Splay Tree

Splay Tree (1985)

Daniel Dominic Kaplan Sleator
1953-
Professor, Computer Science,
Carnegie Mellon University

Robert Tarjan
1948-
Distinguished University Professor,
Computer Science, Princeton University
Recipient of Turing Award 1986

Outline

- Definition of Splay Tree
- Splay Operations
- Examples

Splay Tree

- Splay tree is a self-adjusting binary search tree (BST).
- It contains all the operations of BSTs, like searching, insertion and deletion, but followed by another operation called splaying.
- Splaying is a sequence of tree rotations that promotes the most recently accessed element to the root of the tree.
  - Searching: Promote the node that contains the searched element to the root.
  - Insertion: Promote the newly inserted node to the root.
  - Deletion: Promote the node whose child has been deleted to the root.

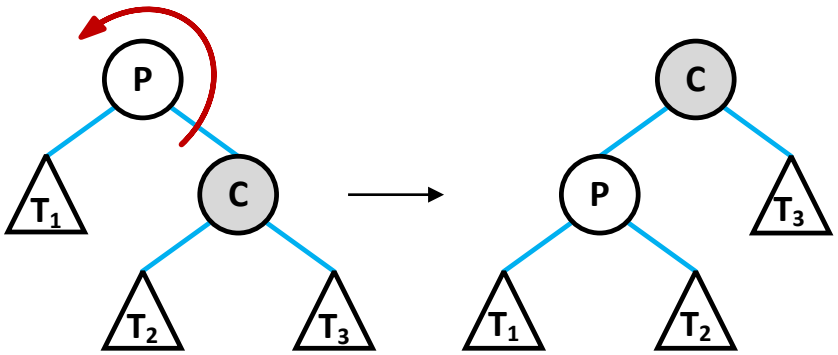
Splay Tree

- The basic idea is to improve the tree's performance by reducing the access time for frequently accessed nodes.
- A node can be splayed at most two levels at a time.
- If after one round of splay, the node is still not the root then we continue to splay the node until we reach the root.

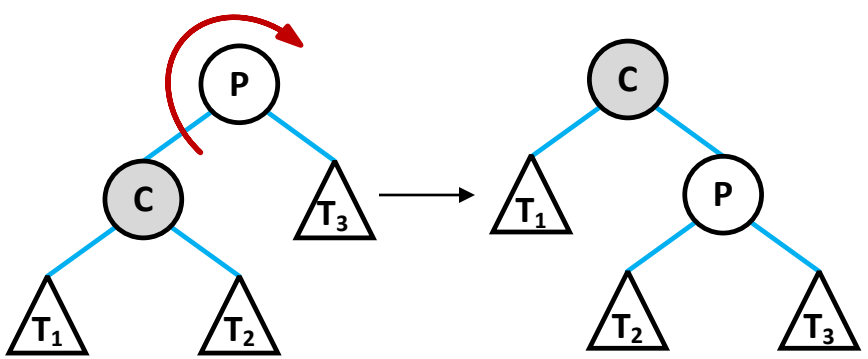
Splaying Algorithm

- The algorithm depends on the node's orientation to its parent and grandparent.
- When the node does not have a grandparent,
  - We promote the node to the position of its parent (i.e., root).
  - We simply perform a left or right rotation to bring the node to the root.

2 Types of Orientation
(when the child does not have a grandparent)



If the node is a **right**-child, rotate **left** about the parent.

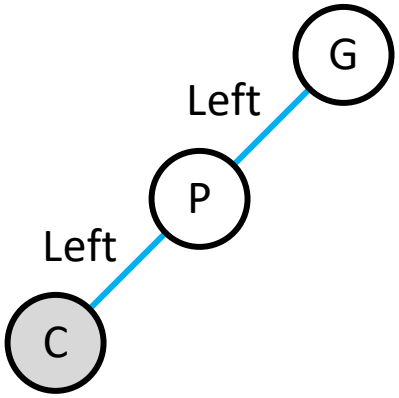
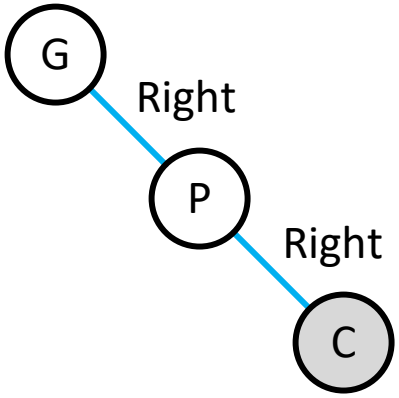
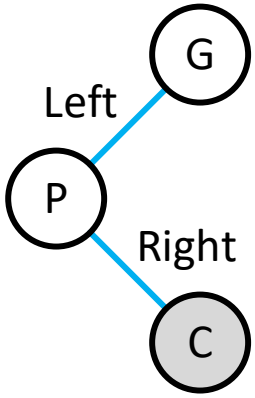
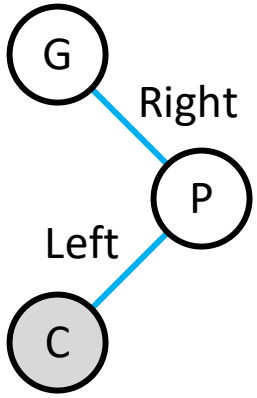


If the node is a **left**-child, rotate **right** about the parent.

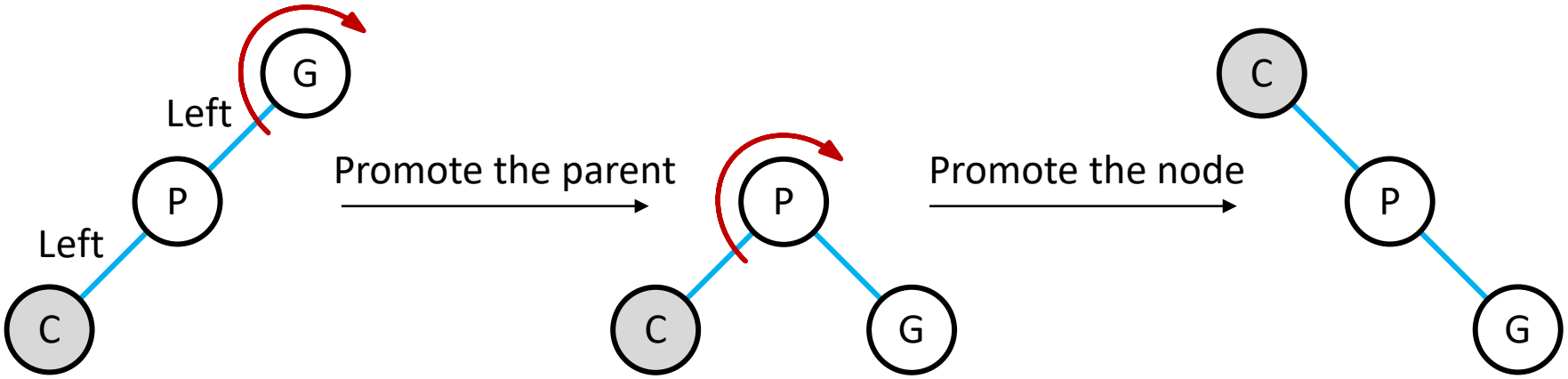
Splaying Algorithm

- When the node has a grandparent,
  - We promote the node to the position of its grandparent.
  - This leads to 4 possible orientations based on the child, parent and grandparent nodes.

4 Types of Orientation (when the child has a grandparent)

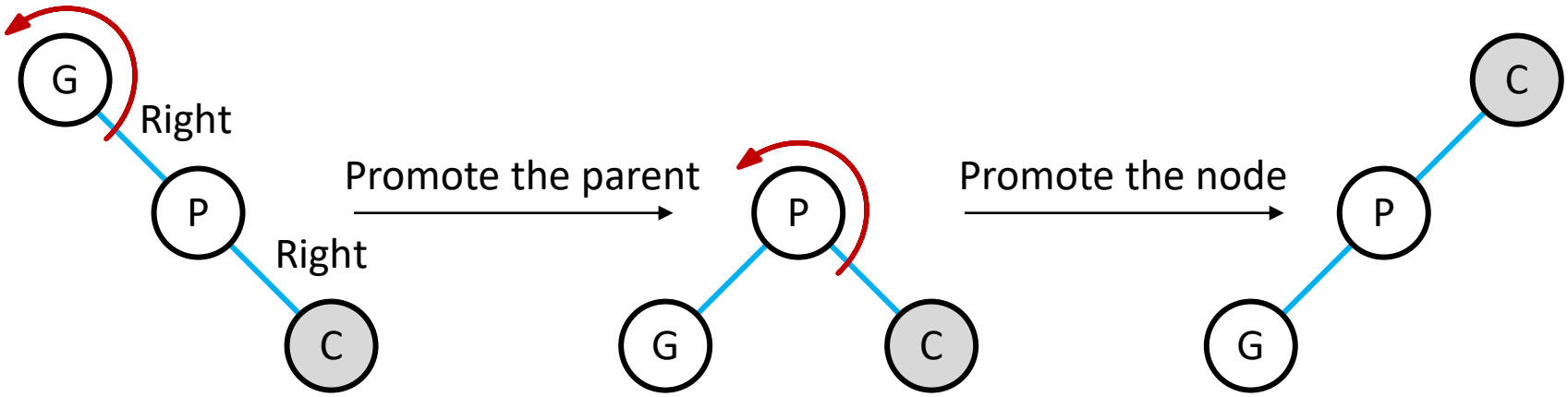
			
Left-left orientation	Right-right orientation	Left-right orientation	Right-left orientation
Promote the parent and then the node	promote the parent and then the node	promote the node twice	promote the node twice
Zig-zig rotation (Right-right rotation)	Zag-zag rotation (Left-left rotation)	Zag-zig rotation (Left-right rotation)	Zig-zag rotation (Right-left rotation)

1. Left-Left Orientation (Zig-zig Rotation)



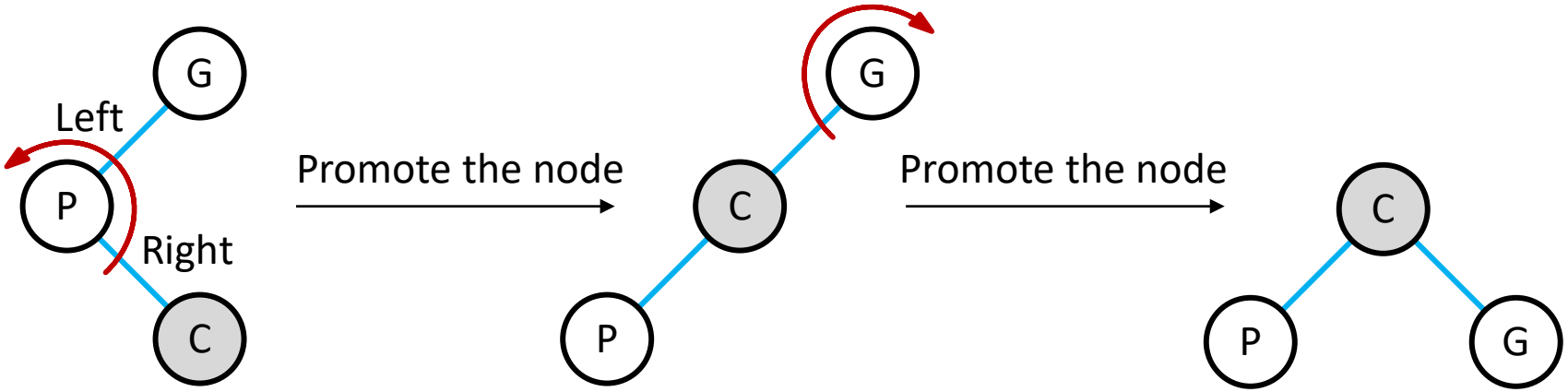
- **Left-left orientation**: promote the parent, promote the node
- **Promoting a node:**
  - If the node is a **right**-child, rotate **left** about the parent.
  - If the node is a **left**-child, rotate **right** about the parent.

2. Right-right Orientation (Zag-zag Rotation)



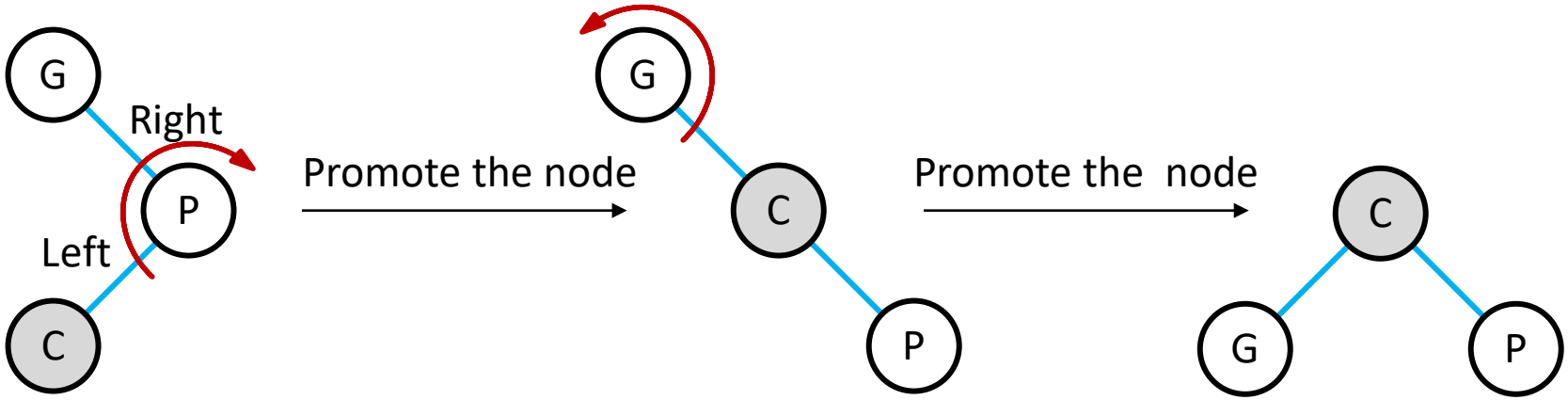
- **Right-right orientation**: promote the parent, promote the node
- **Promoting a node:**
  - If the node is a **right**-child, rotate **left** about the parent.
  - If the node is a **left**-child, rotate **right** about the parent.

3. Left-right Orientation (Zag-zig Rotation)



- **Left-right orientation**: promote the node twice
- **Promoting a node:**
  - If the node is a **right**-child, rotate **left** about the parent.
  - If the node is a **left**-child, rotate **right** about the parent.

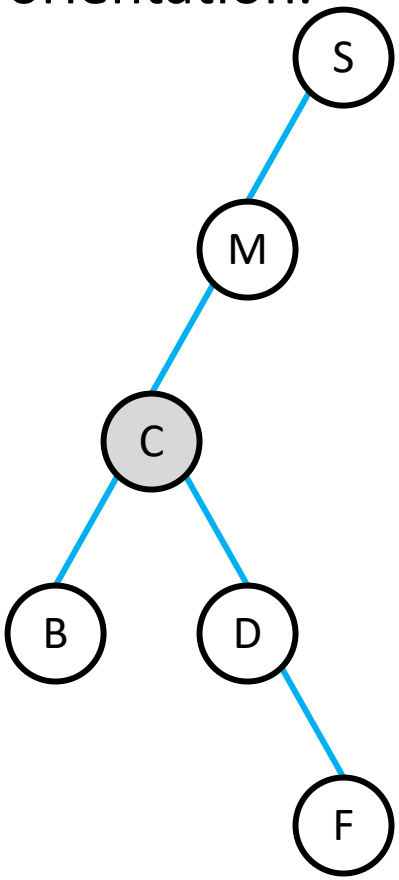
4. Right-left Orientation (Zig-zag Rotation)



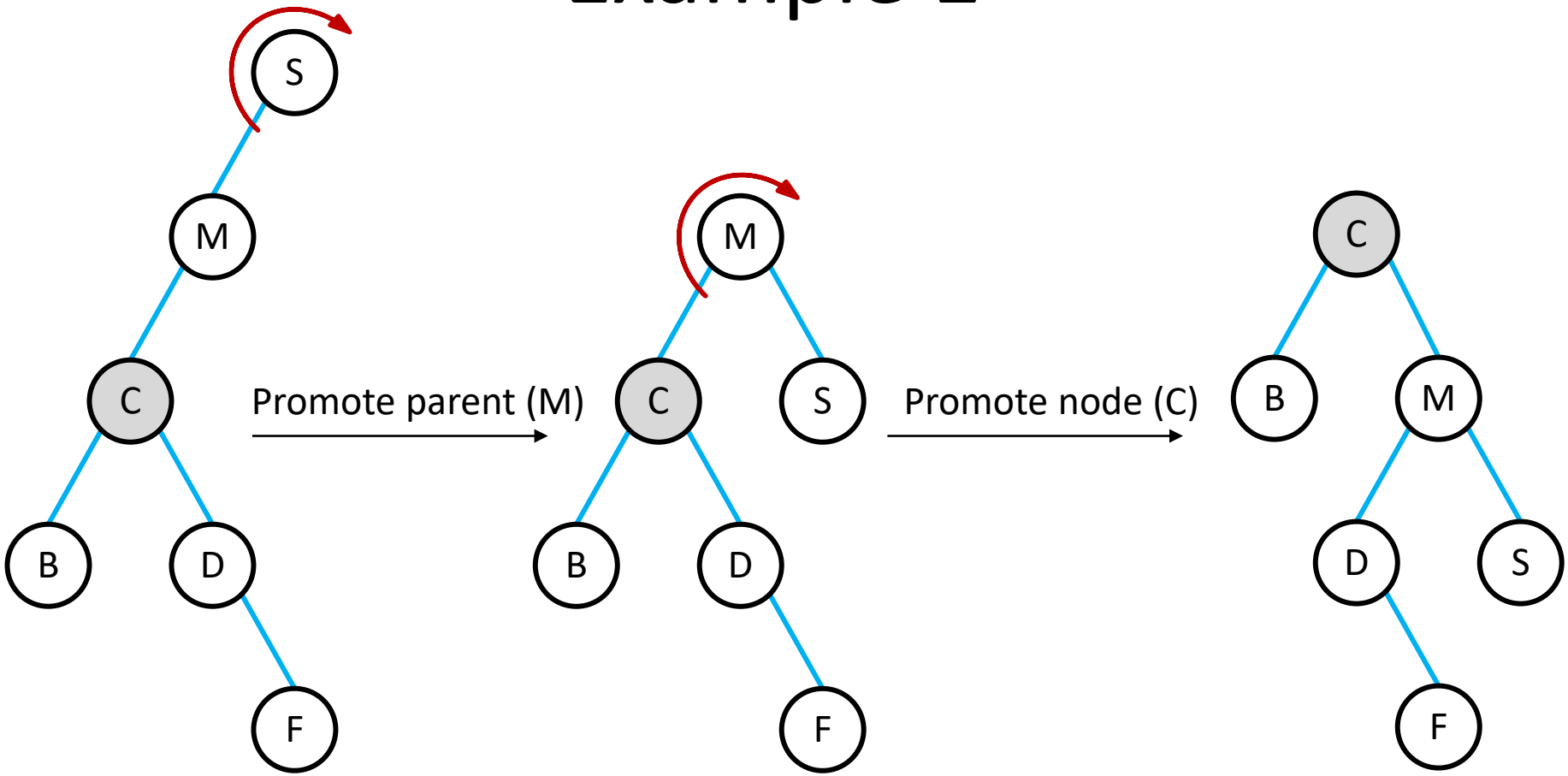
- **Right-left orientation**: promote the node twice
- **Promoting a node:**
  - If the node is a **right**-child, rotate **left** about the parent.
  - If the node is a **left**-child, rotate **right** about the parent.

Example 1

- What would be the result of splaying **C** to the root of the following tree?
  - Hint: Observe the orientation.



Example 1

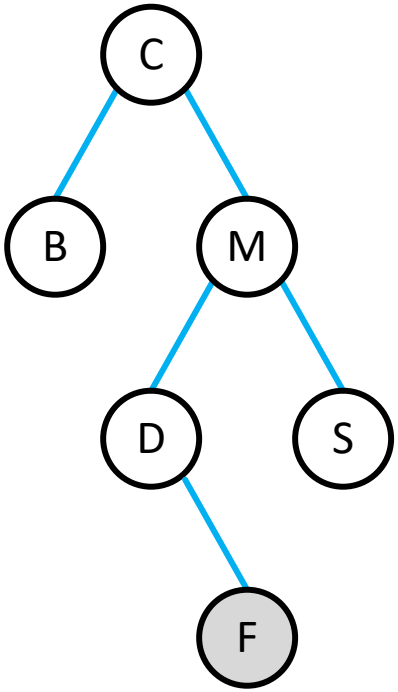


Note:

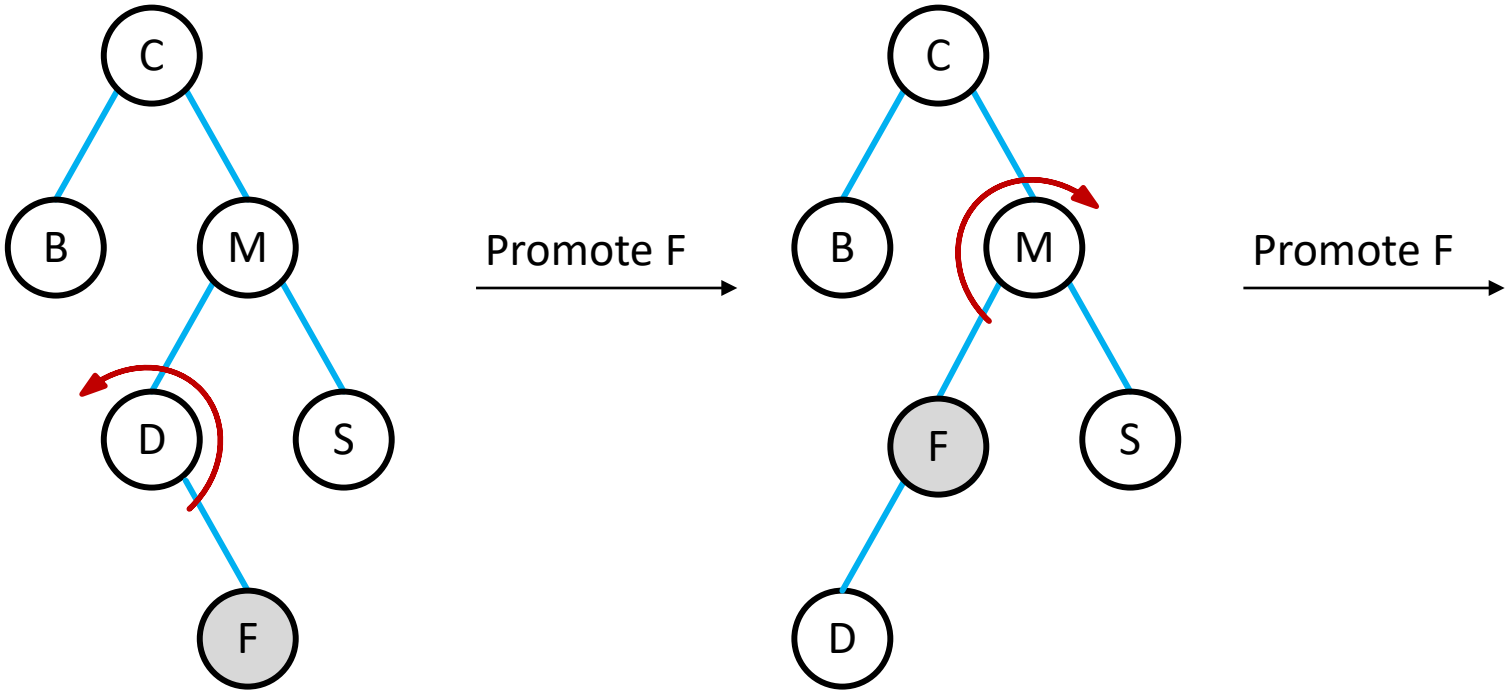
- Left-left orientation
- Splay C ⟶ Promote the parent (M) and then the node (C)

Example 2

- What would be the result of splaying **F** to the root of the following tree?



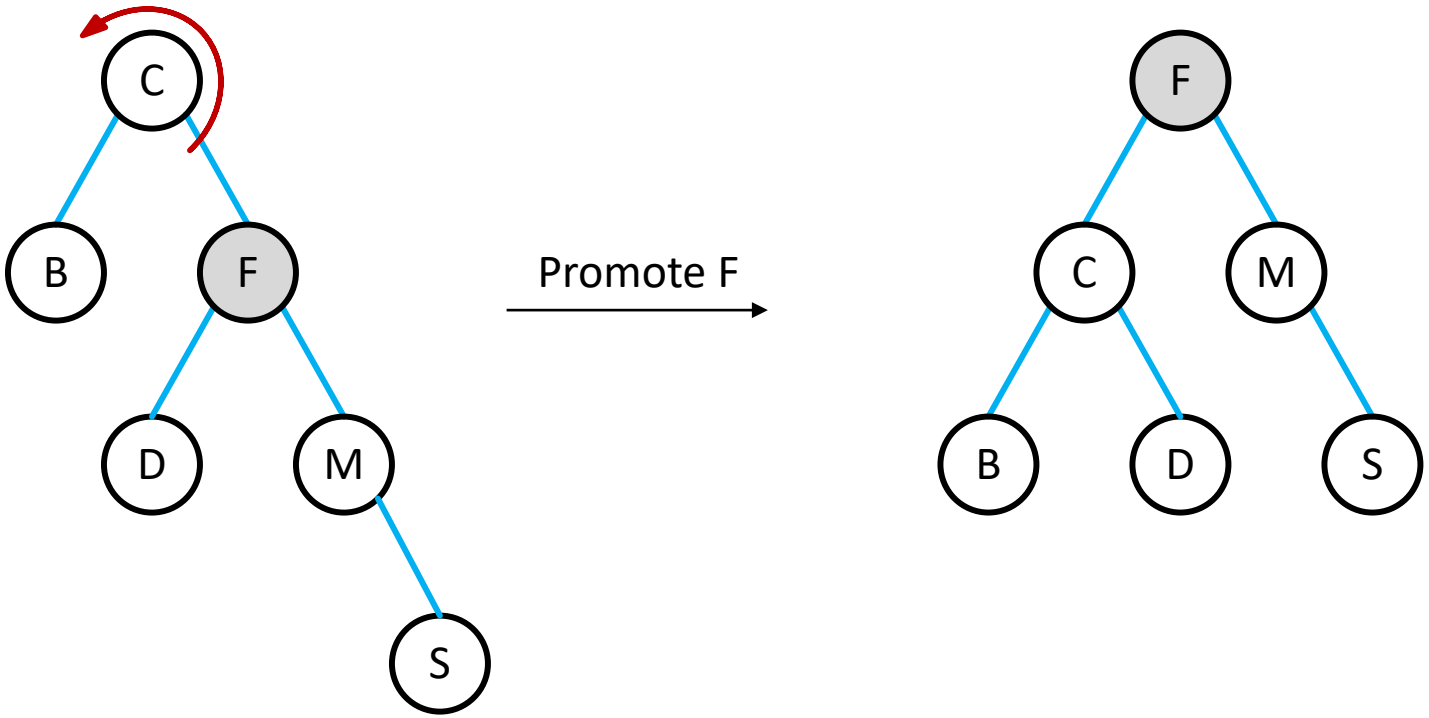
Example 2



Note:

- Left-right orientation
- Splay F ⟶ Promote F twice

Example 2

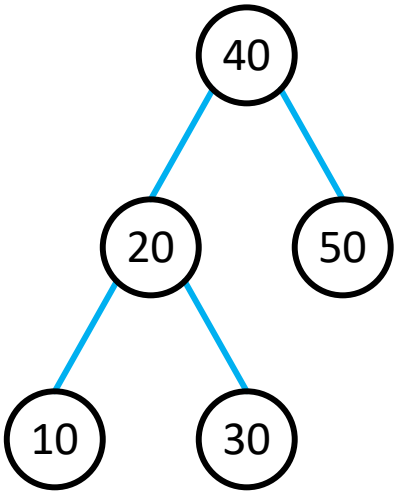


Note:

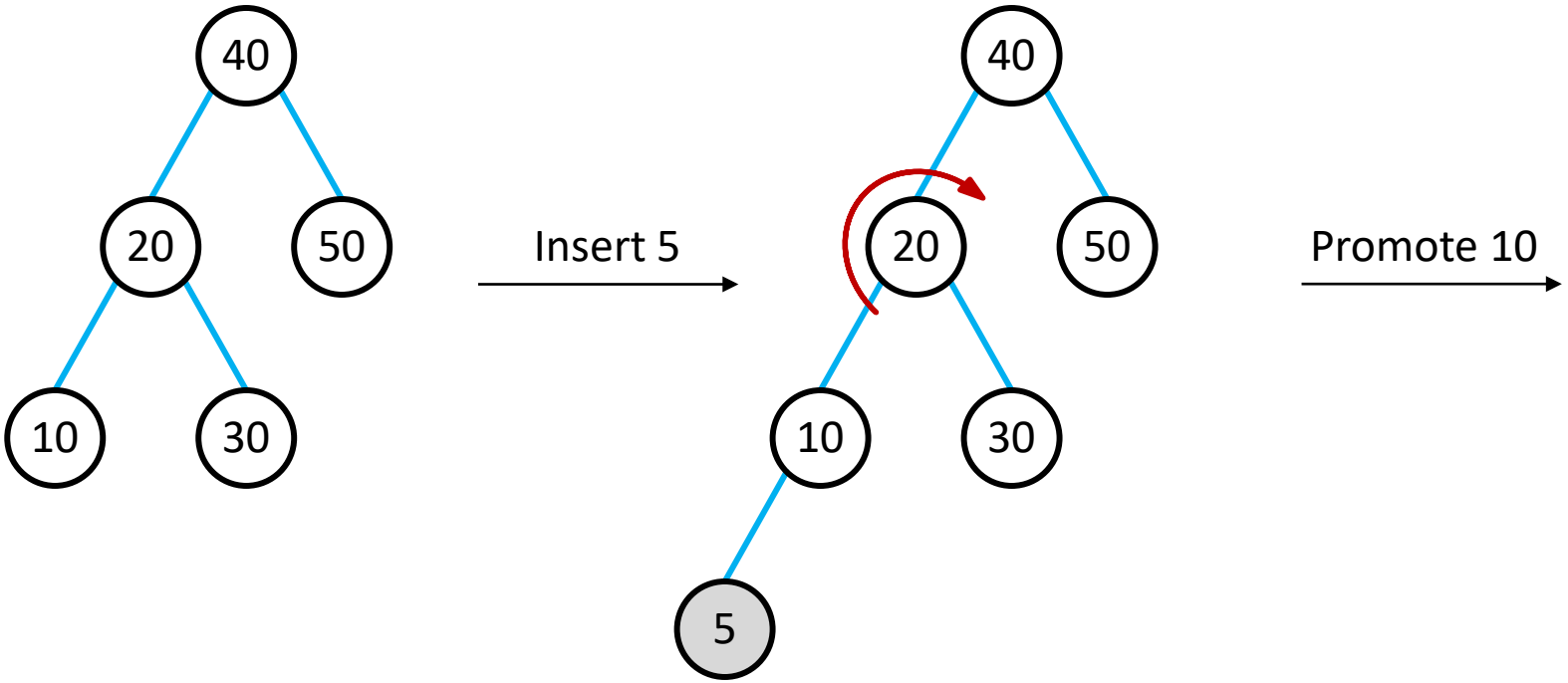
- F is still not the root.
- Splay F again.
- F has no grandparent.

Example 3

- What would be the result if we insert 5 in the following splay tree?



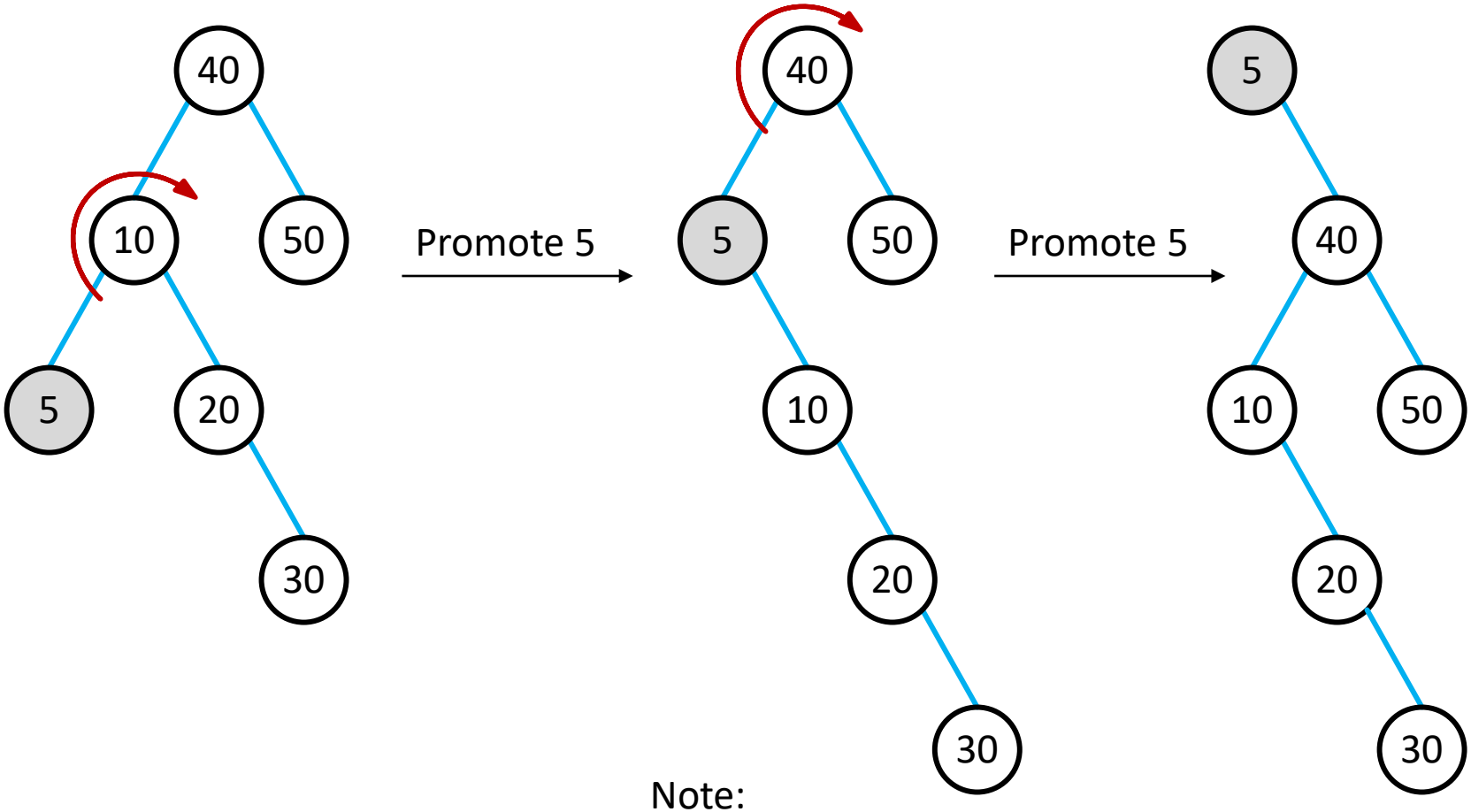
Example 3



Note:

- Left-left orientation
- Splay 5 ⟶ Promote the parent (10) and then the node (5)

Example 3

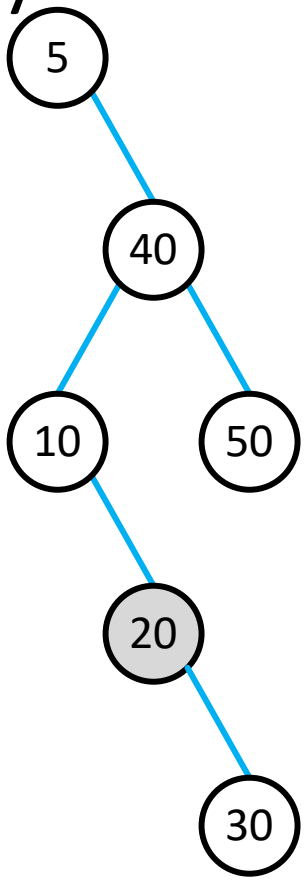


Note:

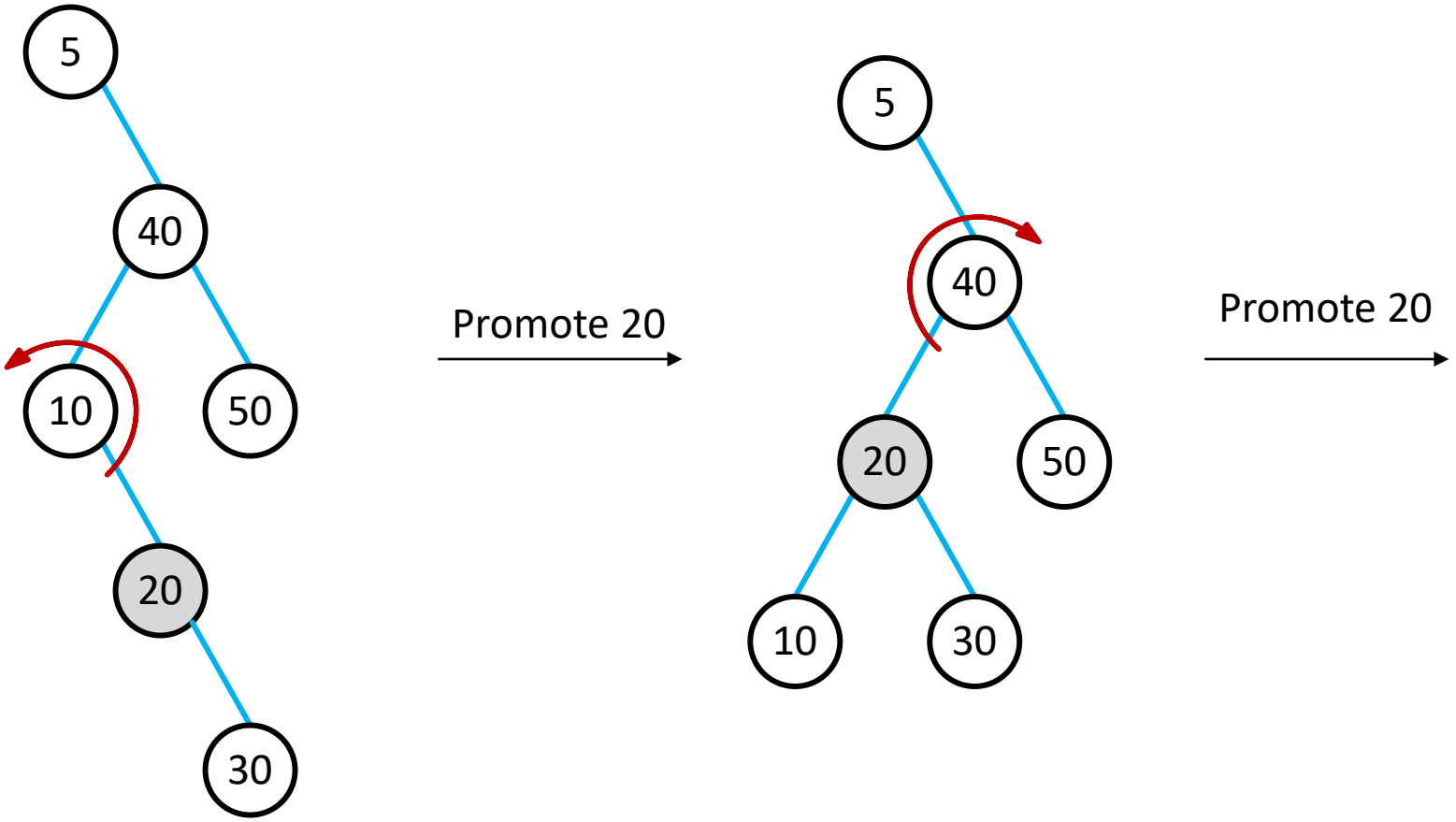
- 5 is still not the root.
- Splay 5 again.
- 5 has no grandparent.

Example 4

- What would be the result if we search 20 in the following splay tree?



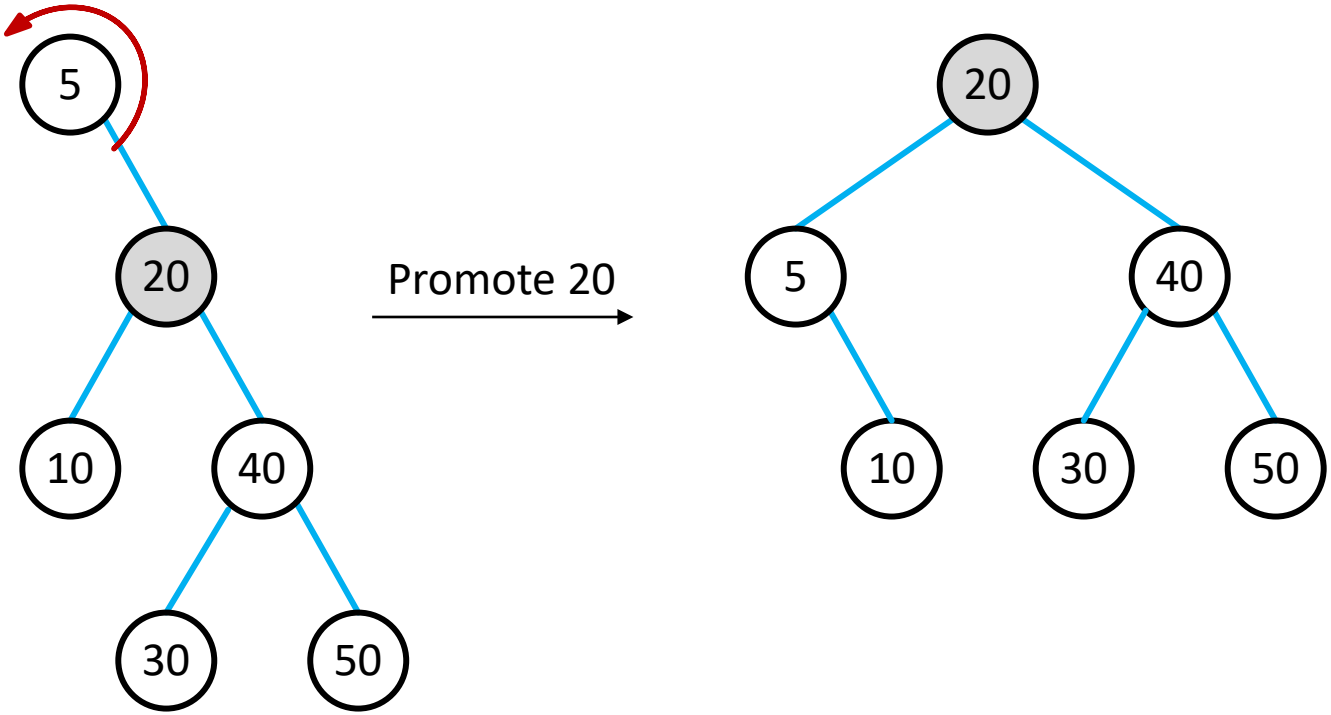
Example 4



Note:

- Left-right orientation
- Splay 20 ⟶ Promote 20 twice

Example 4

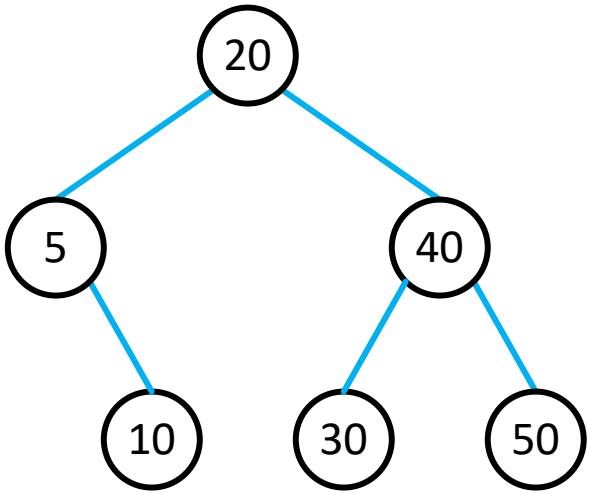


Note:

- 20 is still not the root.
- Splay 20 again.
- 20 has no grandparent.

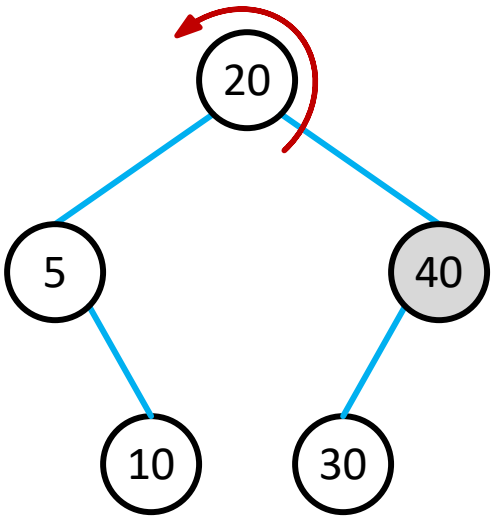
Example 5

- What would be the result if we delete 50 from the following splay tree?

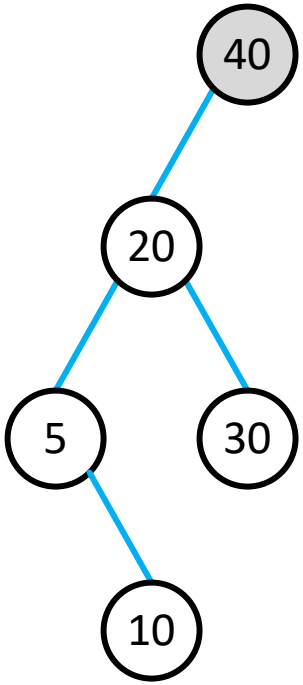


Example 5

Delete 50



Splay 40
(Promote 40)



Note:

- For deletion, promote the node (40) whose child has been deleted to the root.
- 40 has no grandparent.

Remarks on Splay Trees

- Splays trees allows the fast and efficient access of the frequently accessed elements.
  - Suitable for applications that require fast data structures, such as caching systems, and network routing algorithms.
- Splay trees do not guarantee a balanced tree structure.
  - Not suitable for applications that require guaranteed worst-case performance, such as real-time systems or safety-critical systems.