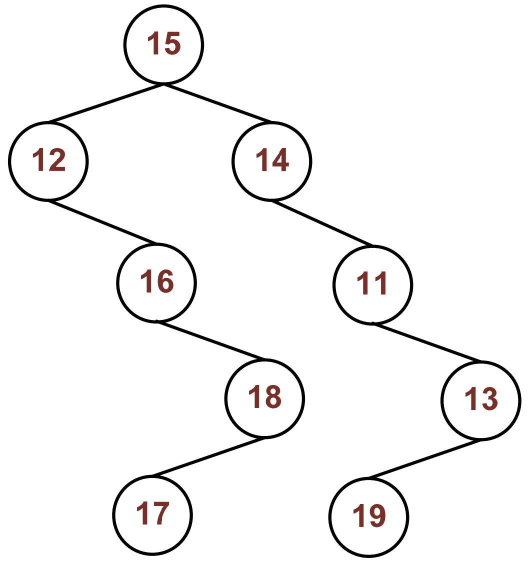
Review Activity 14 Solutions

Tree Construction and Heaps: Additional Practice

1) A binary tree was processed using pre-order and in-order traversals. For pre-order, the output derived is `[15, 12, 16, 18, 17, 14, 11, 13, 19]`. For in-order, the output derived is `[12, 16, 17, 18, 15, 14, 11, 19, 13]`. Draw a binary tree that complies with the traversals above. Show how you have derived the tree.

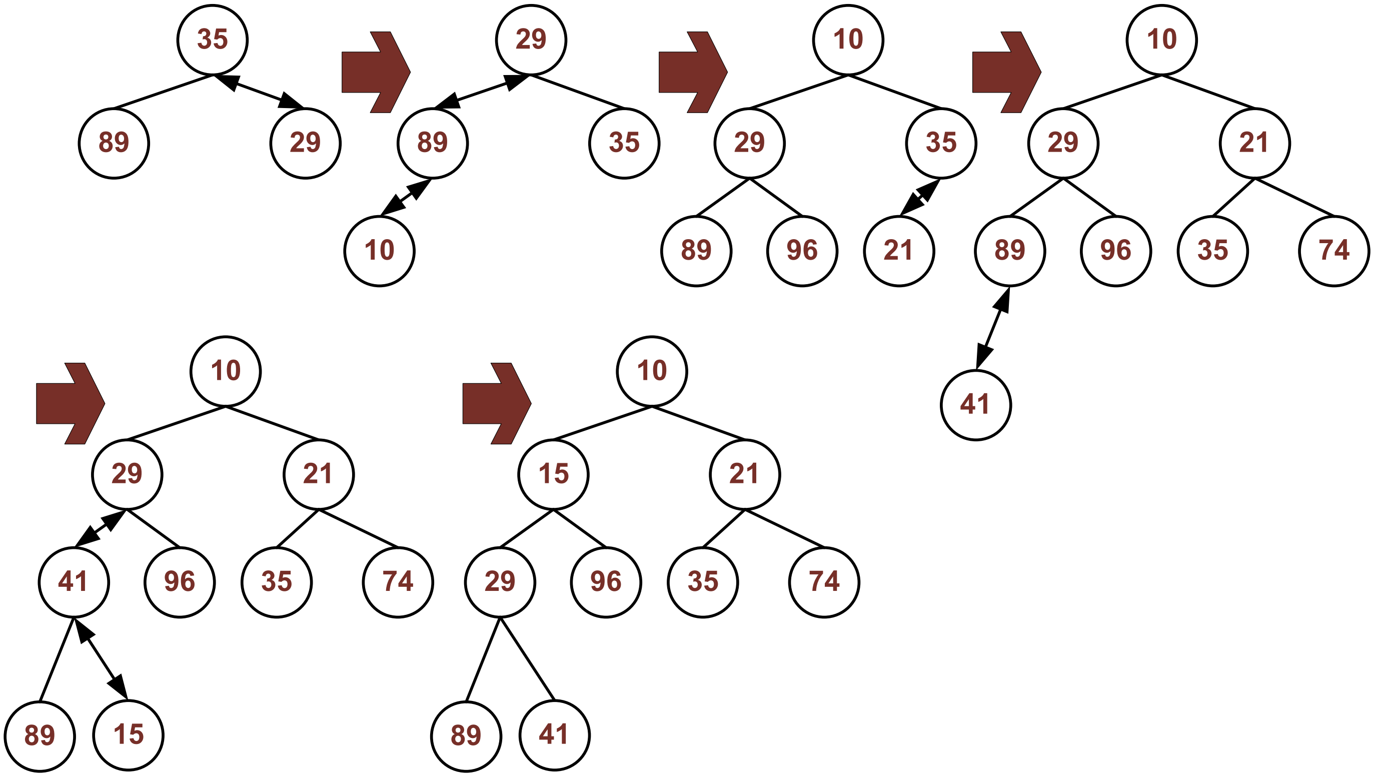


Use pre-order to discover the roots of each subtree, starting with 15.

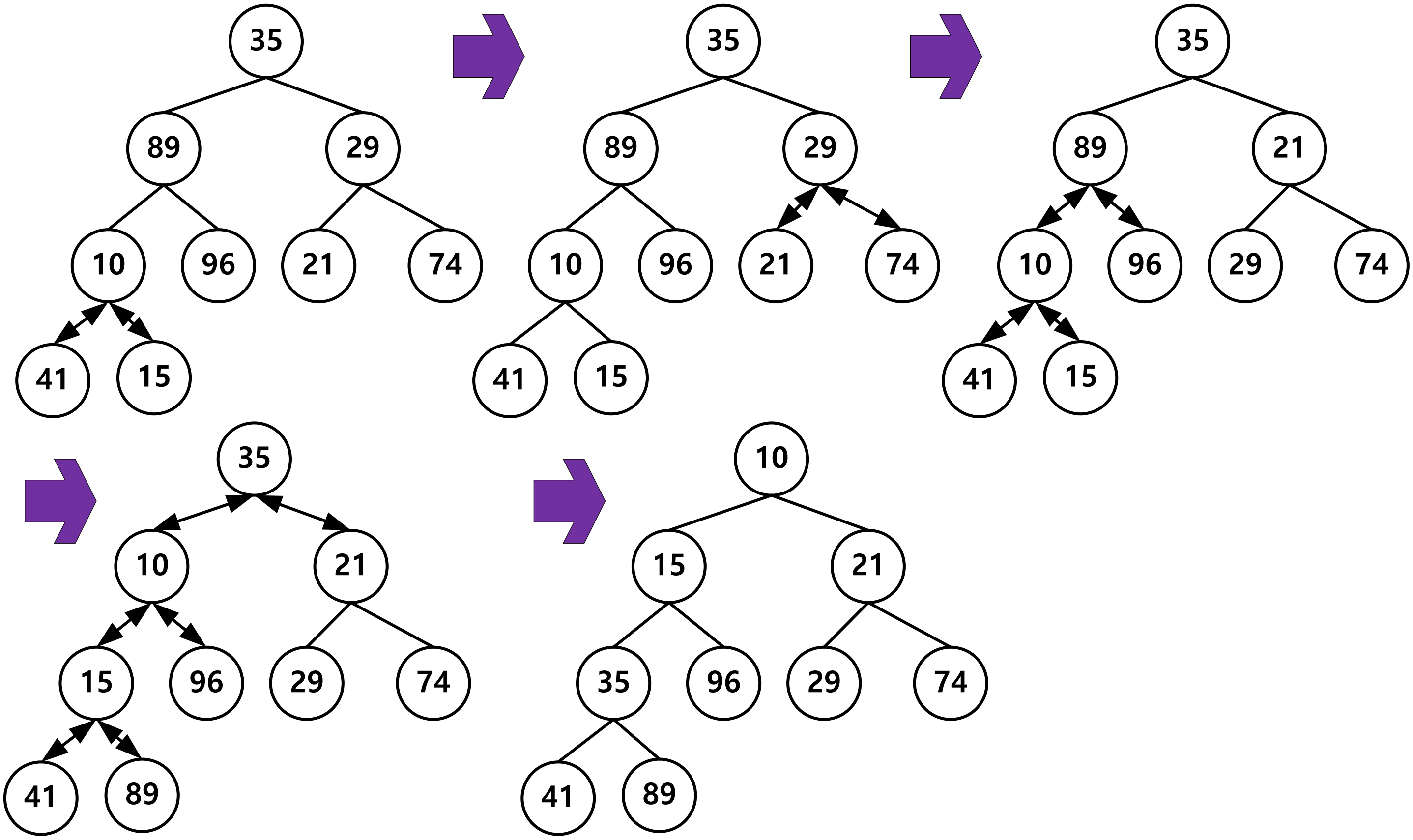
Use in-order to discover the left and right subtrees, starting with [12, 16, 17, 18] and [14, 11, 19, 13].

Additional/alternative explanation:
The in-order traversal indicates the position, left or right, of each subroot in reference to its parent. For instance, 17 is left of 18 based on its position in in-order traversal.

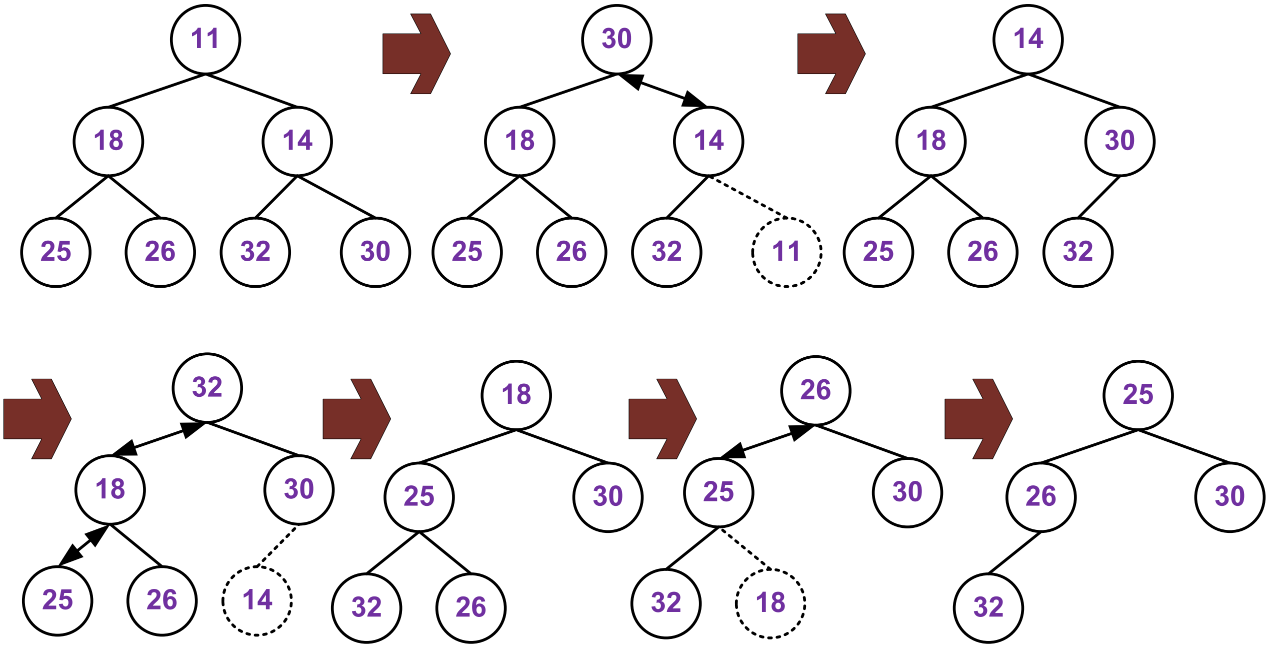
2) Insert the following nodes one at a time into an empty min-heap (minimum element is always at the top): `35 89 29 10 96 21 74 41 15`. Show important steps used in deriving your solution.



3) Insert the following nodes into an empty min-heap using Heapify() rather than Insert() method: 35 89 29 10 96 21 74 41 15. That is, insert all the nodes at once into a complete tree, and then apply heapification to turn it into appropriate min-heap. Show all important steps.



4) Run the remove function three times on the following min-heap (minimum element is always at the top). That is, remove three elements from the heap. Show steps used in deriving your solution.


11
18 14
25 26 32 30
30
18 14
25 26 32 11
14
18 30
25 26 32
32
18 30
25 26 14
18
25 30
32 26
26
25 30
32 18
25
26 30
32