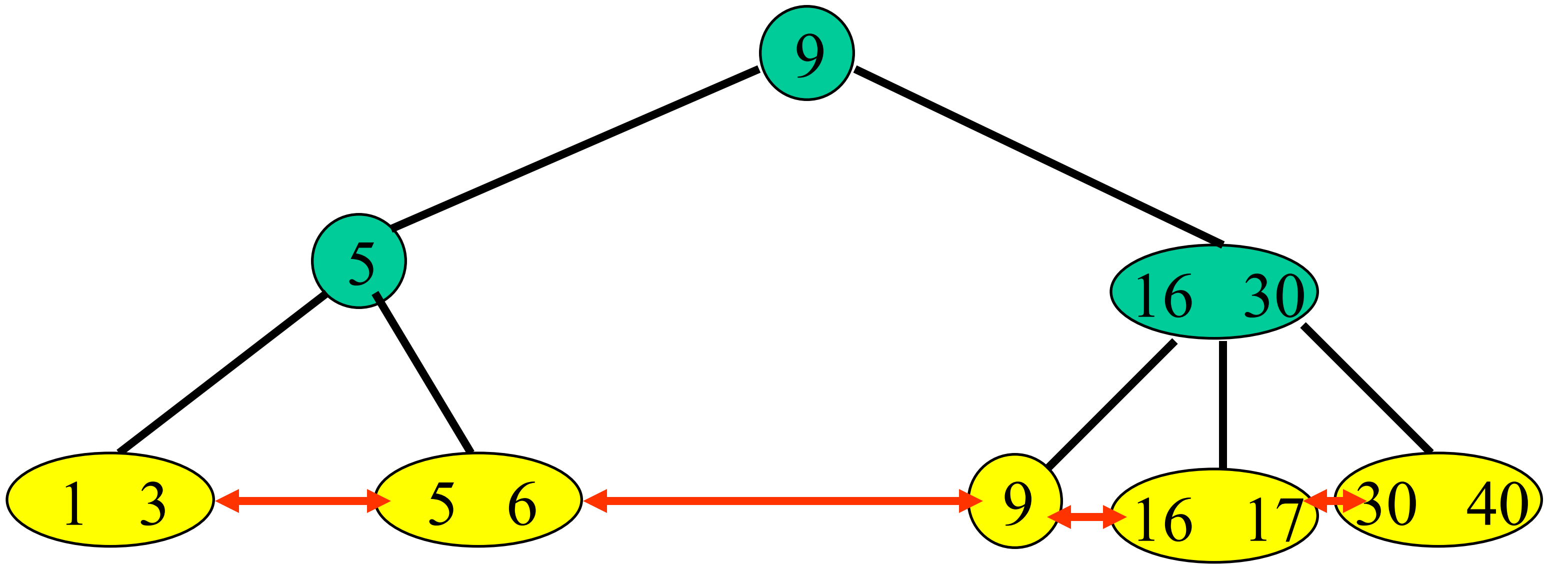
Example B+-tree



→ index node

→ leaf/data node

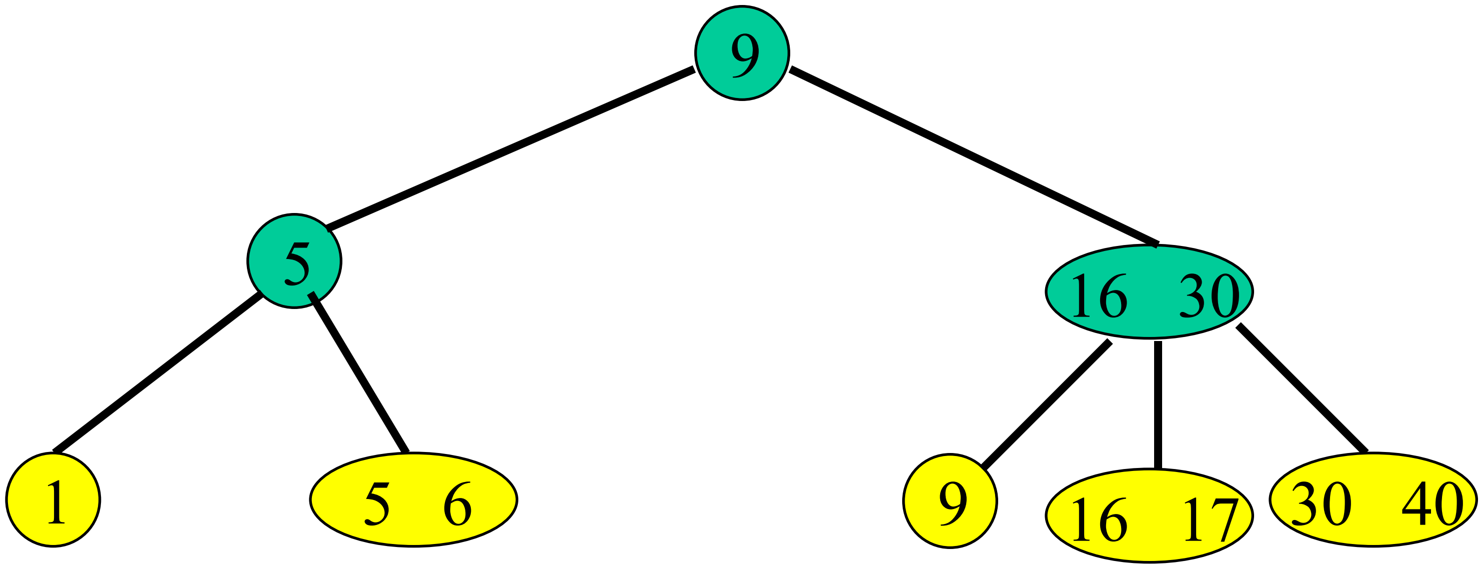
B+-tree—Search



key = 5

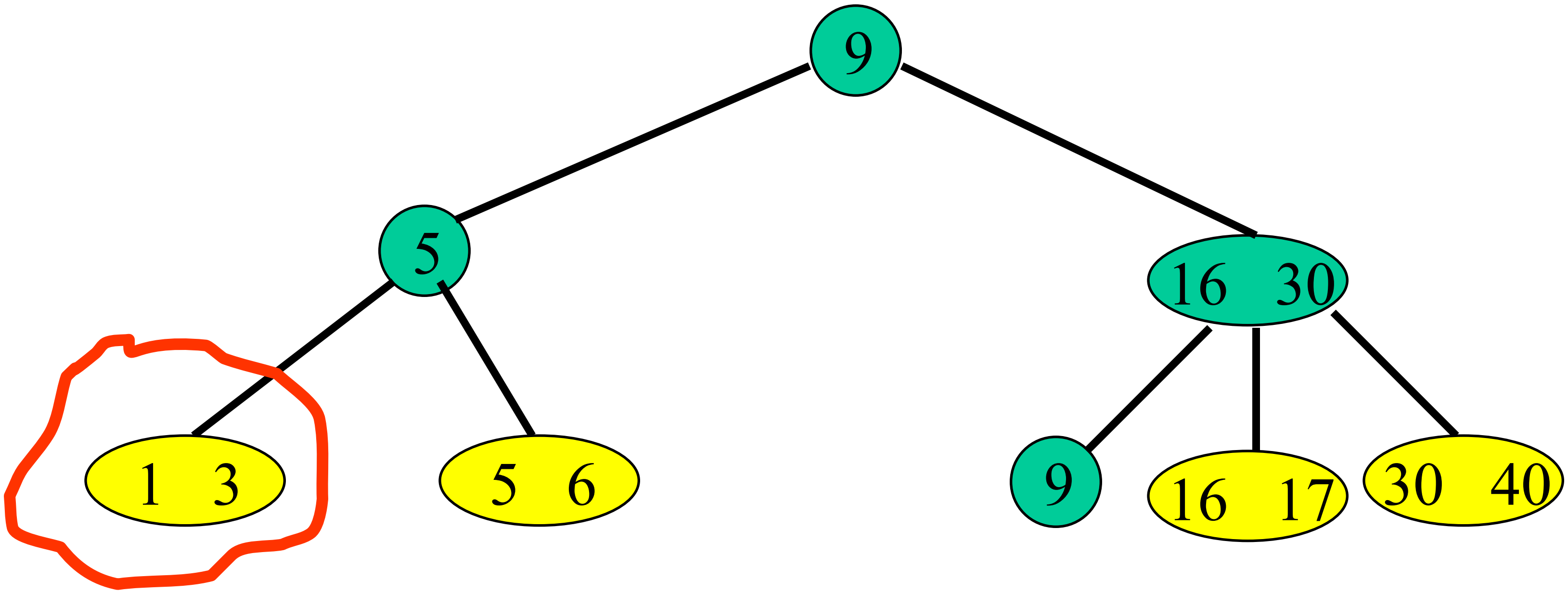
6 <= key <= 20

B+-tree—Insert



Insert 10

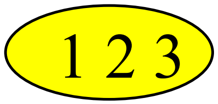
Insert



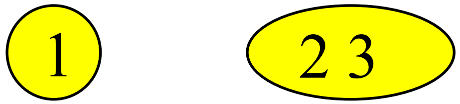
- Insert a pair with key = 2.
- New pair goes into a 3-node.

Insert Into A 3-node

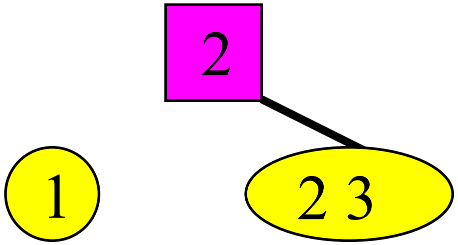
- Insert new pair so that the keys are in ascending order.



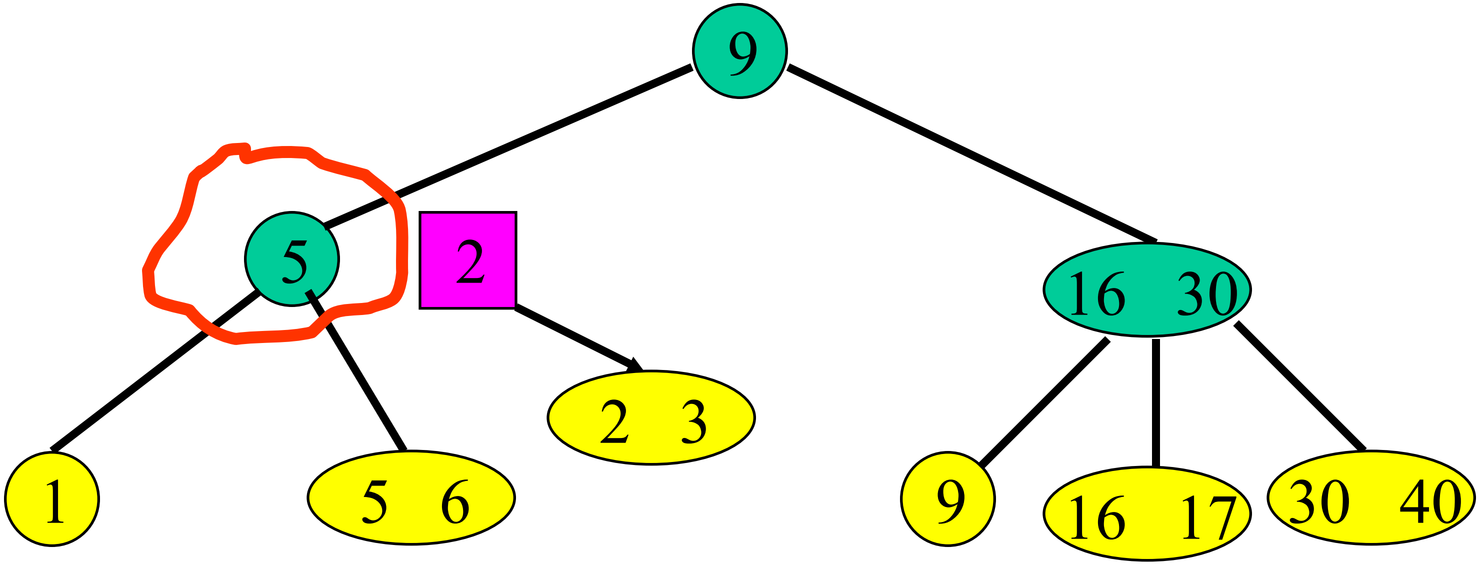
- Split into two nodes.



- Insert smallest key in new node and pointer to this new node into parent.

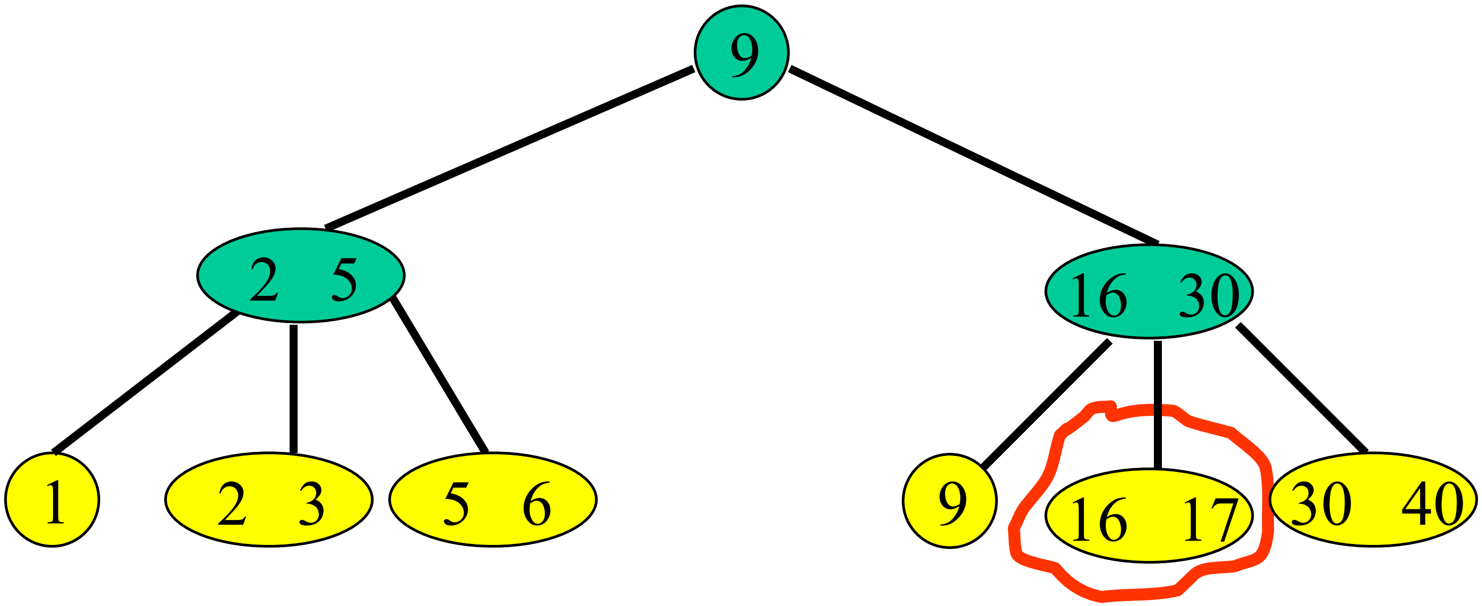


Insert



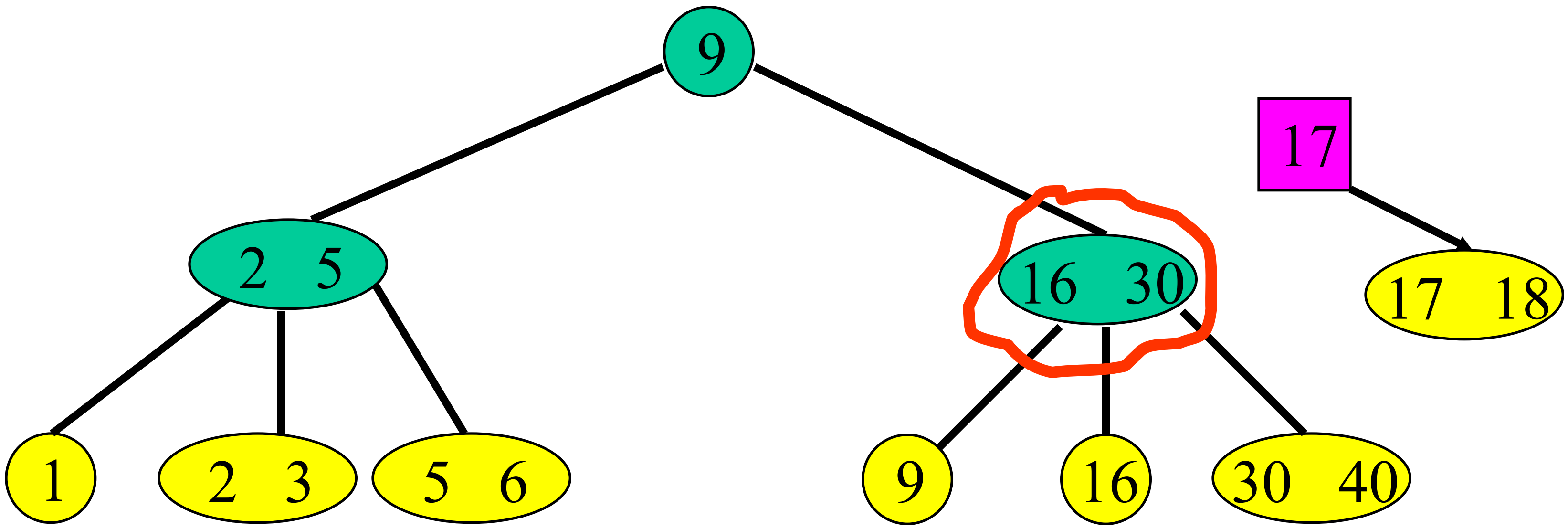
- Insert an index entry 2 plus a pointer into parent.

Insert



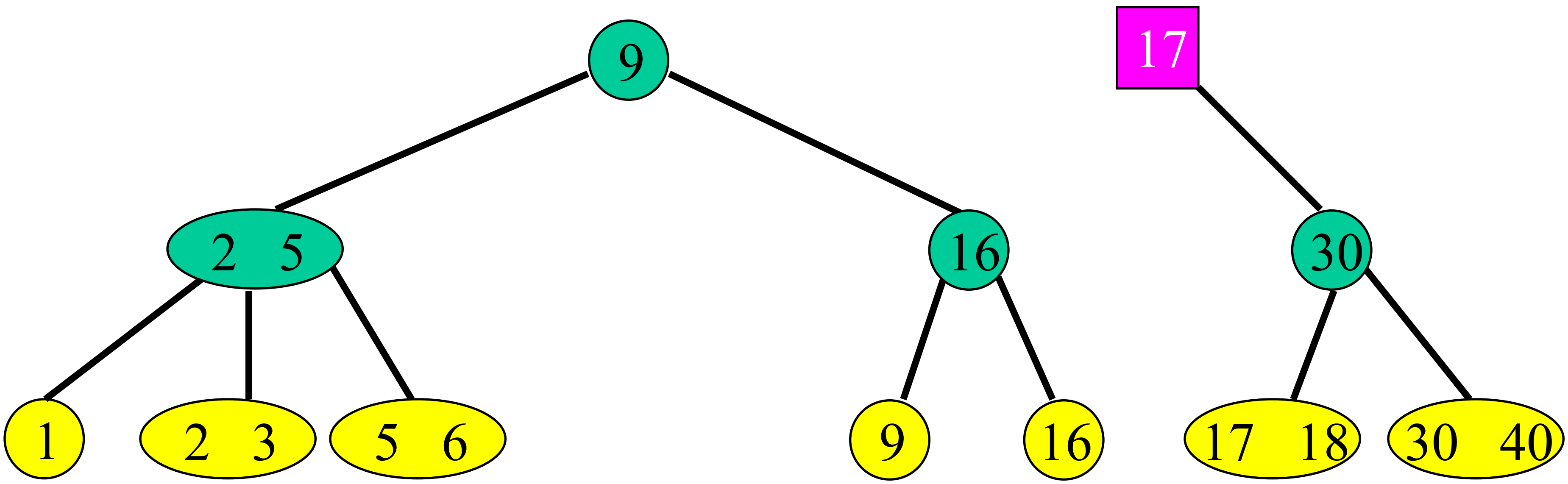
- Now, insert a pair with key = 18.

Insert



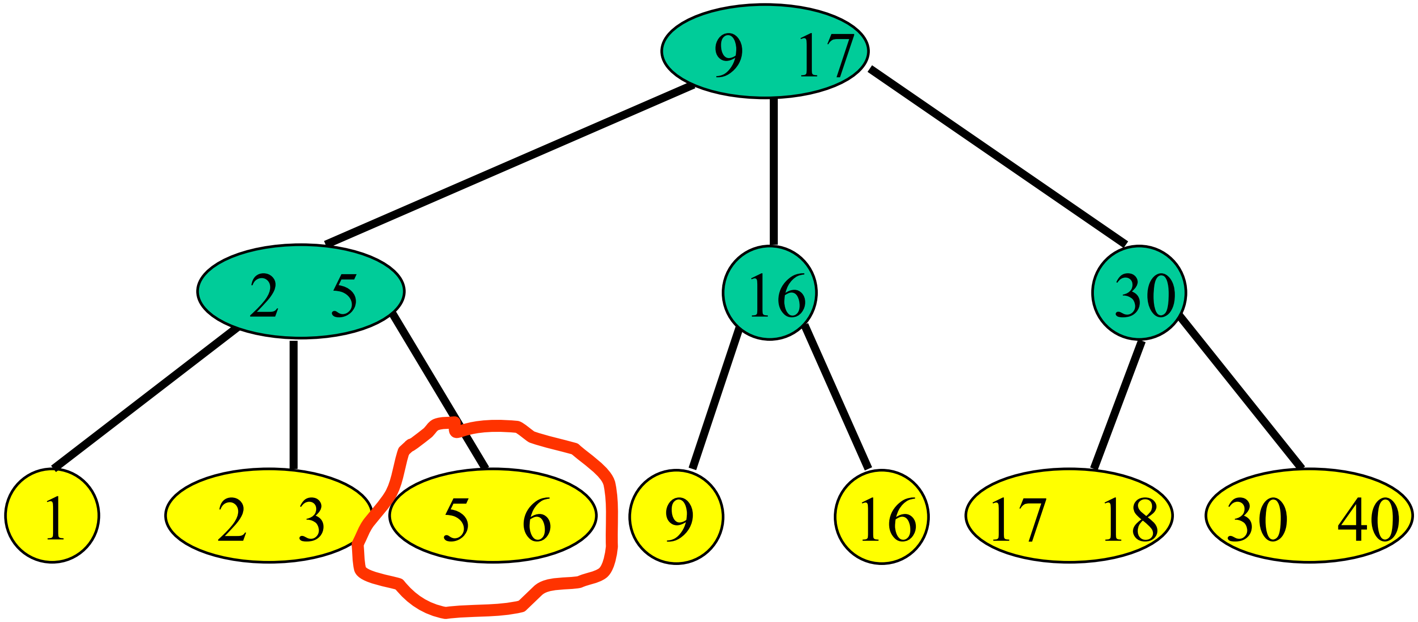
- Now, insert a pair with key = 18.
- Insert an index entry 17 plus a pointer into parent.

Insert



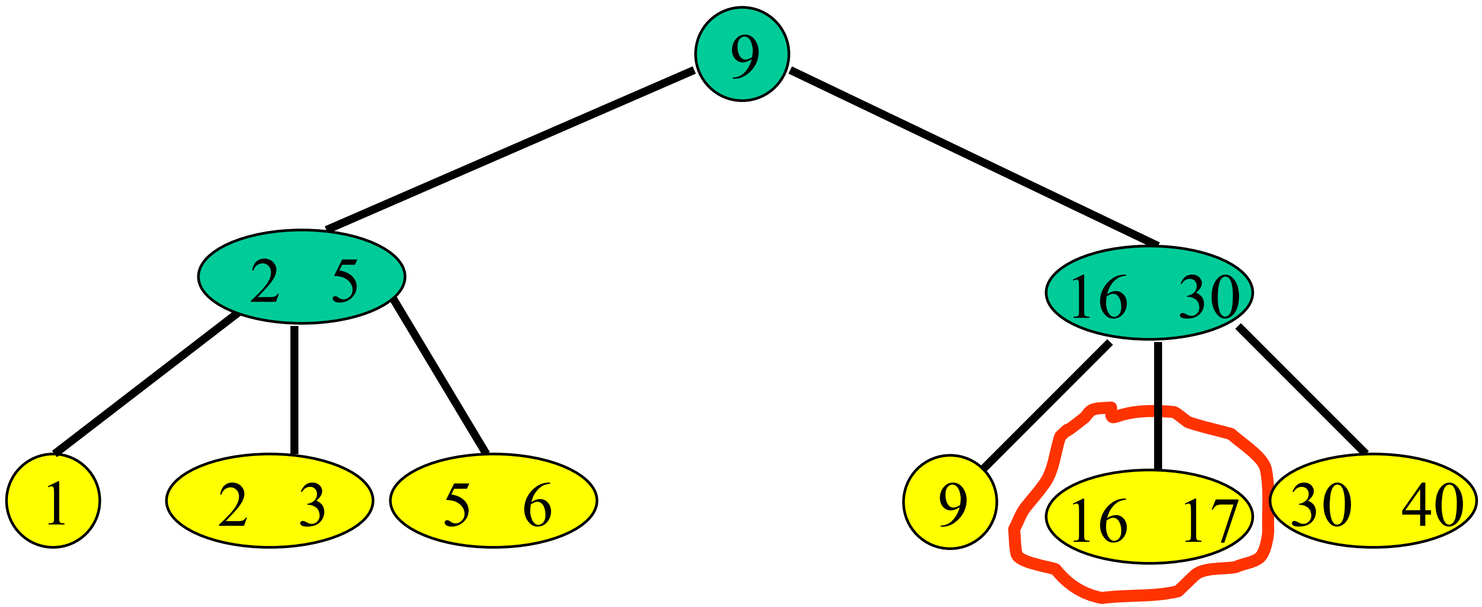
- Now, insert a pair with key = 18.
- Insert an index entry17 plus a pointer into parent.

Insert



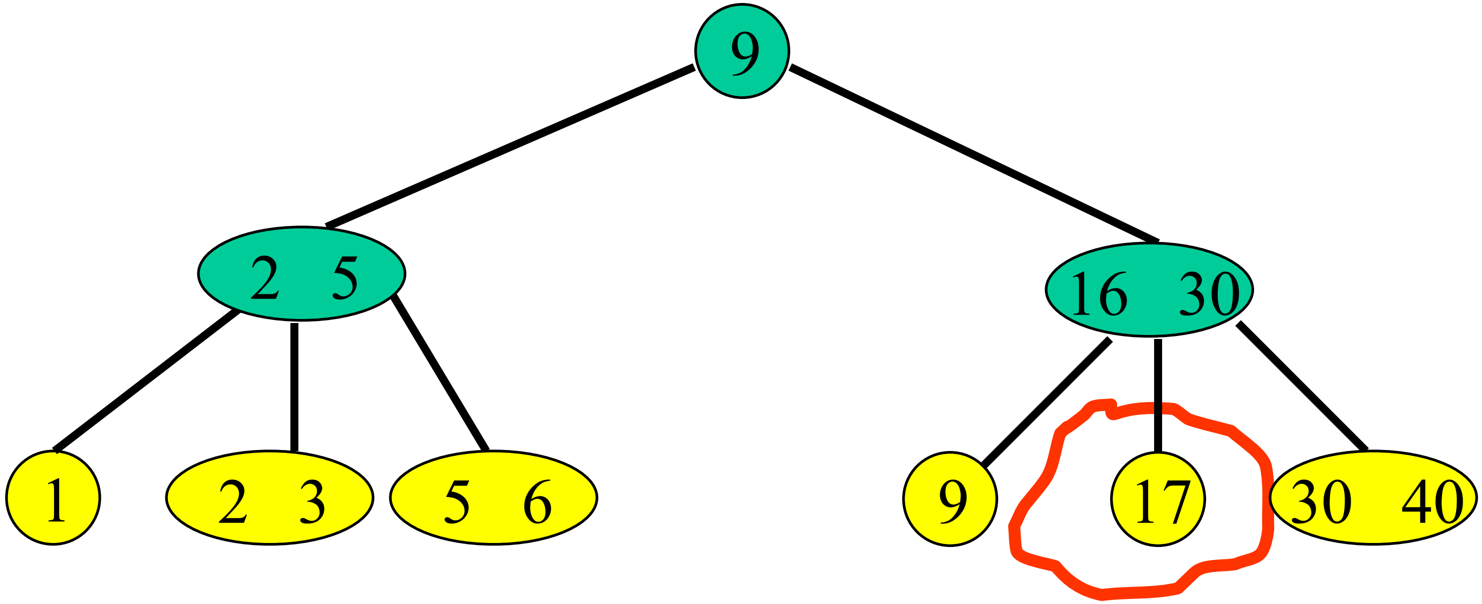
- Now, insert a pair with key = 7.

Delete



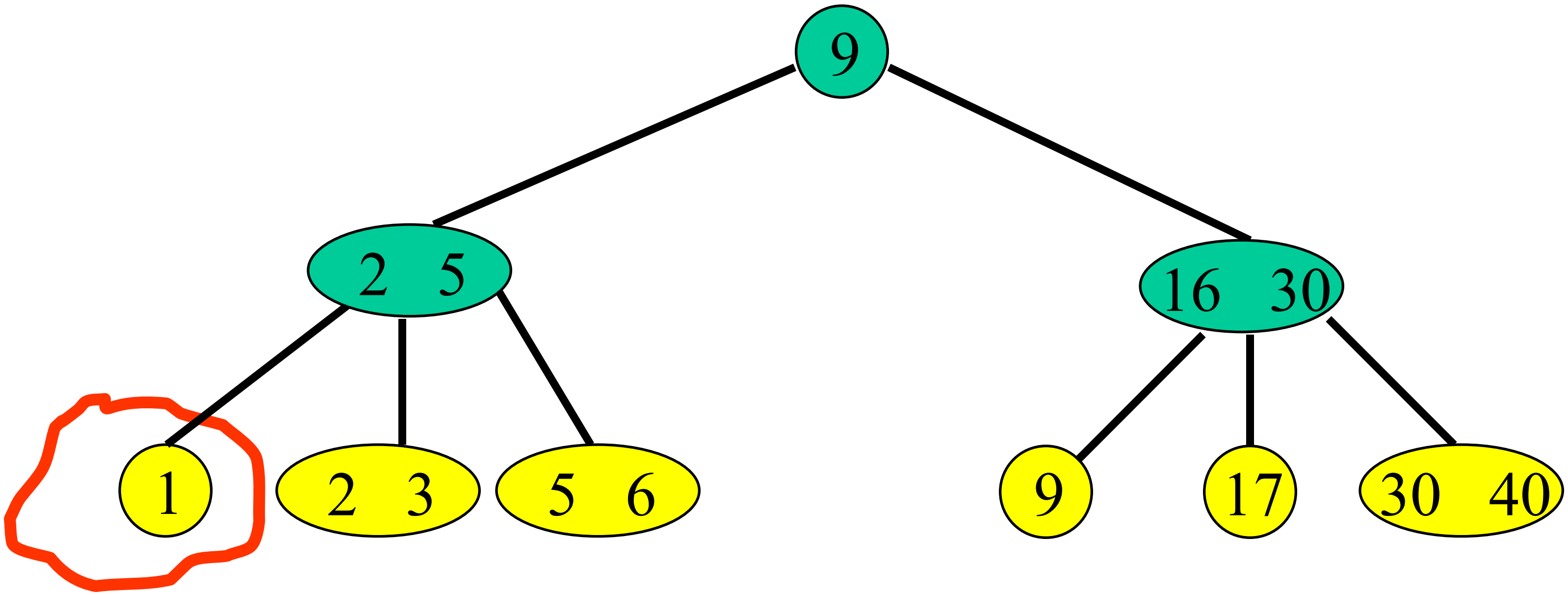
- Delete pair with key = 16.
- Note: delete pair is always in a leaf.

Delete



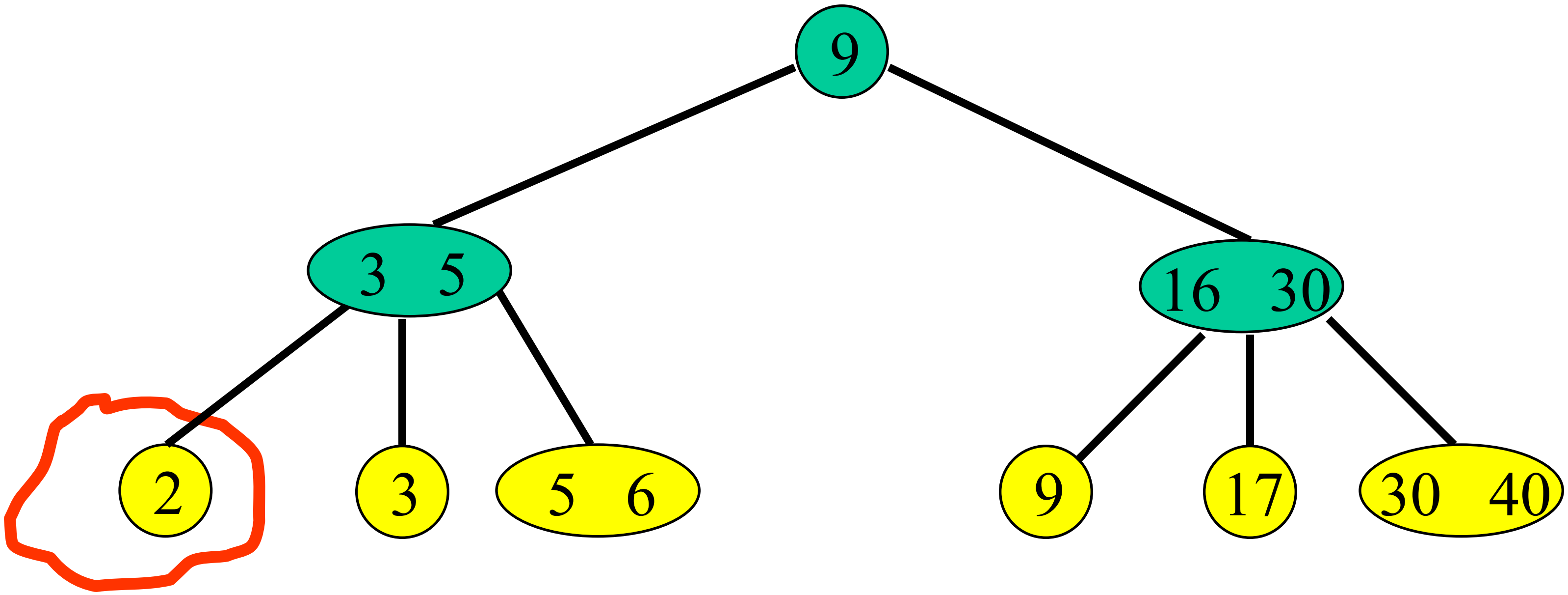
- Delete pair with key = 16.
- Note: delete pair is always in a leaf.

Delete



- Delete pair with key = 1.
- Get >= 1 from sibling and update parent key.

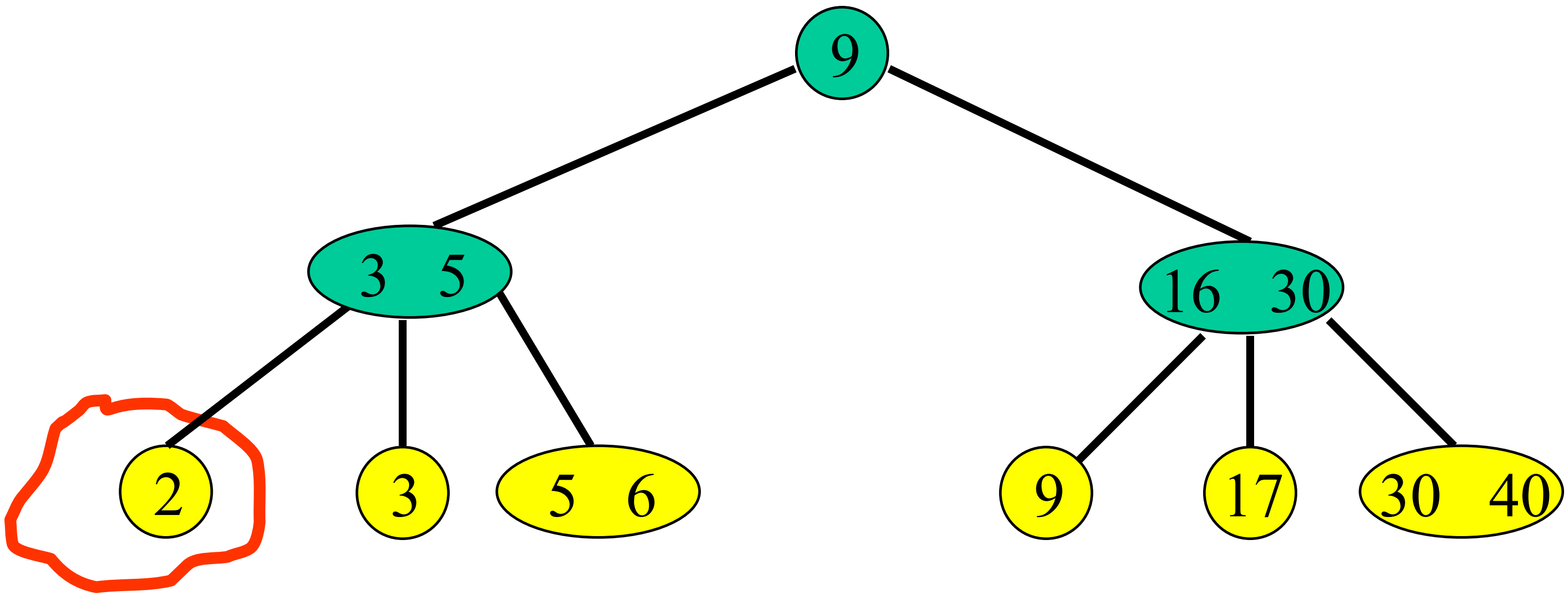
Delete



- Delete pair with key = 1.

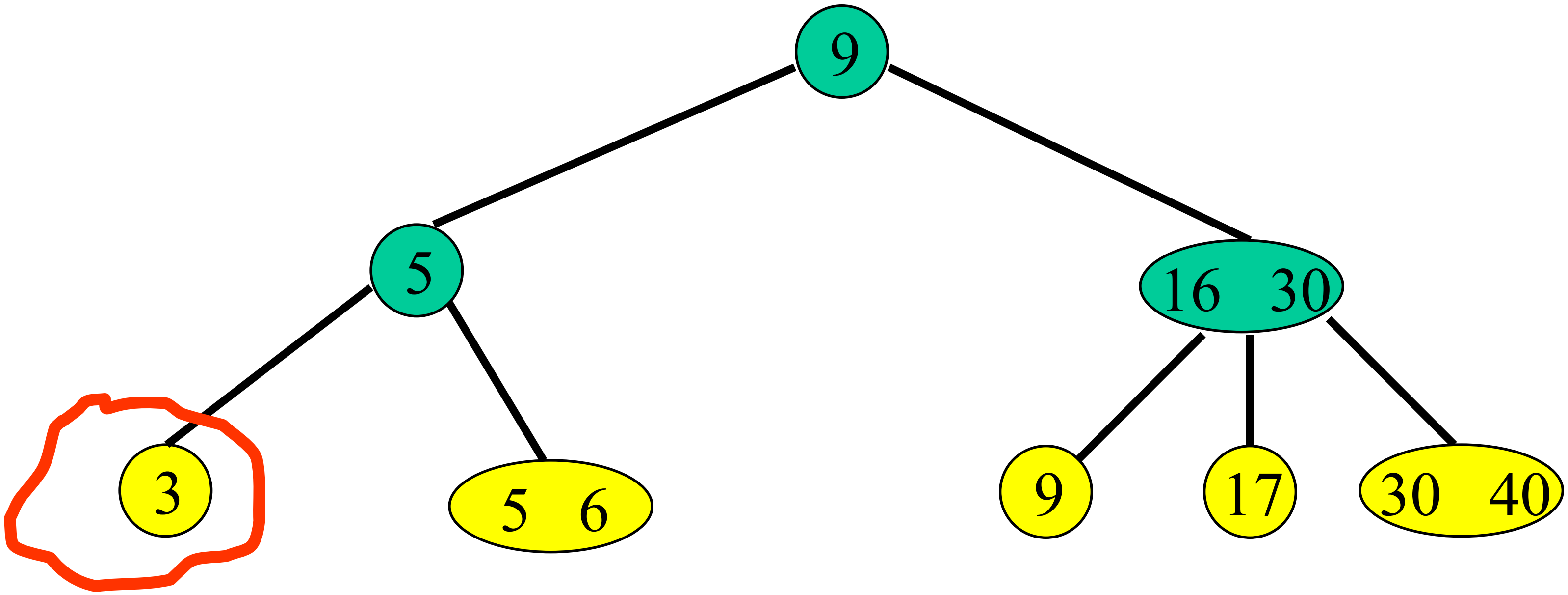
- Get >= 1 from sibling and update parent key.

Delete



- Delete pair with key = 2.
- Merge with sibling, delete in-between key in parent.

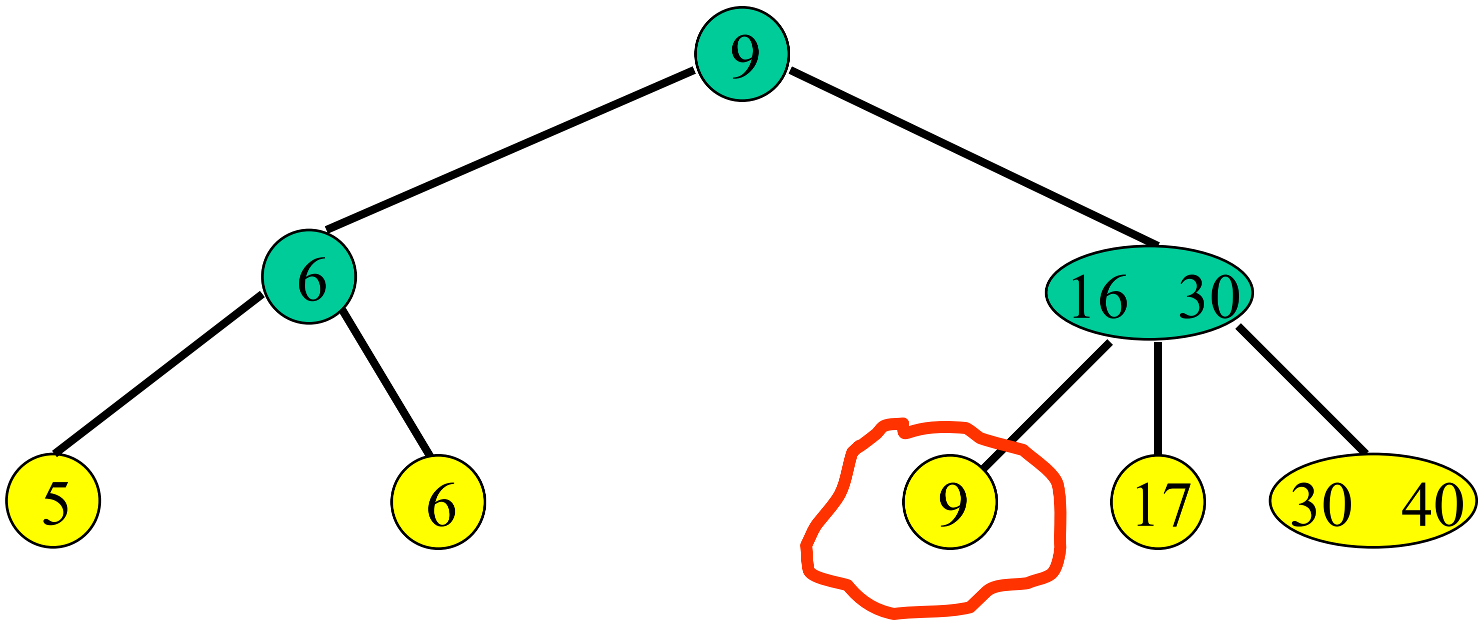
Delete



- Delete pair with key = 3.

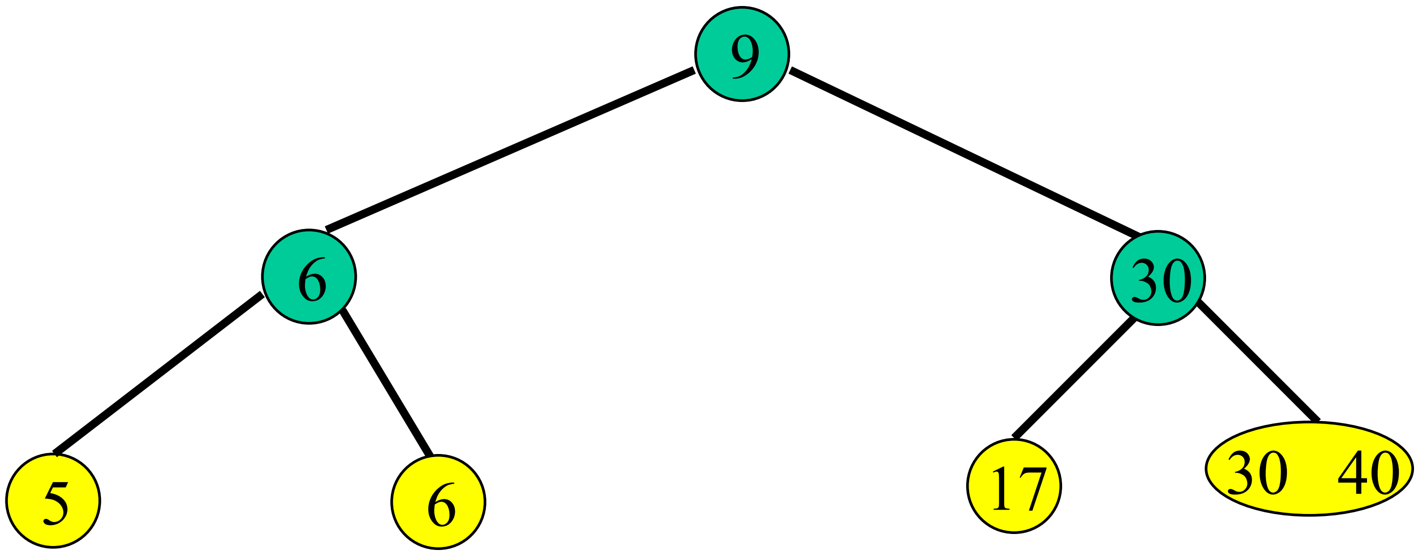
- Get >= 1 from sibling and update parent key.

Delete

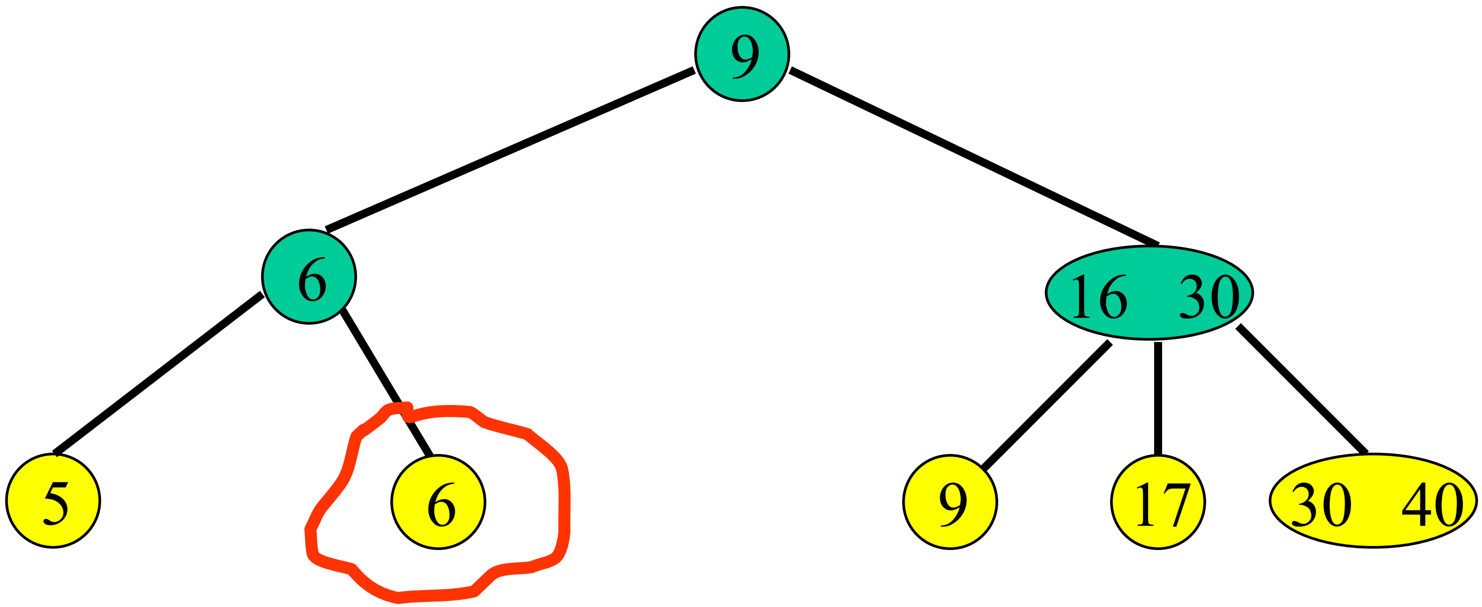


- Delete pair with key = 9.
- Merge with sibling, delete in-between key in parent.

Delete



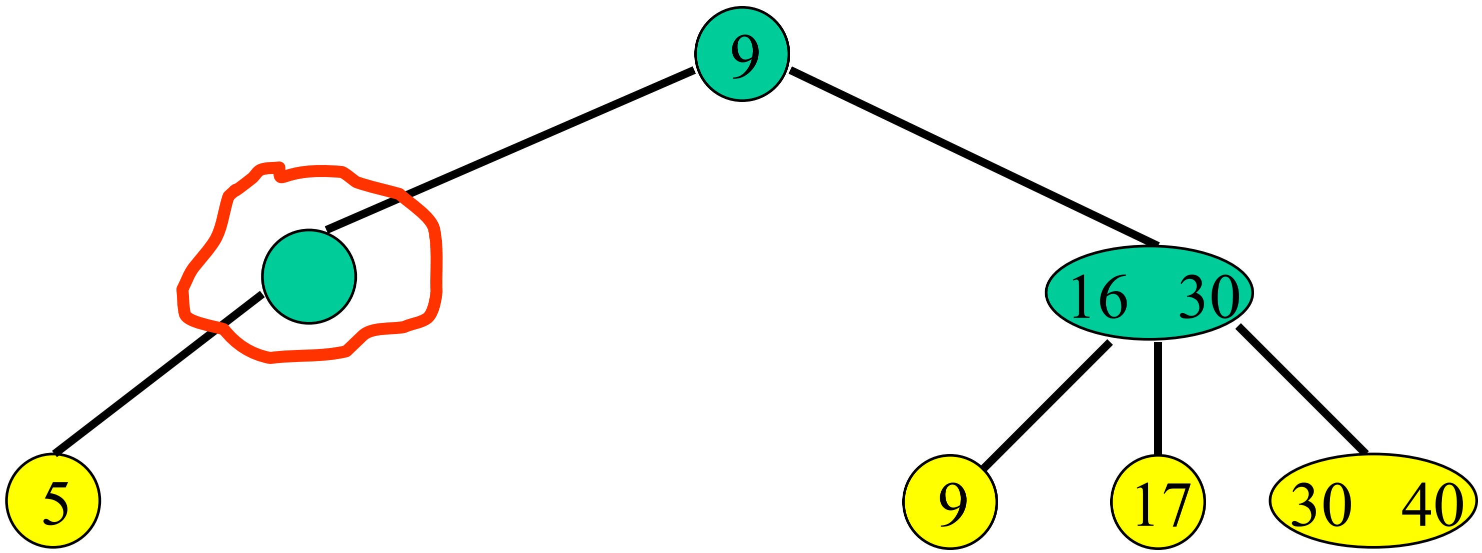
Delete



- Delete pair with key = 6.

- Merge with sibling, delete in-between key in parent.

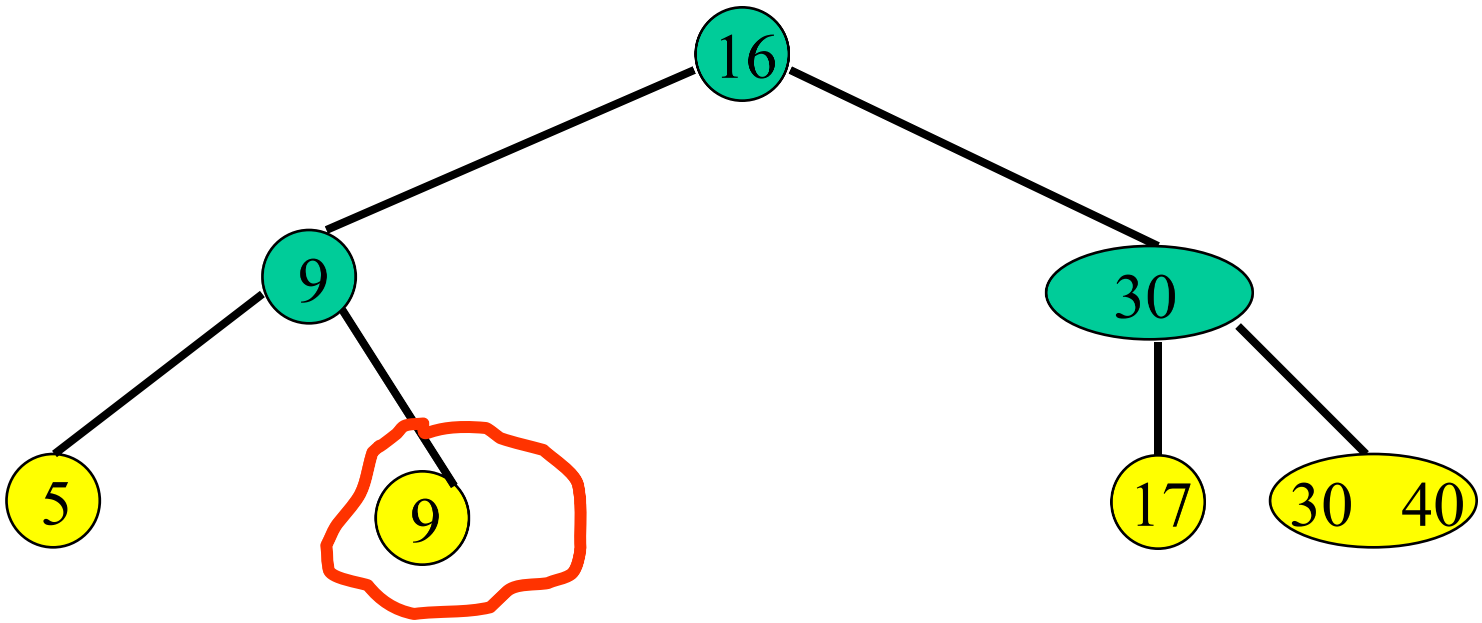
Delete



- Index node becomes deficient.

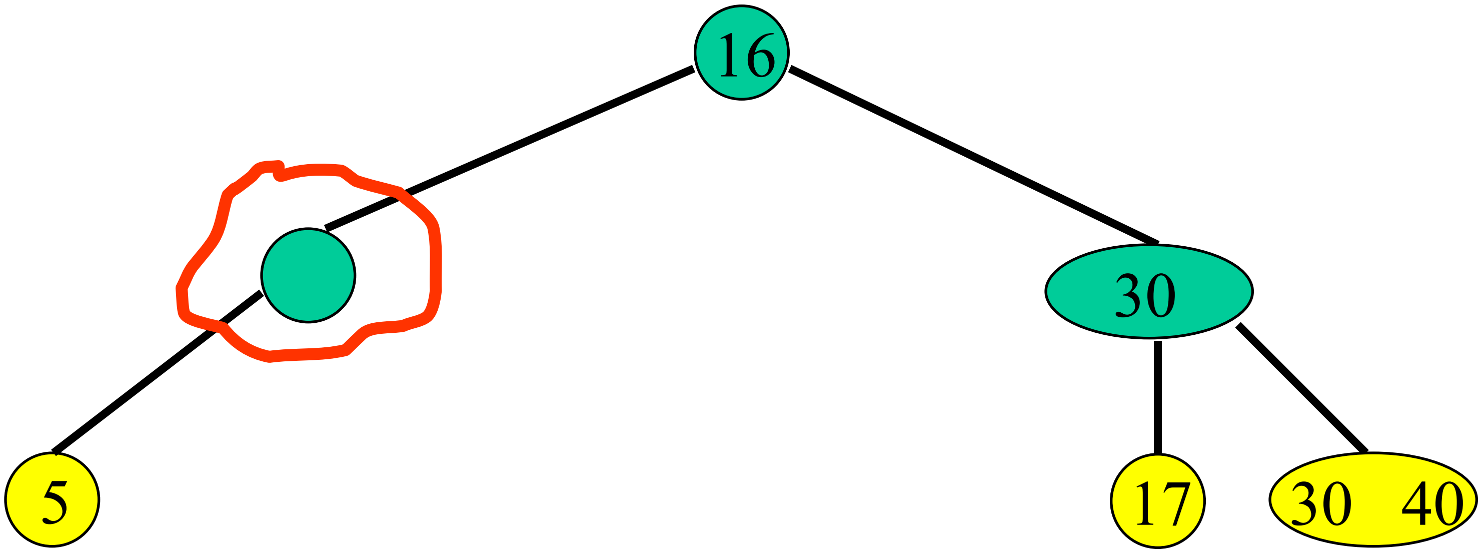
•Get >= 1 from sibling, move last one to parent, get parent key.

Delete



- Delete 9.
- Merge with sibling, delete in-between key in parent.

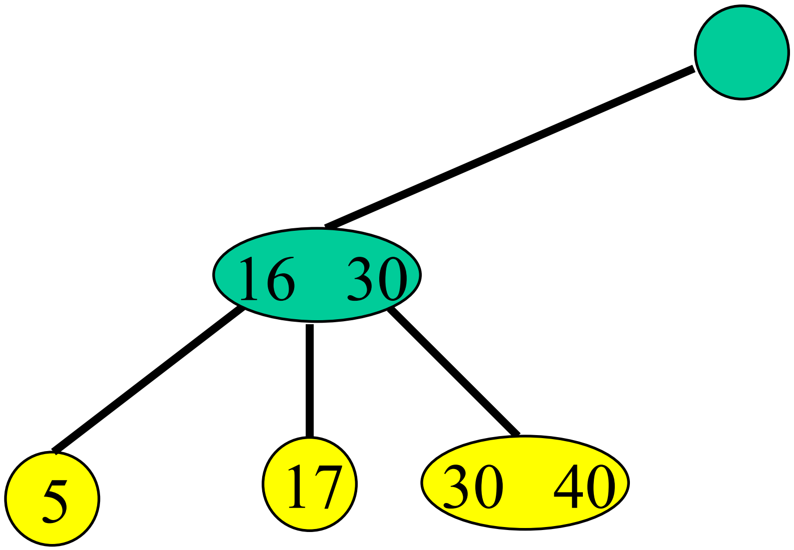
Delete



•Index node becomes deficient.

• Merge with sibling and in-between key in parent.

Delete



- Index node becomes deficient.
- It's the root; discard.