ITD62-124 Data Structure

Binary Search Tree

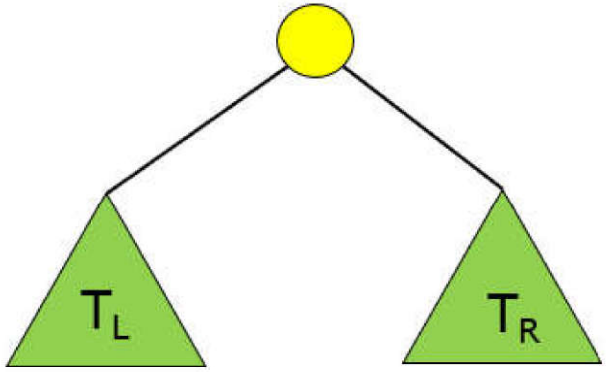
Outline:

- Introduction
- Binary Search Trees Operations

Introduction

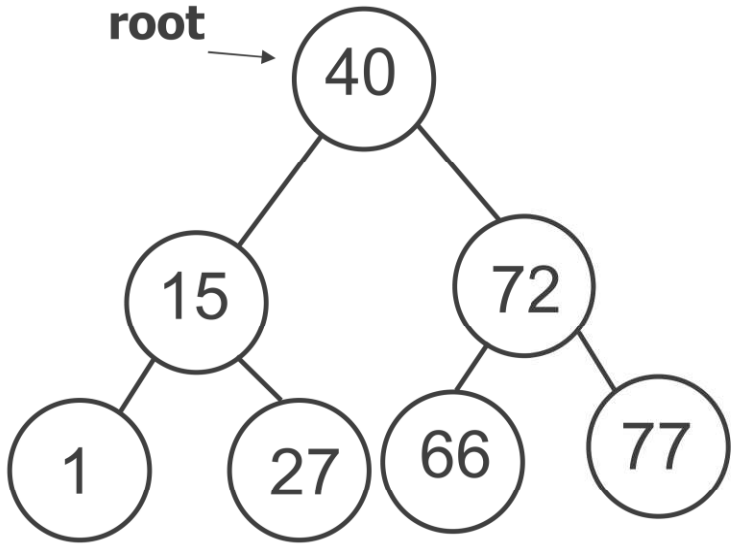
- **Binary Search Tree** (BST): a binary tree where each non-empty node *R* has the following properties:
  - ✓ elements of *R*'s left subtree contain data "less than" *R*'s data
  - ✓ elements of *R*'s right subtree contain data either equal to or "greater than" *R*'s data
  - ✓ *R*'s left and right subtrees are also binary search trees

Introduction



Introduction

- BSTs store their elements in sorted order, which is helpful for searching/sorting tasks

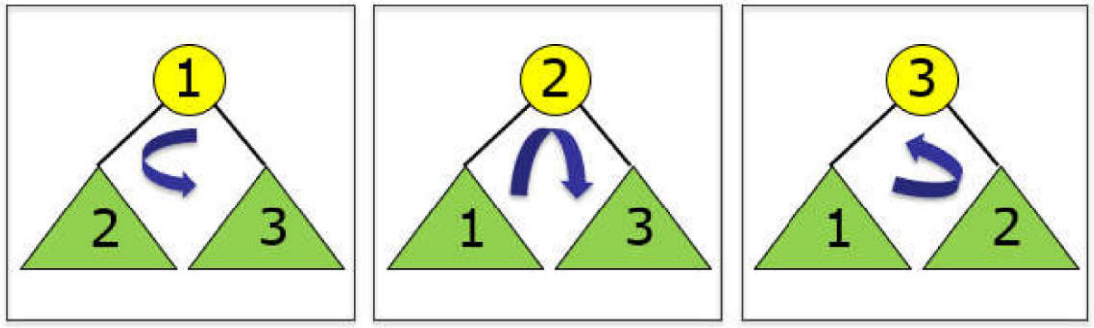


Binary Search Tree Operation

- Traversal
- Searching
- Insertion
- Deletion

Binary Search Tree Operation

- Traversal



Pre order In order Post order

Binary Search Tree Operation

- **Searching in the BST**
  - ✓Find a requested node
  - ✓Find the smallest node
  - ✓Find the largest node

Binary Search Tree Operation

- Searching in the BST: Find a requested node

  ✓implements the binary search based on comparison of the items in the tree

  ✓the items in the BST must be comparable

  ✓The search starts at the root. It probes down, comparing the values in each node with the target, till it finds the first item equal to the target. **Returns this item or null if there is none.**

Binary Search Tree Operation

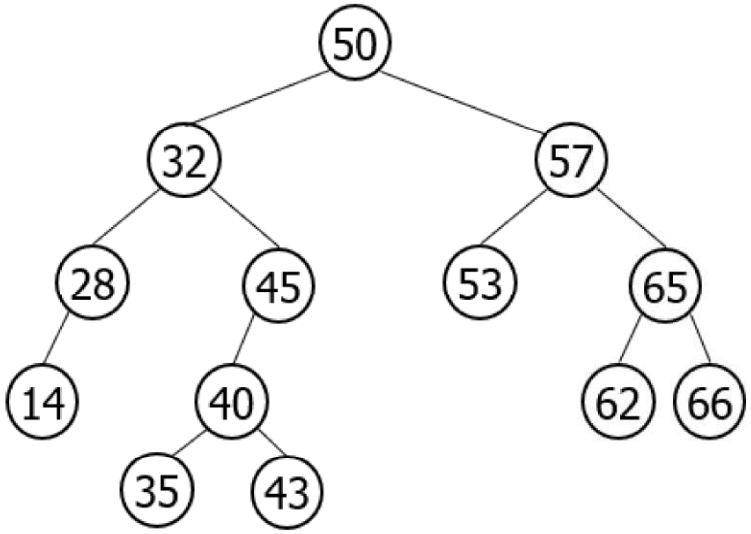
- Searching in the BST: Find a requested node

✓Pseudocode

```
if the tree is empty
        return NULL
else if  the item in the node equals the target
        return the node value
else if  the item in the node is greater than the target
        return the result of searching the left subtree
else if  the item in the node is smaller than the target
        return the result of searching the right subtree
```

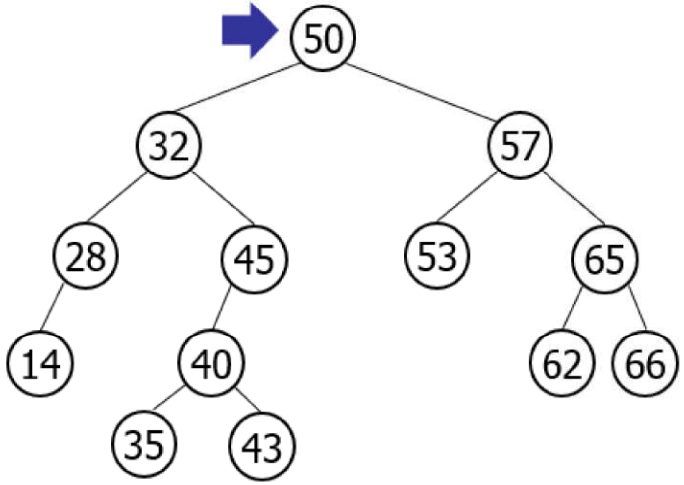
Binary Search Tree Operation

- Searching in the BST: Find a requested node
  - ✓Example: Searching for the value 40 from a given binary search tree



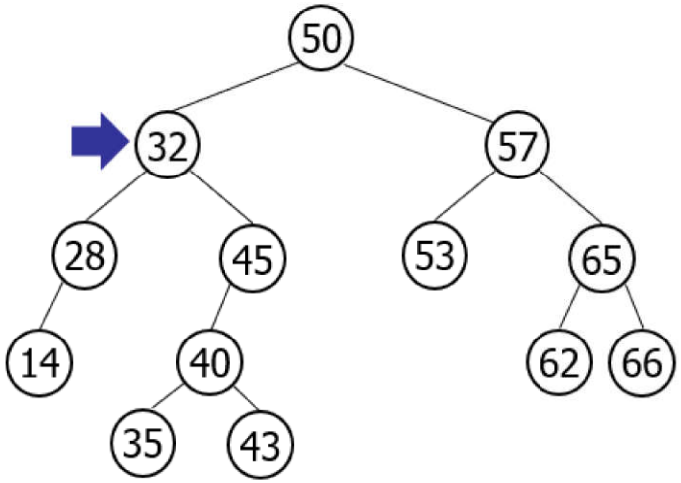
Binary Search Tree Operation

- Searching in the BST: Find a requested node
  - ✓Solution:



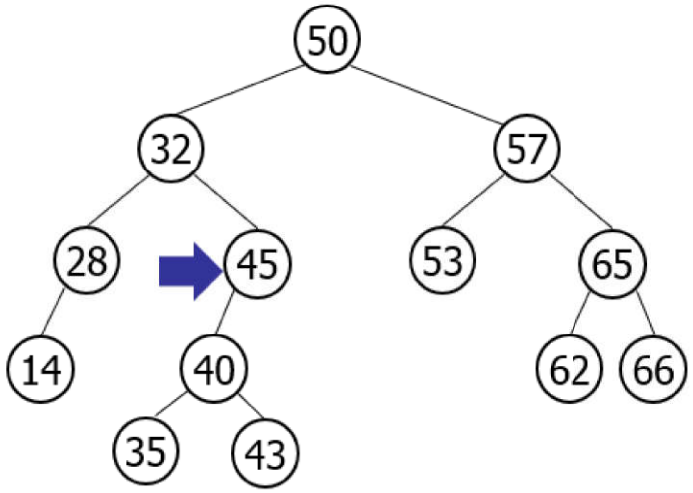
Binary Search Tree Operation

- Searching in the BST: Find a requested node
  - ✓Solution:



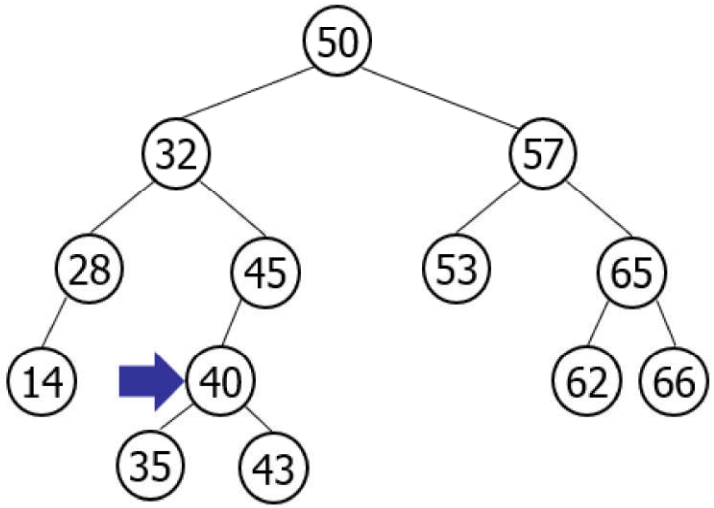
Binary Search Tree Operation

- Searching in the BST: Find a requested node
  - ✓Solution:



Binary Search Tree Operation

- Searching in the BST: Find a requested node
  - ✓Solution:



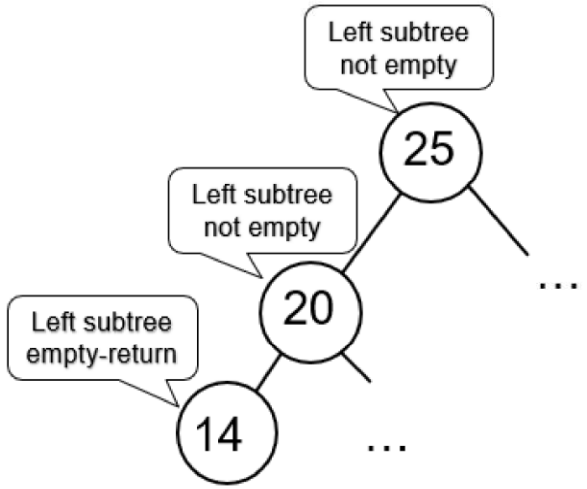
Binary Search Tree Operation

- Searching in the BST: Find the smallest node

```
Algorithm findSmallestBST (root)
    if (left subtree empty)
        return (root)
    end if
    Return findSmallestBST (left subtree)
end findSmallestBST
```

Binary Search Tree Operation

- Searching in the BST: Find the smallest node
  - ✓ Example:



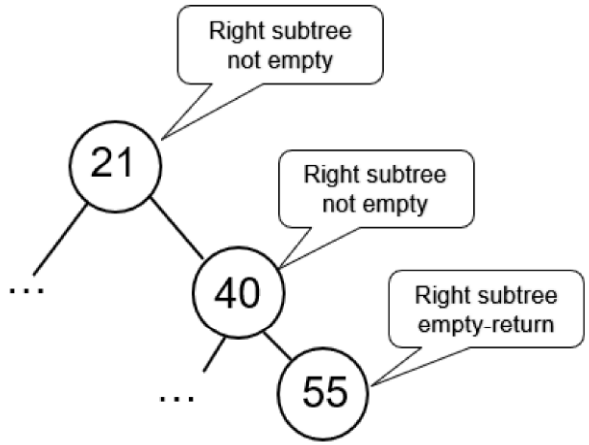
Binary Search Tree Operation

- Searching in the BST: Find the largest node

```
Algorithm findLargestBST (root)
   if (right subtree empty)
      return (root)
   end if
   Return findLargestBST (right subtree)
end findLargestBST
```

Binary Search Tree Operation

- Searching in the BST: Find the largest node
  - ✓ Example:



Binary Search Tree Operation

- **Insertion**:
  - ✓ To insert data all we need to do is follow the branches to an empty subtree and then insert the new node
  - ✓ In other words, all inserts take place at a leaf or at a leaflike node (a node that has only one null subtree)

Binary Search Tree Operation

- **Insertion**:
  - ✓ **Case 1**: The Tree is Empty

    Set the root to a new node containing the item
  - ✓ **Case 2**: The Tree is Not Empty

    Call a recursive helper method to insert the item

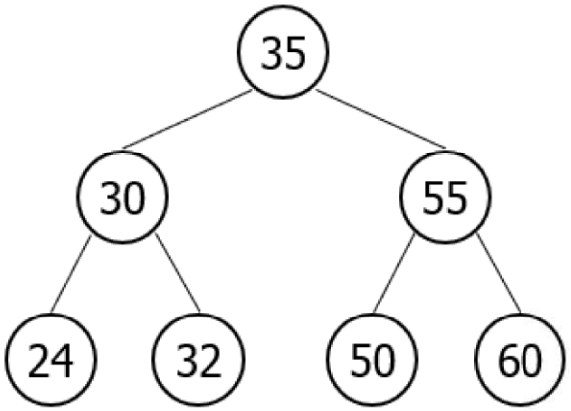
Binary Search Tree Operation

- Insertion:

```
Algorithm addBST (root, newNode)
  if (empty tree)
       set root to newNode
       return newNode
  end if
  If (newNode < root)
       return addBST (left subtree, newNode)
  else
       return addBST (right subtree, newNode)
  end if
end addBST
```

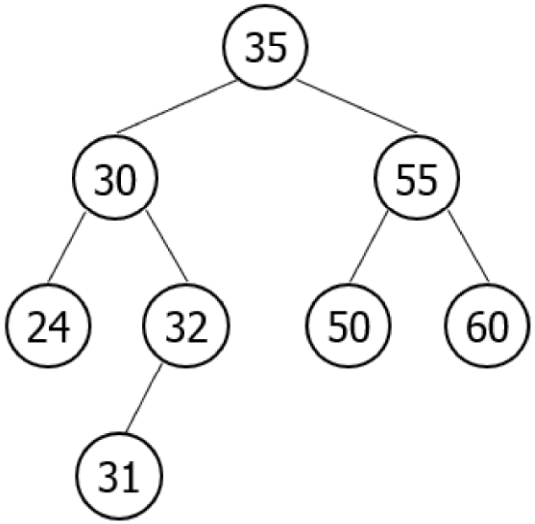
Binary Search Tree Operation

- Insertion:
  - ✓ Example: Insert 31 into a given BST



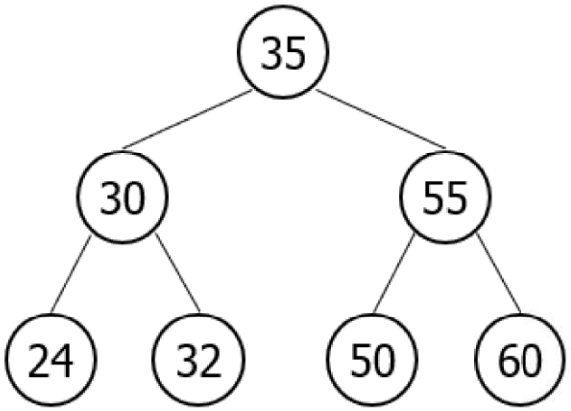
Binary Search Tree Operation

- Insertion:
  - ✓ Solution:



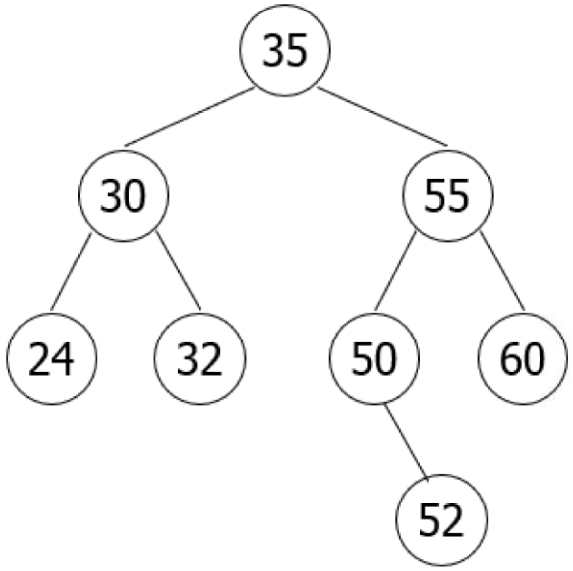
Binary Search Tree Operation

- Insertion:
  - ✓ Example: Insert 52 into a given BST



Binary Search Tree Operation

- Insertion:
  - ✓ Solution:



Binary Search Tree Operation

- **Deletion**:
  - ✓ removes a specified item from the BST and adjusts the tree
  - ✓uses a binary search to locate the target item:
    - ➢starting at the root it probes down the tree till it finds the target or reaches a leaf node (target not in the tree)
  - ✓removal of a node must not leave a ‘gap’ in the tree

Binary Search Tree Operation

- **Deletion**:
  - ✓ **Case 1**: if the node has two empty subtrees
    - replace the link in the parent with null
  - ✓ **Case 2**: if the node has a left and a right subtree

    2.1
    - replace the node's value with the largest value in the left sub-tree
    - delete the largest value in the left sub-tree

    2.2
    - replace the node's value with the smallest value in the right sub-tree
    - delete the smallest value in the right sub-tree

Binary Search Tree Operation

- **Deletion**:
  - ✓ **Case 3**: if the node has no left child

    - link the parent of the node to the right (non-empty) subtree
  - ✓ **Case 4**: if the node has no right child

    - link the parent of the target to the left (non-empty) subtree

Binary Search Tree Operation

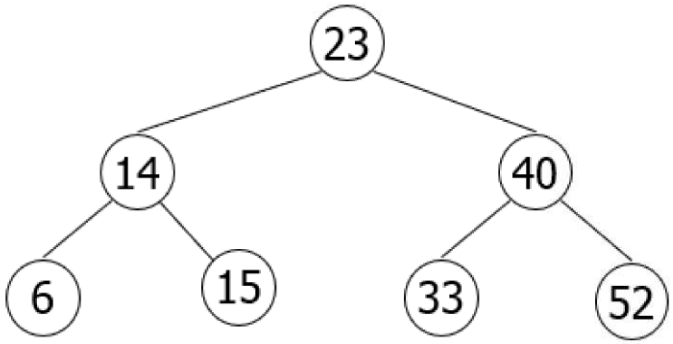
- Deletion:

Deletion Method (key)

```
1  if the tree is empty
       return false
2  Attempt to locate the node containing the target using the binary search algorithm
       if the target is not found
          return false
       else
          the target is found, so remove its node
```

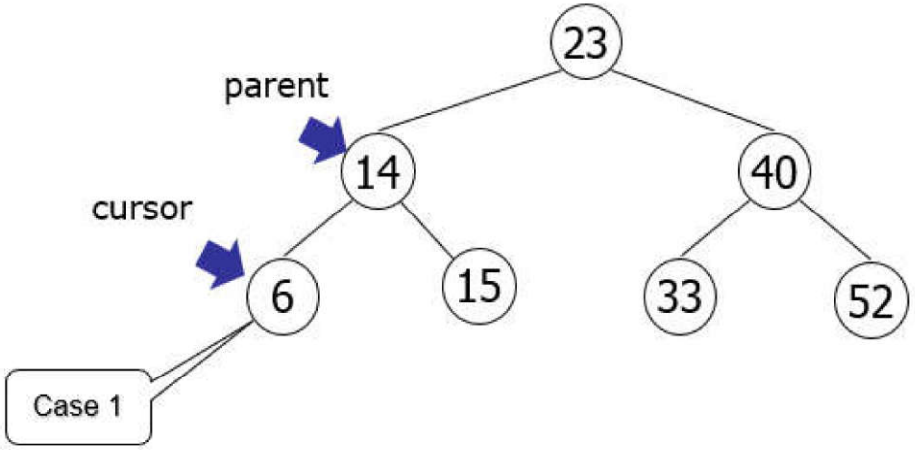
Binary Search Tree Operation

- Deletion:
  - ✓ Example: delete 6 from the given BST



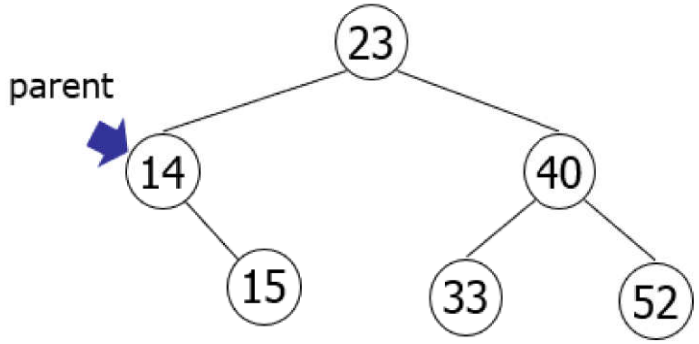
Binary Search Tree Operation

- Deletion:
  - ✓ Solution:



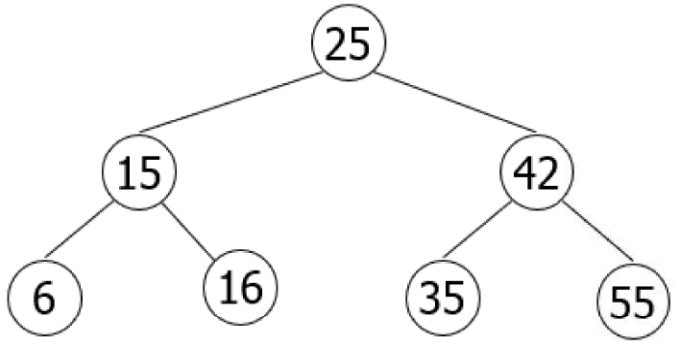
Binary Search Tree Operation

- Deletion:
  - ✓ Solution:



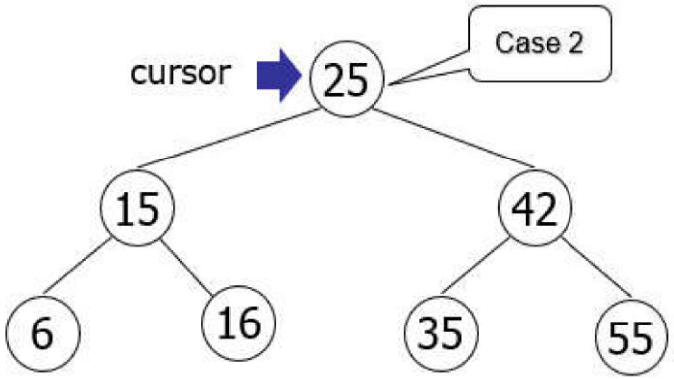
Binary Search Tree Operation

- Deletion:
  - ✓ Example: delete 25 from the given BST



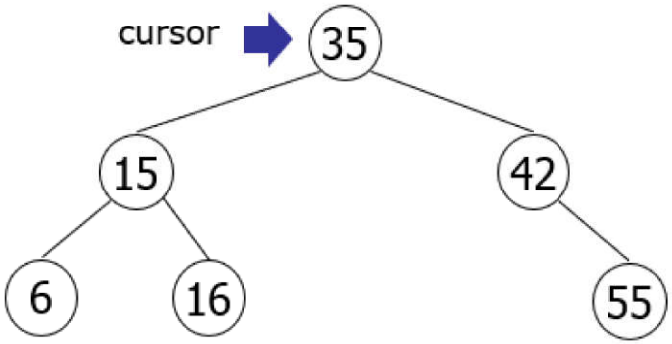
Binary Search Tree Operation

- Deletion:
  - ✓ Solution:



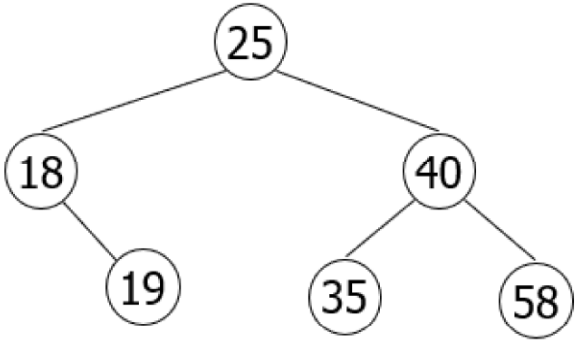
Binary Search Tree Operation

- Deletion:
  - ✓ Solution:



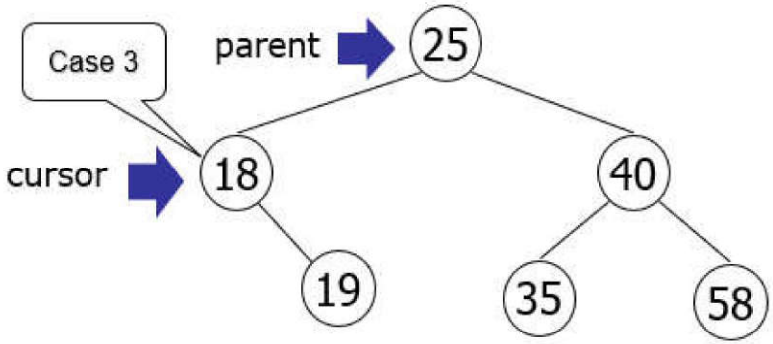
Binary Search Tree Operation

- Deletion:
  - ✓ Example: delete 18 from the given BST



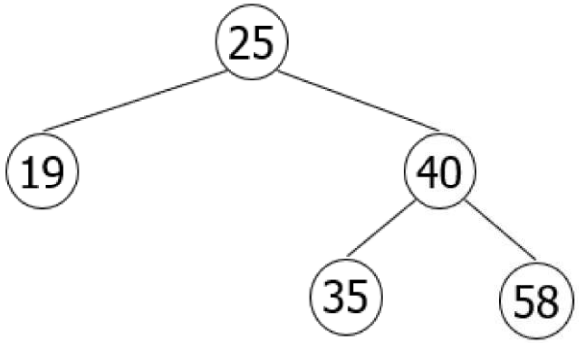
Binary Search Tree Operation

- Deletion:
  - ✓ Solution:



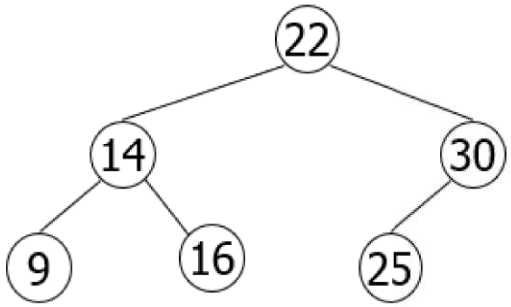
Binary Search Tree Operation

- Deletion:
  - ✓ Solution:



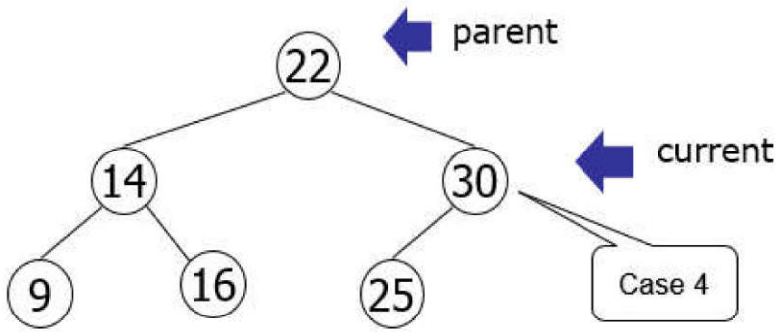
Binary Search Tree Operation

- Deletion:
  - ✓ Example: delete 30 from the given BST



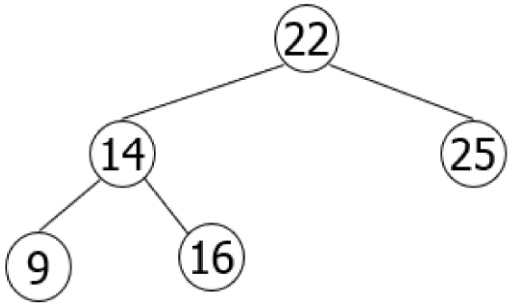
Binary Search Tree Operation

- Deletion:
  - ✓ Solution:



Binary Search Tree Operation

- Deletion:
  - ✓ Solution:



Class Activity

Q & A

Formative Assessment