CS062

DATA STRUCTURES AND ADVANCED PROGRAMMING

18–19: Balanced Search Trees

Alexandra Papoutsaki

Lecture 18-19: Balanced Search Trees

- 2-3-4 trees
- Search
- Insertion
- Construction
- Performance
- Red-Black Trees

Visualization of insertion into a binary search tree

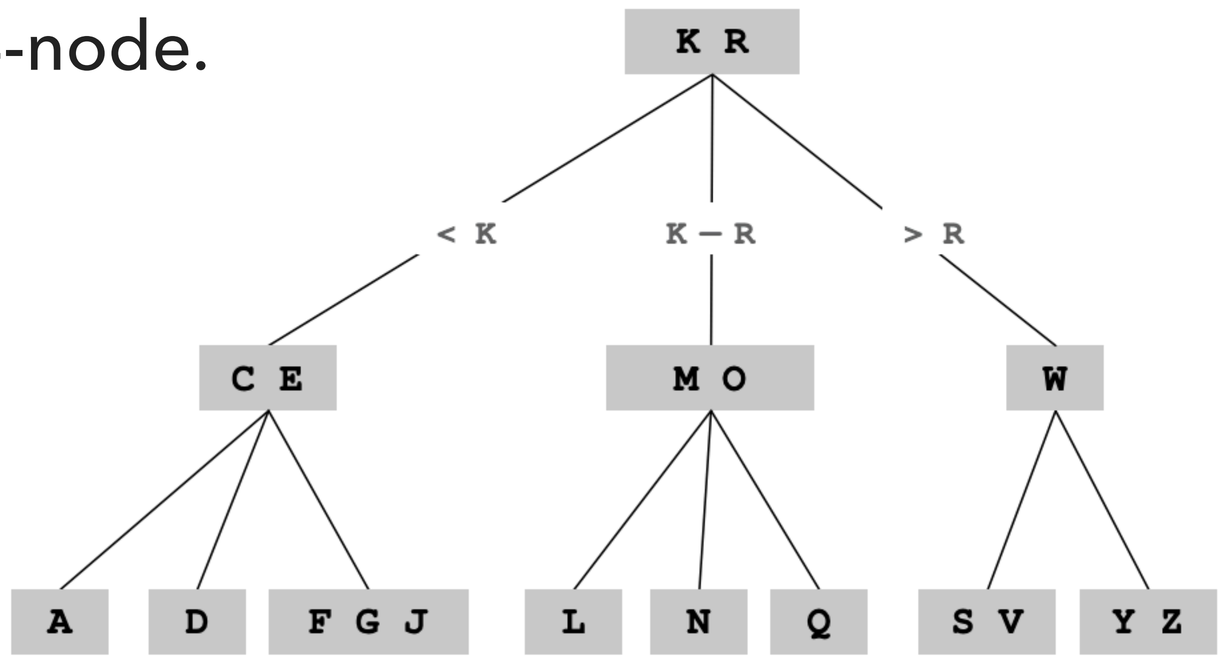
- 255 insertions in random order.

Order of growth for dictionary operations

	Worst case			Average case		
	Search	**Insert**	**Delete**	**Search**	**Insert**	**Delete**
BST	n	n	n	$\log n$	$\log n$	$\sqrt{n}$
Goal	$\log n$	$\log n$	$\log n$	$\log n$	$\log n$	$\log n$

2-3-4 tree

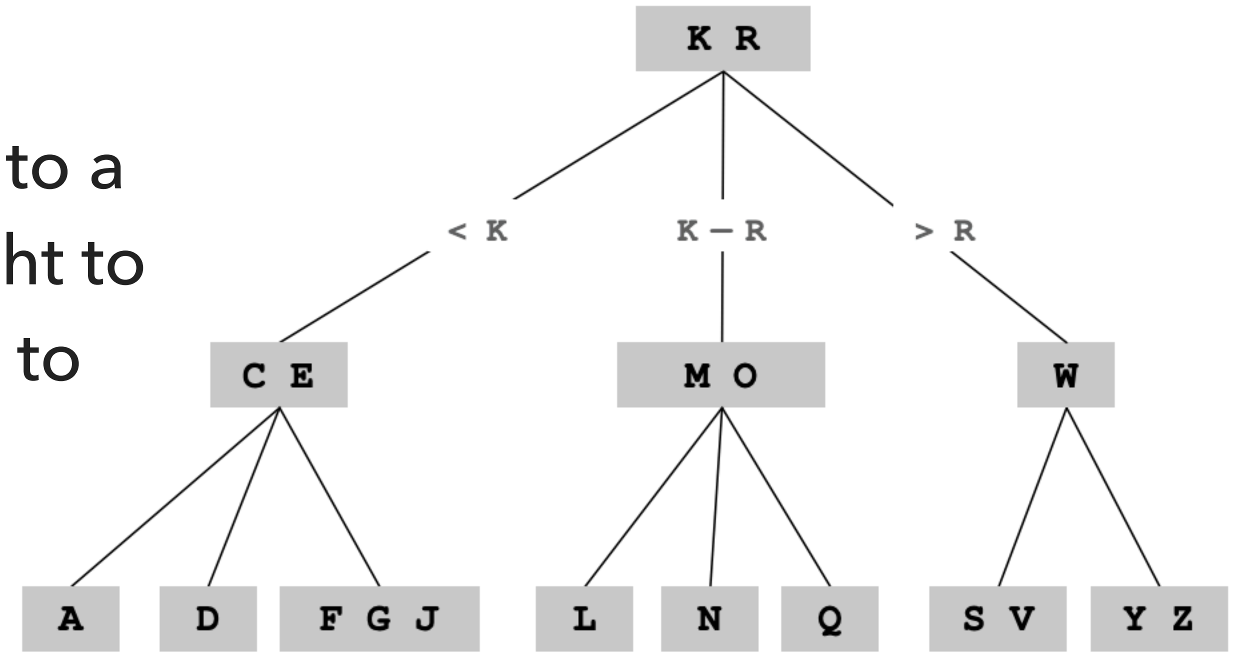
- Definition: A 2-3-4 search tree is either empty or consists of three types of nodes: 2-node, a 3-node, or a 4-node.
    - 2-node: one key, two children
    - 3-node: two keys, three children
    - 4-node: three keys, four children



- Balanced 2-3-4 tree: A 2-3-4 search tree with with all paths from root to a null link has the same length, that is all leaves have the same depth.
    - From now on, 2-3-4 trees are assumed to be balanced.

2-node

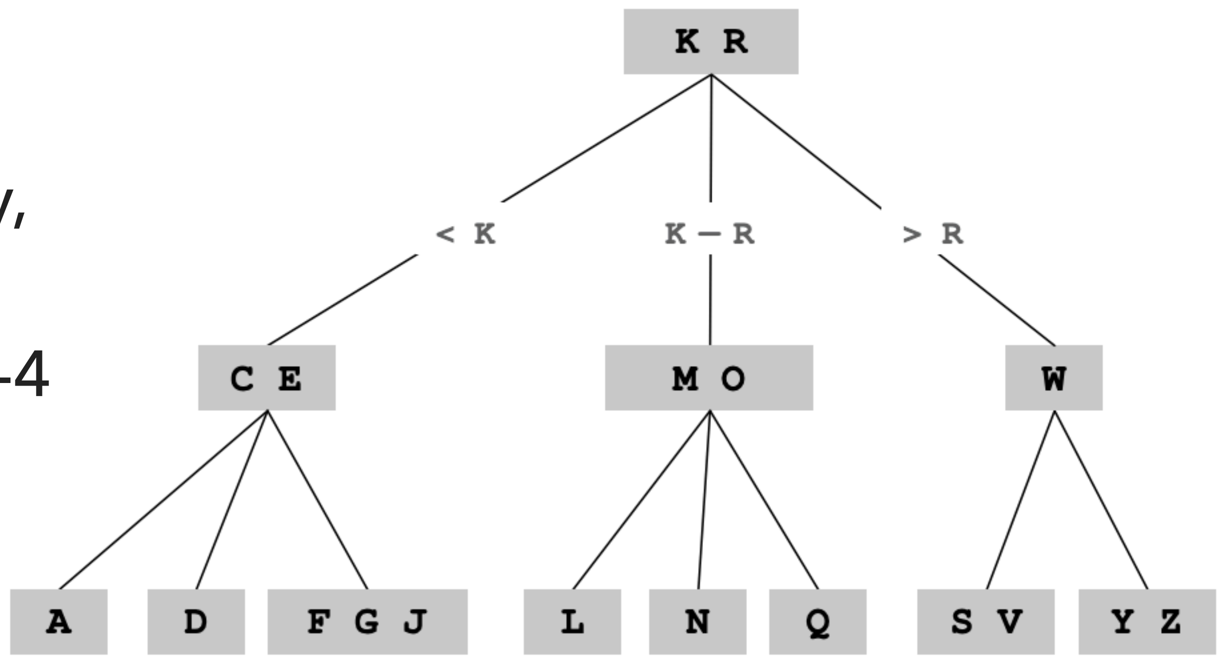
- Definition: a node with one key (and associated value) and two links, a left to a 2-3-4 tree with smaller keys, and a right to a 2-3-4 tree with larger keys (similarly to standard BSTs).



- E.g., the node that holds W is a 2-node. It has two children. Left child is a 2-3-4 tree with keys smaller than W and right child is a 2-3-4 tree with keys larger than W.

3-node

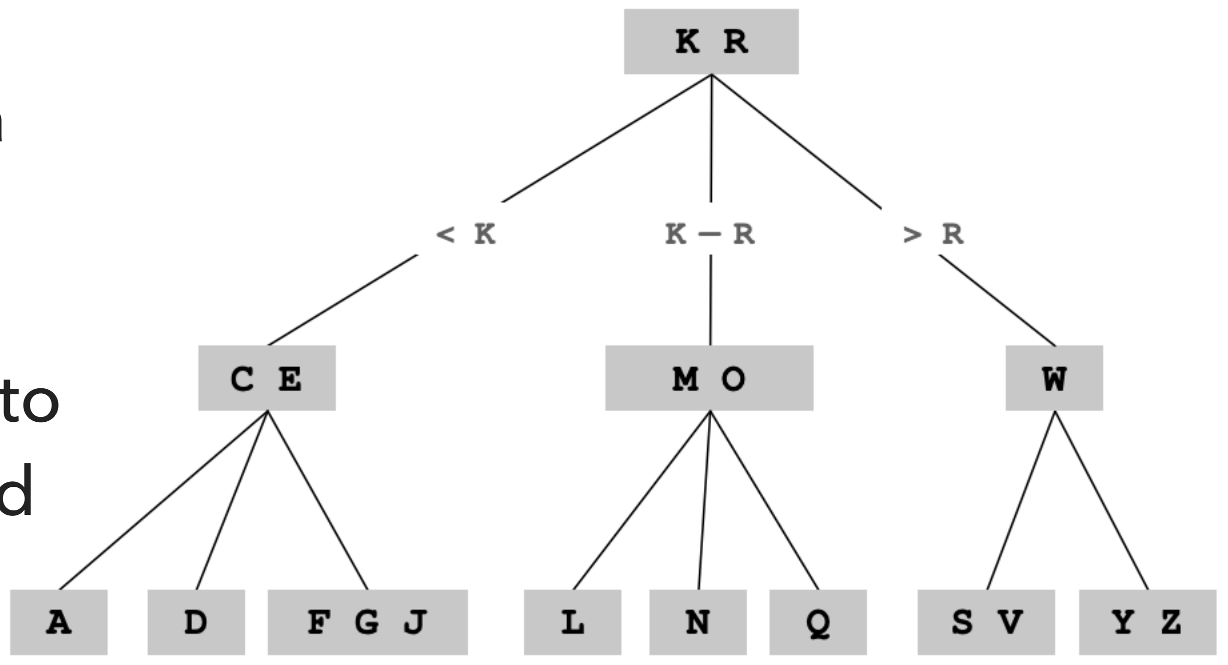
- Definition: a node with two keys (and associated values) and three links, a left to a 2-3-4 tree with smaller keys than the first key, a middle to a 2-3-4 tree with keys between the first and second key, and a right to a 2-3-4 tree with larger keys than the second key.



- E.g., the node that holds K and R is a 3-node. It has three children. Left child is a 2-3-4 tree with keys smaller than K, middle is a 2-3-4 tree with keys between K and and R, and right child is a 2-3-4 tree with keys larger than R.

4-node

- Definition: a node with three keys (and associated values) and four links, first to a 2-3-4 tree with smaller keys than the first key, a second to a 2-3-4 tree with keys between the first and second key, a third to a 2-3-4 tree with keys between the second and third key, and a fourth to a 2-3-4 tree with larger keys than the fourth key.



- E.g., the node that holds F, G, and J is a 4-node.

Lecture 18-19: Balanced Search Trees

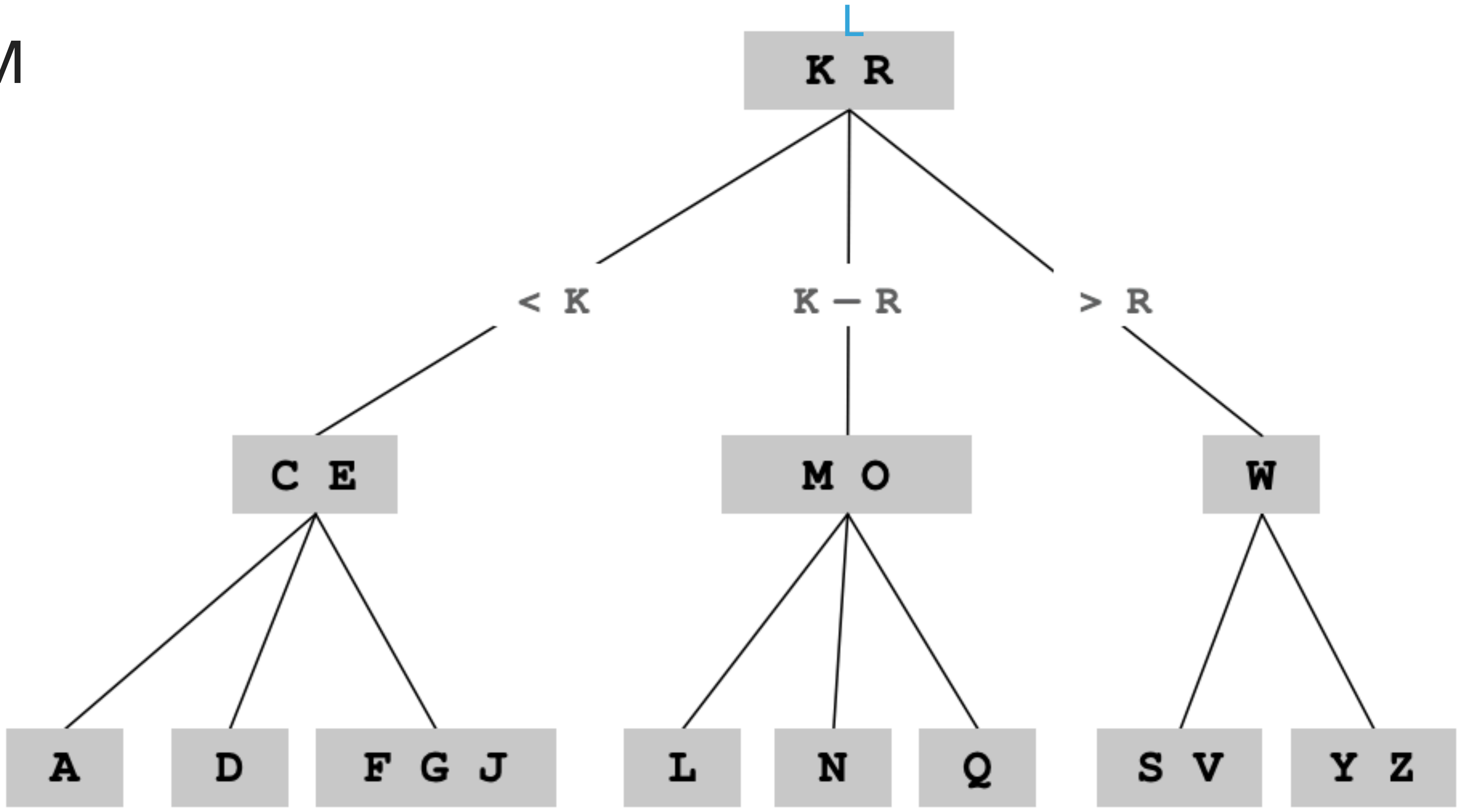
- 2-3-4 trees
- Search
- Insertion
- Construction
- Performance
- Red-Black Trees

How to search for a key

- Compare search key against (every) key in node.
- Find interval containing search key
- Follow associated link, recursively.

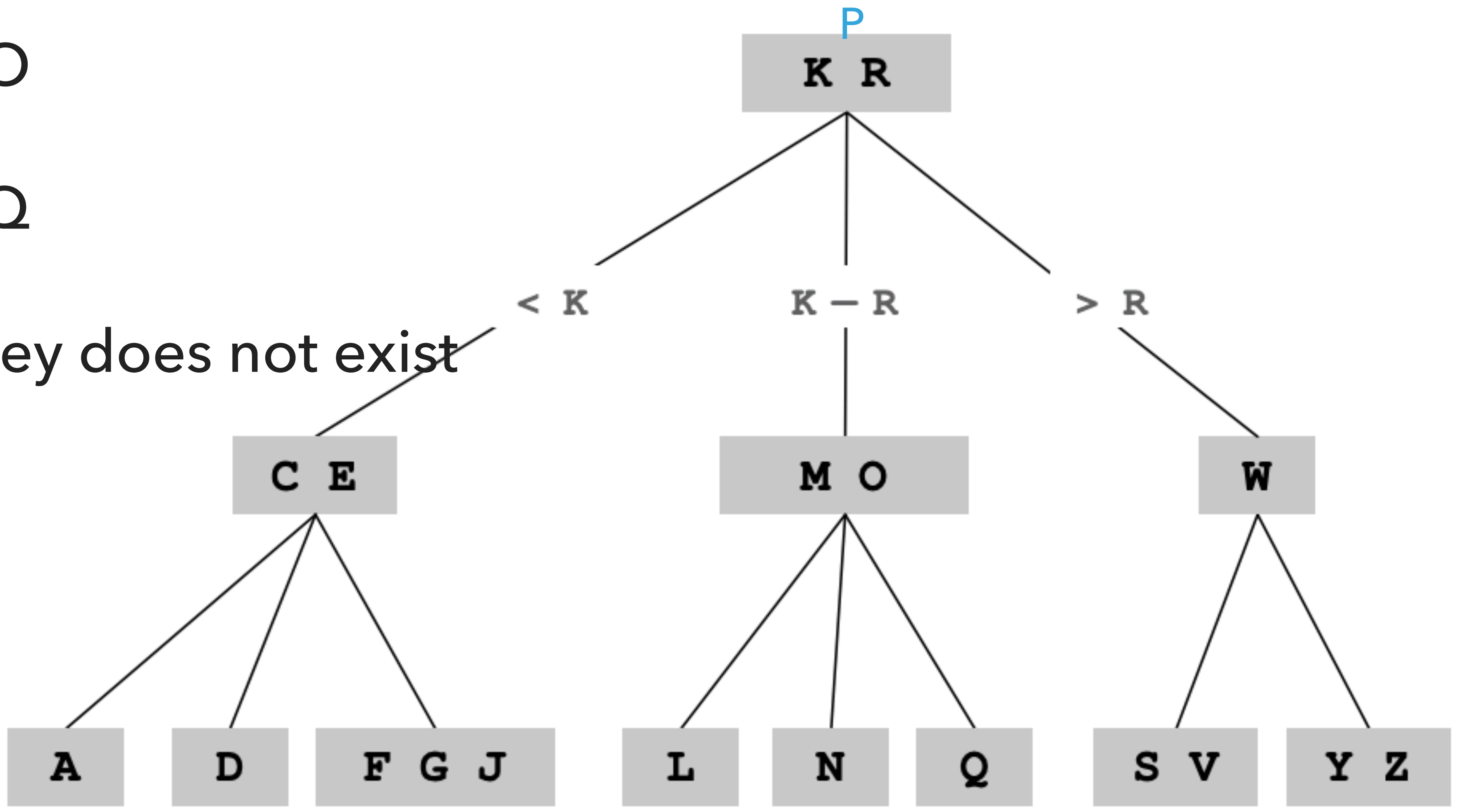
Search Hit

- Example: Search for key L
- L is greater than K and smaller than R
- L is smaller than M
- L is found!



Search Miss

- Example: Search for key P
- P is greater than K and smaller than R
- P is greater than O
- P is smaller than Q
- Reached null so key does not exist

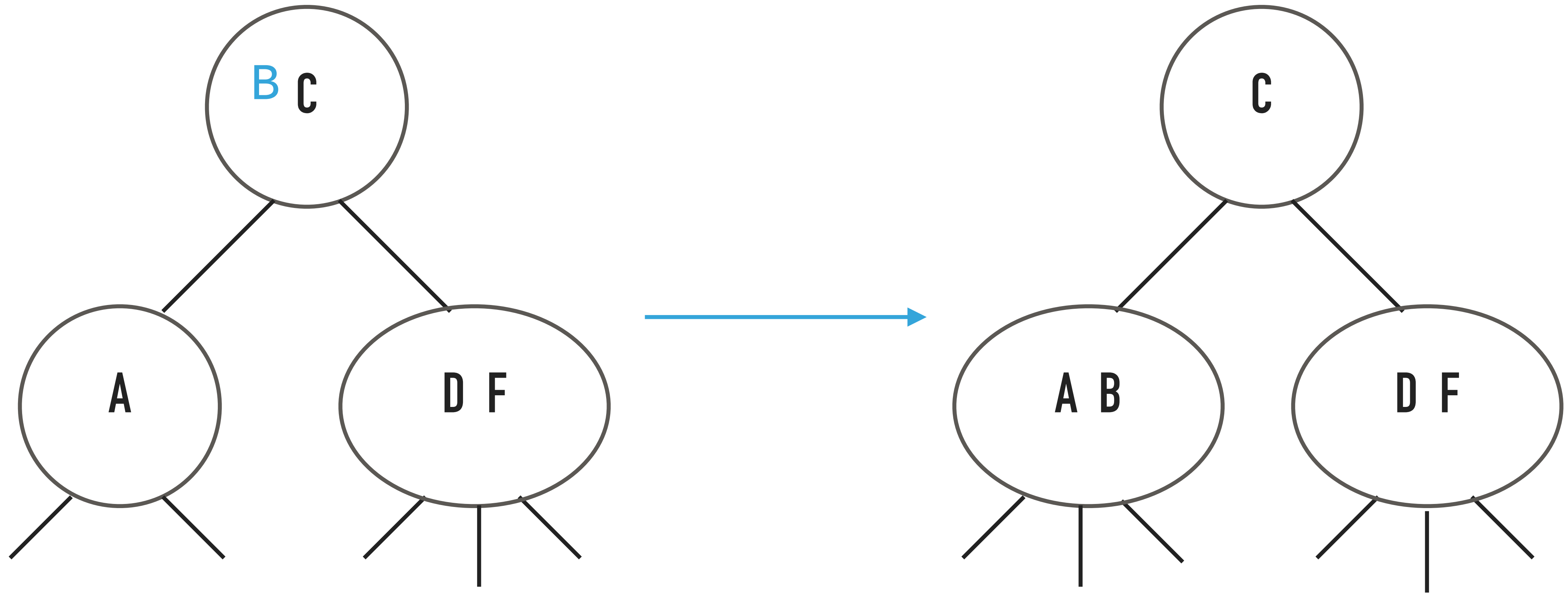


Lecture 18-19: Balanced Search Trees

- 2-3-4 trees
- Search
- Insertion
- Construction
- Performance
- Red-Black Trees

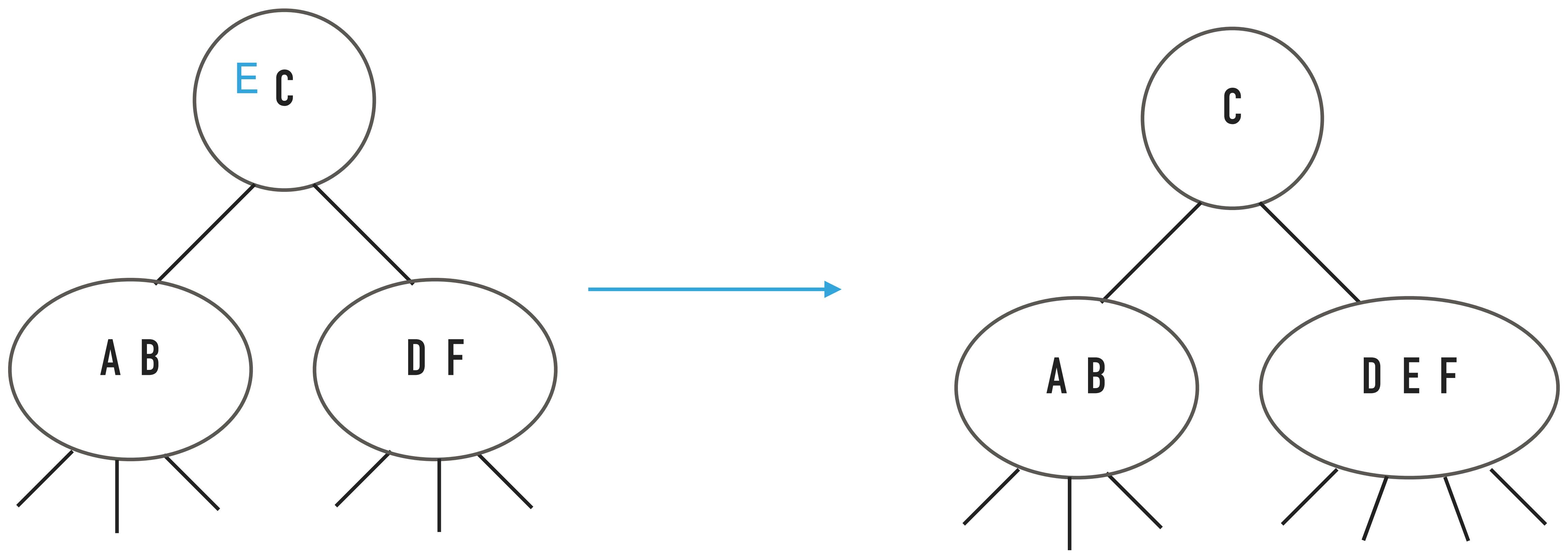
How to insert into a 2-node

- Search for key to bottom and add new key to 2-node to create a 3-node.
- E.g., insert B in the following 2-3-4 tree.



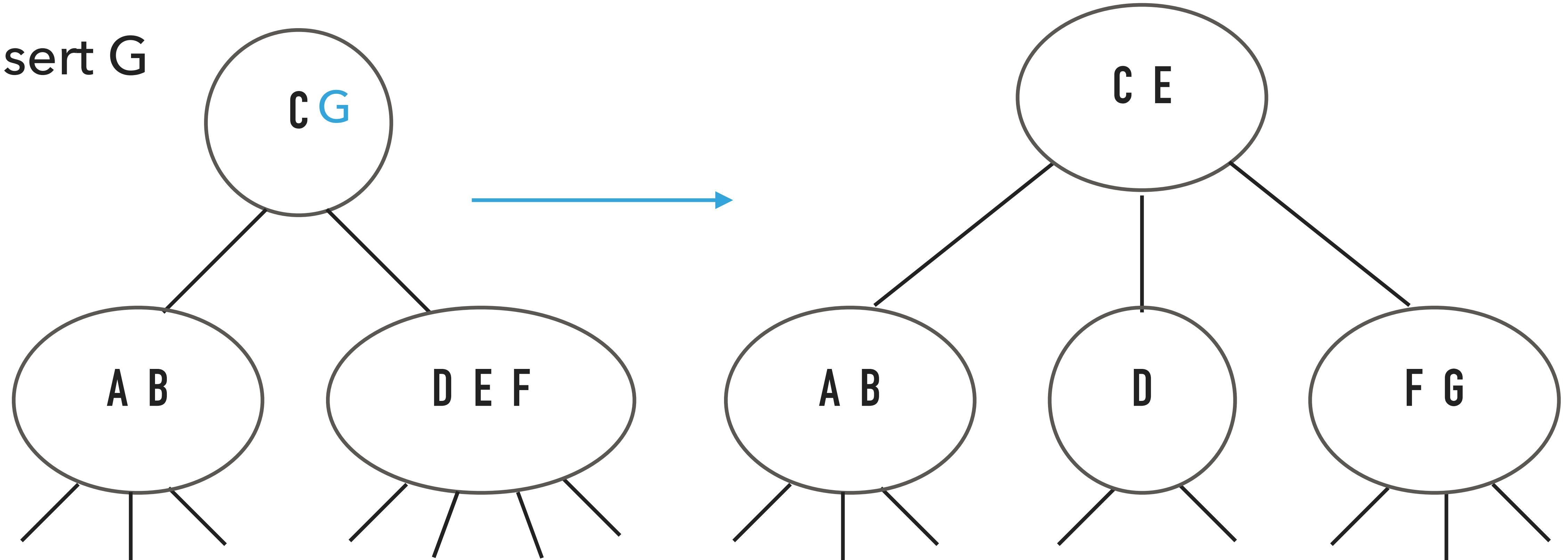
How to insert into a 3-node

- Search for key to bottom and add new key to 3-node to create a 4-node.
- E.g., insert E in the following 2-3-4 tree.



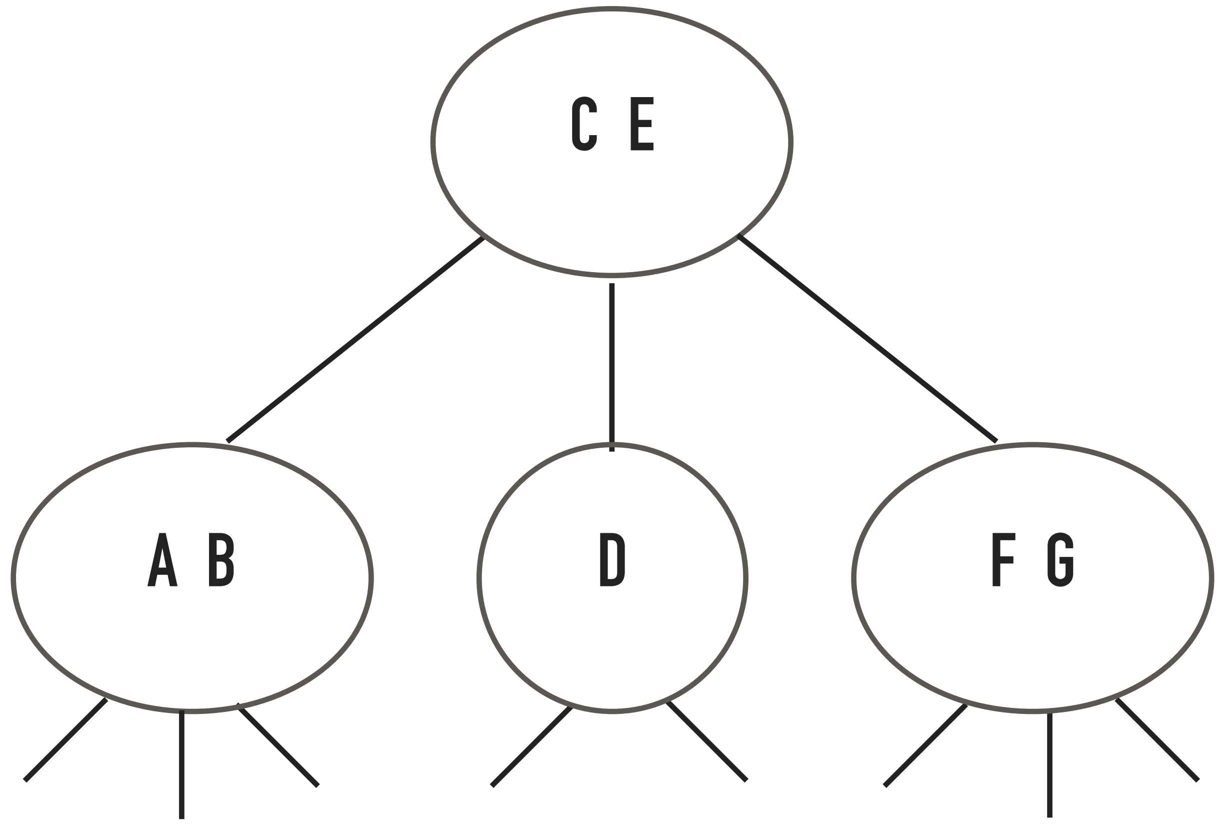
How to insert into a 4-node

- Search for key to bottom. A node can only have up to three keys/ four children so what do we do?
- We split the node by moving the middle key to its parent node and then add the key to the appropriate child. If the parent node has more than three keys, we split it again.
- Ex. insert G

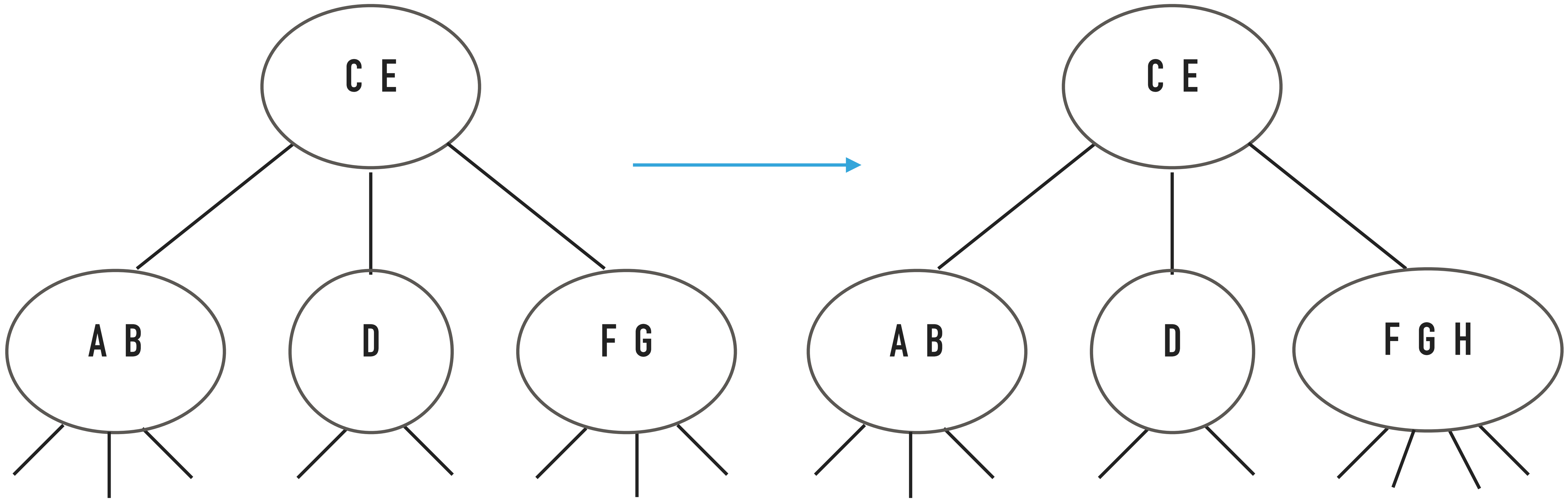


PRACTICE TIME

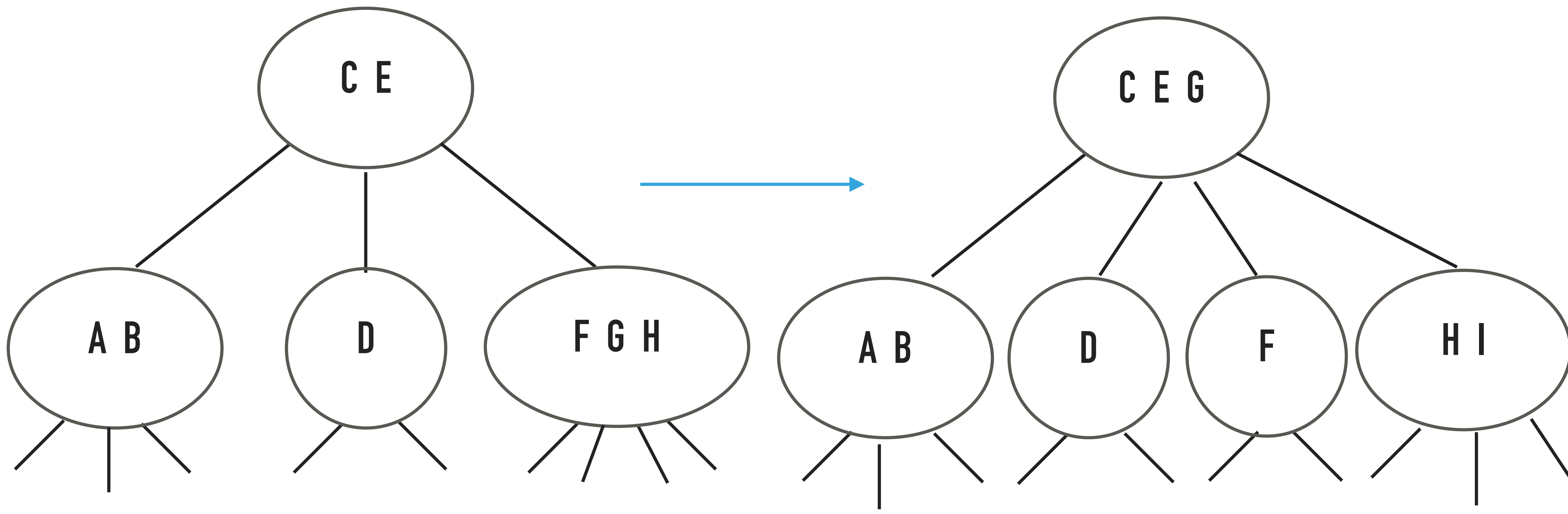
- Given the following 2-3-4 tree, insert keys H, I, J, K.



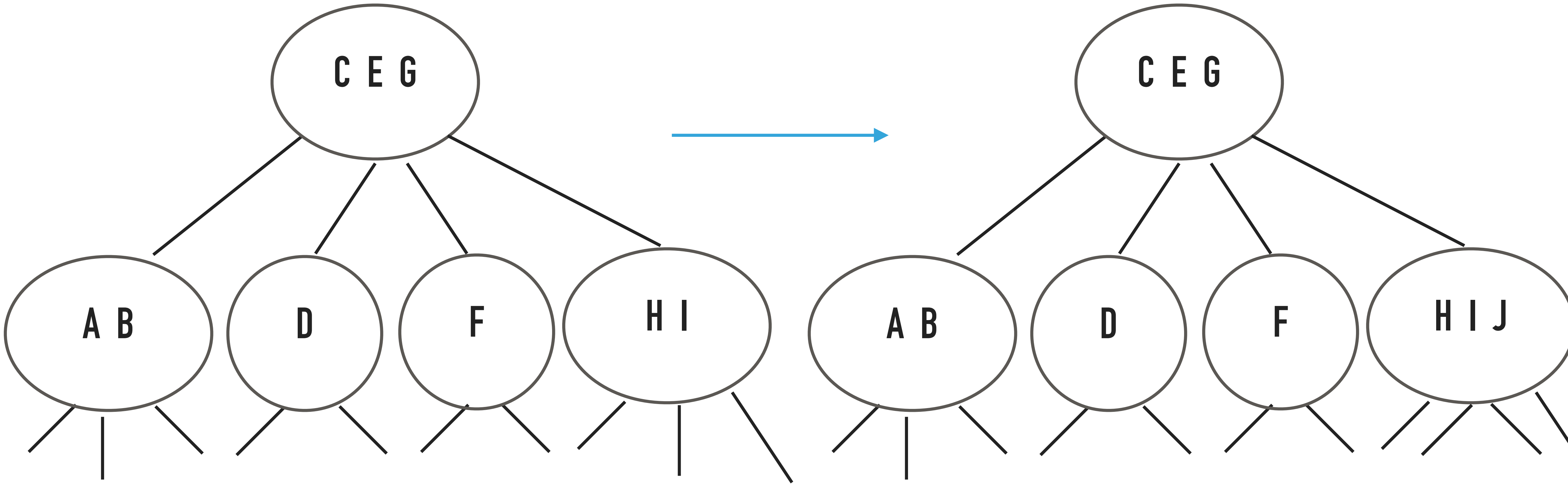
ANSWER - Insert H



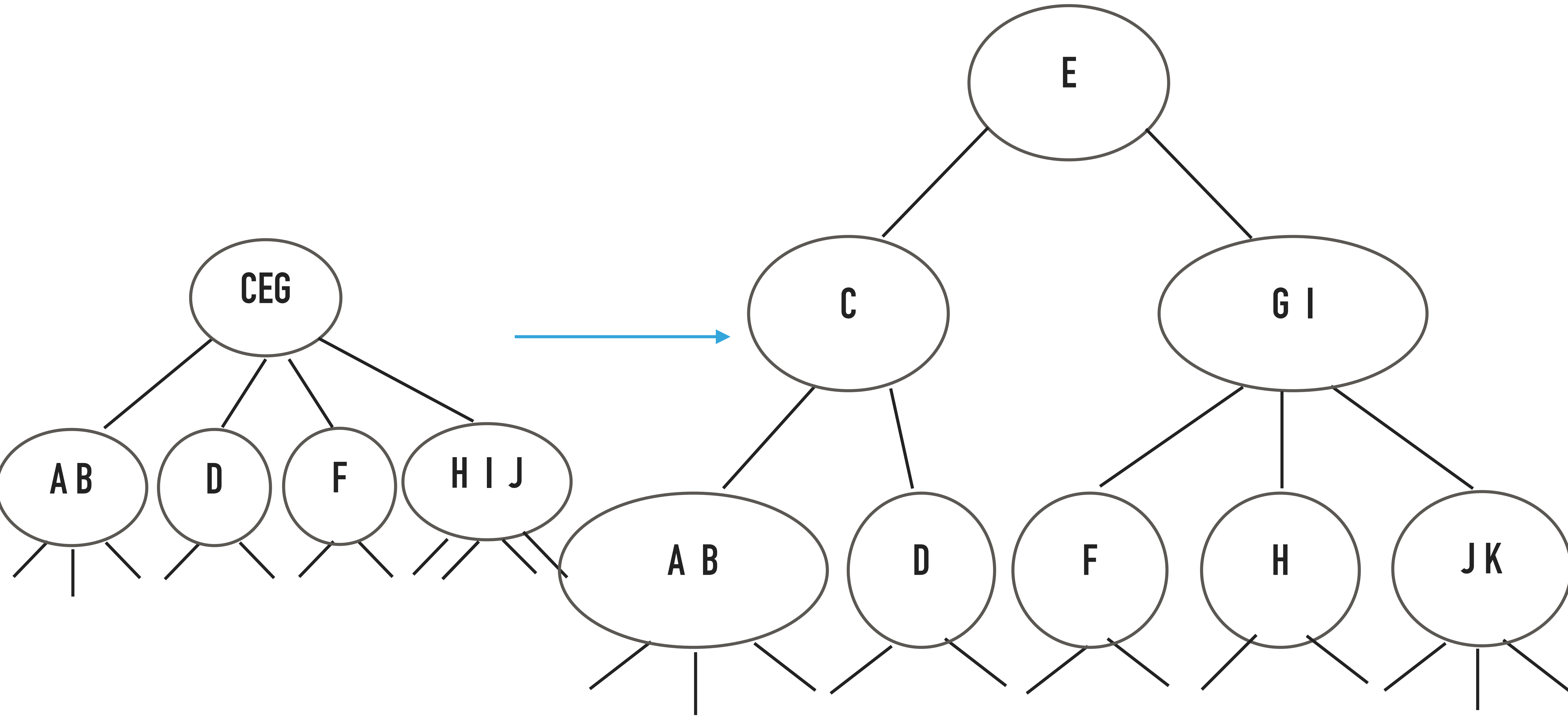
ANSWER - Insert I



ANSWER - Insert J



ANSWER - Insert K



Lecture 18-19: Balanced Search Trees

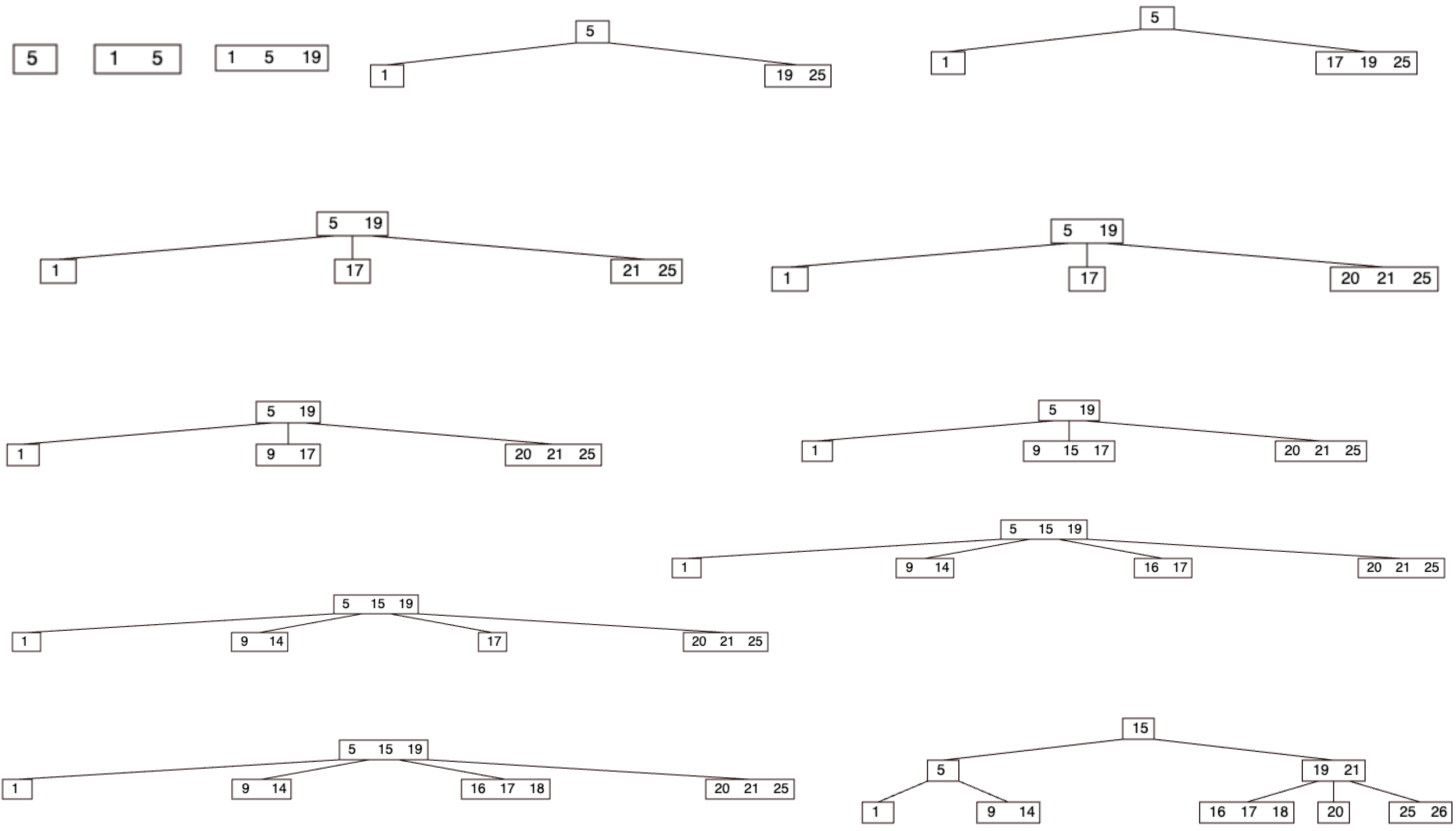
- 2-3-4 trees
- Search
- Insertion
- Construction
- Performance
- Red-Black Trees

Practice Time - Worksheet

- Draw the 2-3-4 tree that results when you insert the keys:
  5, 1, 19, 25, 17, 21, 20, 9, 15, 14, 16, 18, 26 in that order in an initially empty tree.

ANSWER

- 5, 1, 19, 25, 17, 21, 20, 9, 15, 14, 16, 18, 26



Lecture 18-19: Balanced Search Trees

- 2-3-4 trees
- Search
- Insertion
- Construction
- Performance
- Red-Black Trees

Height of 2-3-4 trees

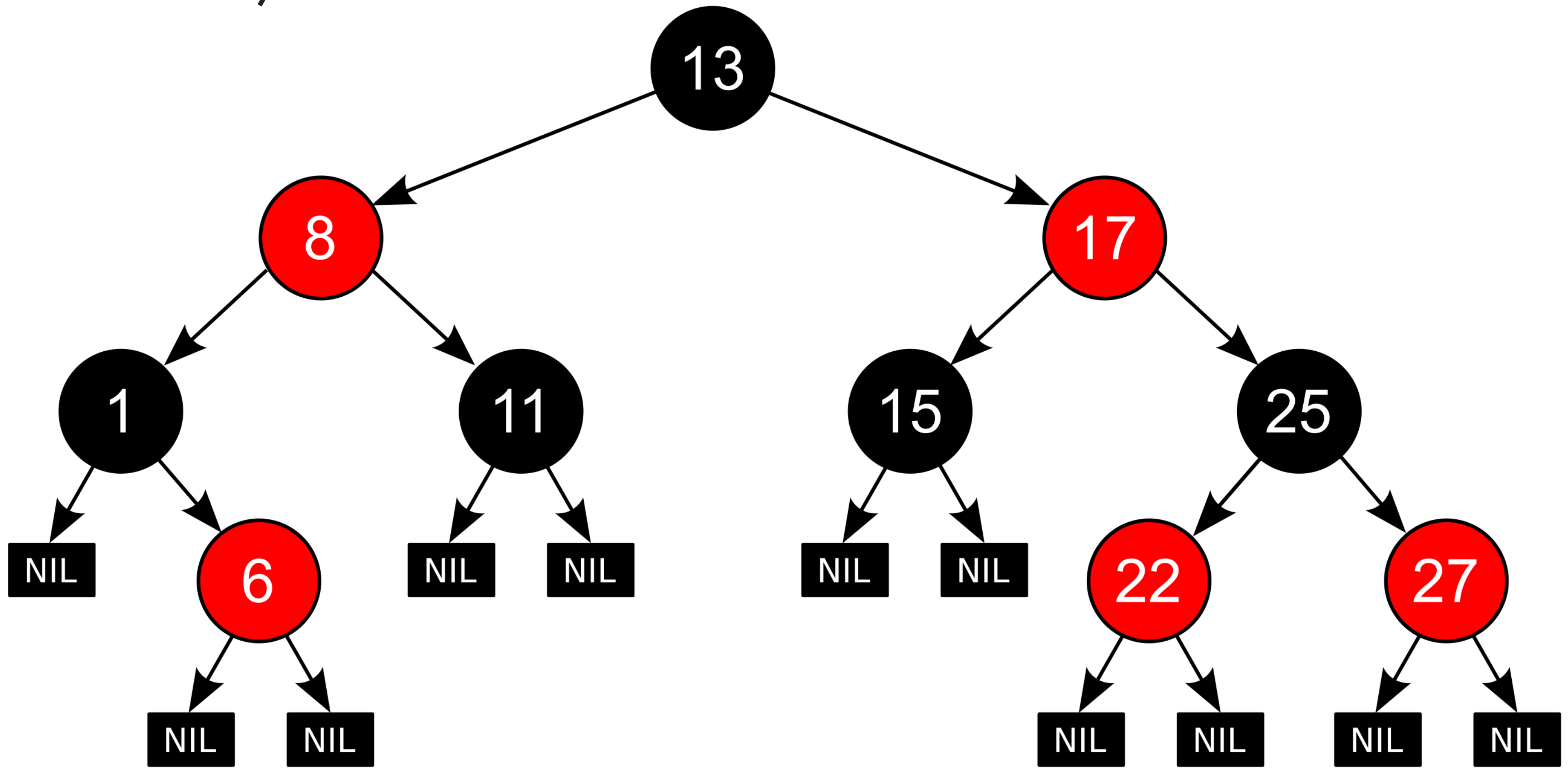
- Worst case: $\log_2(n+1)$ (all 2-nodes).
- Best case: $\log_4(n+1) = 0.5\log_2(n+1)$ (all 4-nodes)
    - That means that storing a million nodes will lead to a tree with height between 10 and 20, and storing a billion nodes to a tree with height between 13 and 27 (not bad!).
- Search and insert (and deletion) are $O(\log n)$!
- But implementation is a pain because of multiple node types and the overhead incurred could make the algorithms slower than standard BST search and insert.

2-3-4 trees are a special type of B-trees

- B-trees: trees of order m where every node has at most m links to children (and equivalently m-1 keys).
    - 2-3-4 search trees are a specific subcategory of B-trees with m=4.
    - Broadly used in file systems and databases.

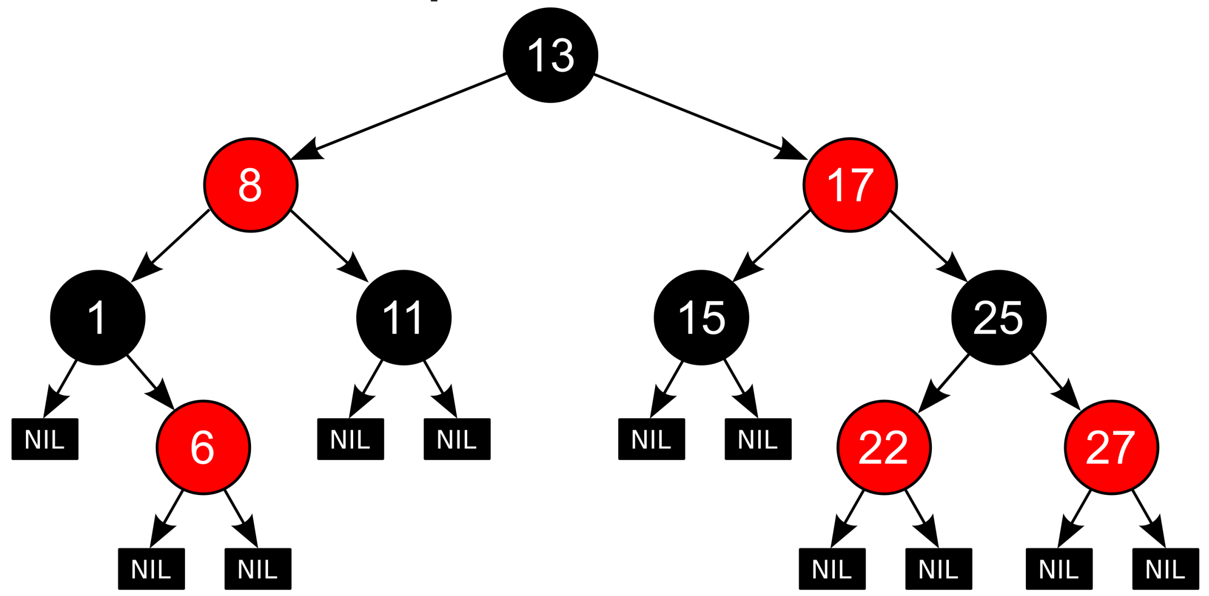
Other balanced search trees - Red-black trees

- Red-black trees: balanced **binary** search trees whose nodes, in addition to the key (and value), carry a color, either red or black. These nodes are known as internal nodes or non-NIL nodes. The leaf nodes do not contain keys and are known as NIL (they can also be omitted and be considered in place of a null link).



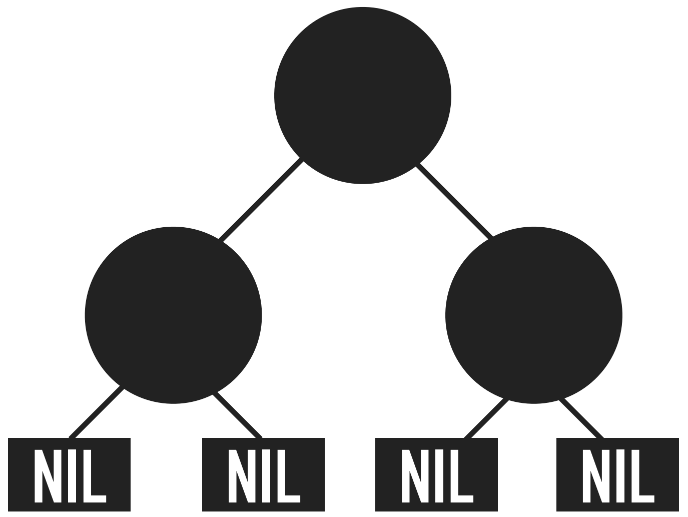
Properties of red-black Trees

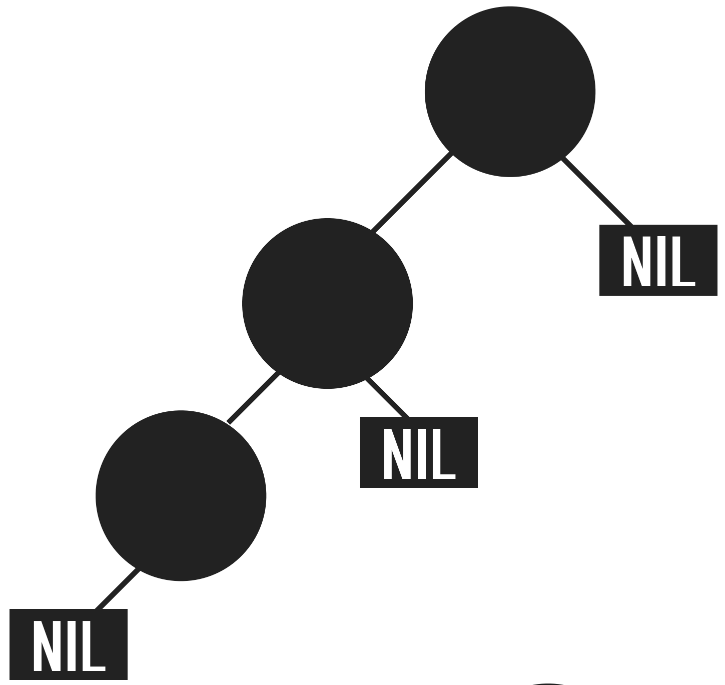
- Every interval node can either be red or black.
- The root has to be black.
- All leaves/NIL nodes are considered black.
- A red node cannot have a red child, i.e. no two consecutive red nodes on any path.
- Any root-leaf (i.e. NIL node) path has the same number of black nodes.

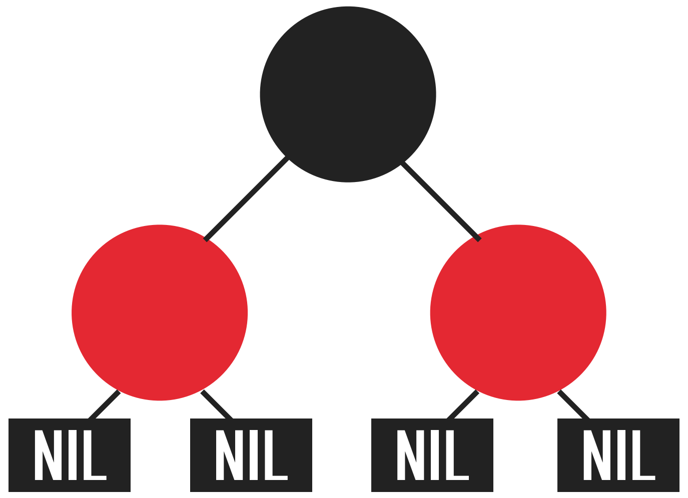


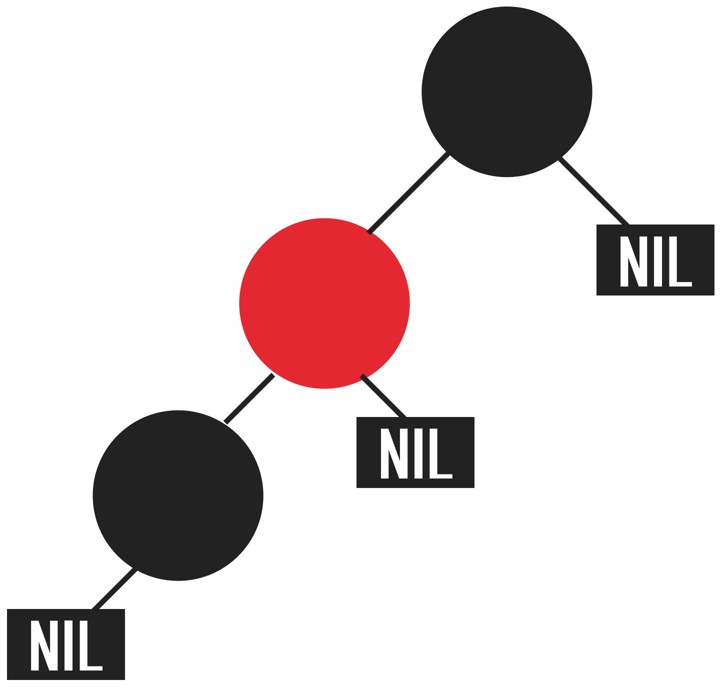
Practice Time

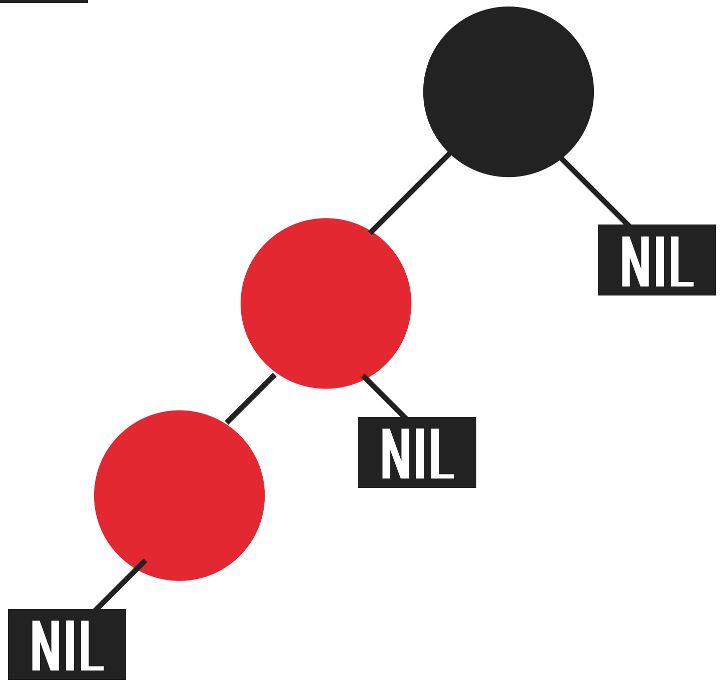
- Given the following trees, which ones are valid red-black trees?

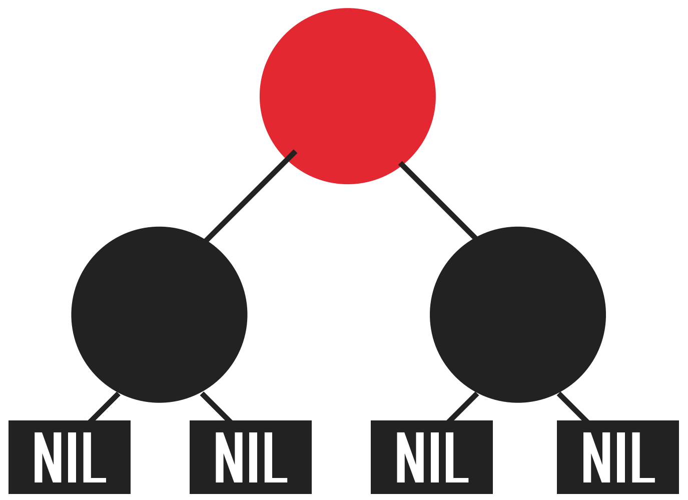






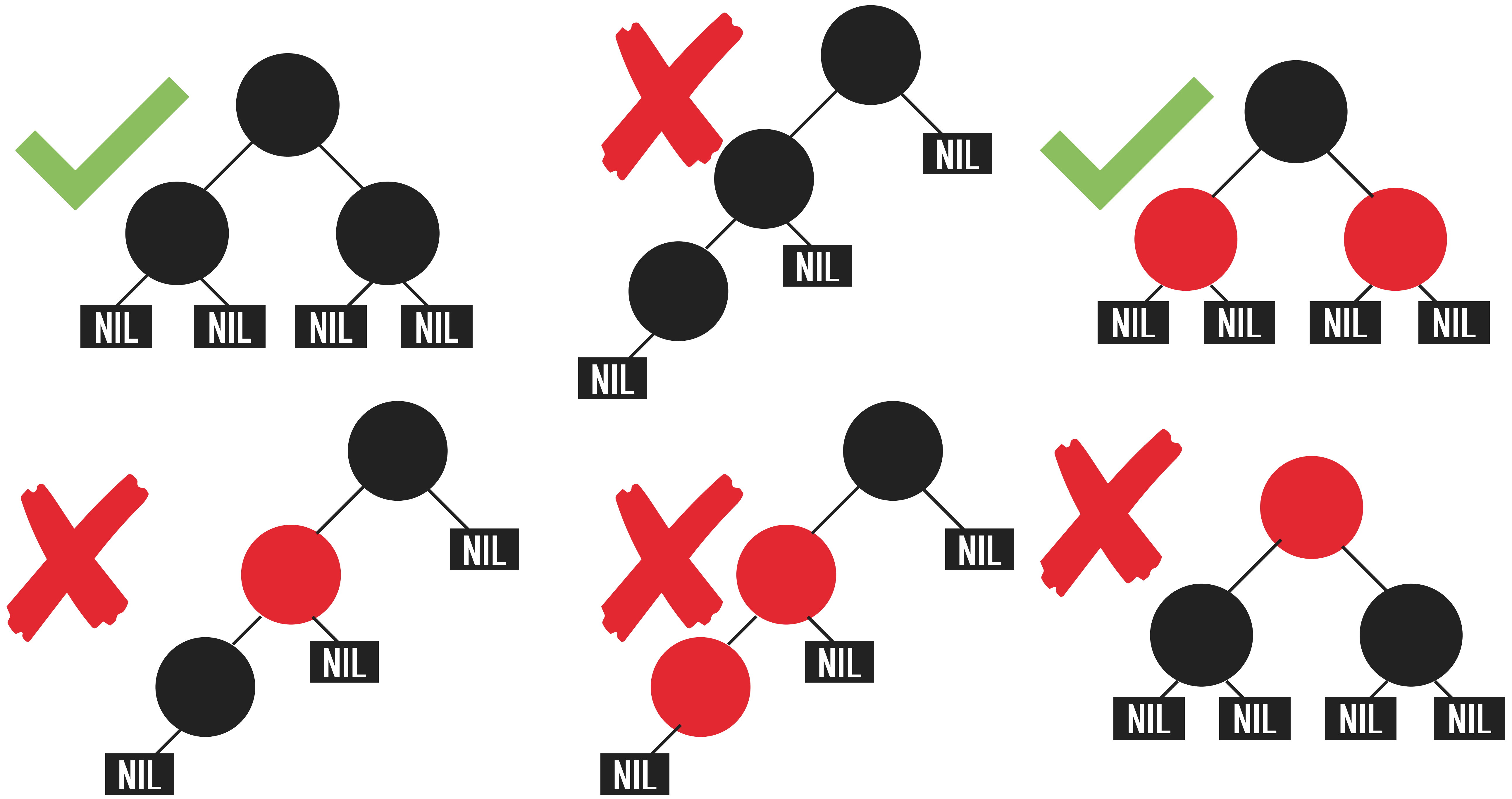






ANSWER

- Given the following trees, which ones are valid red-black trees?

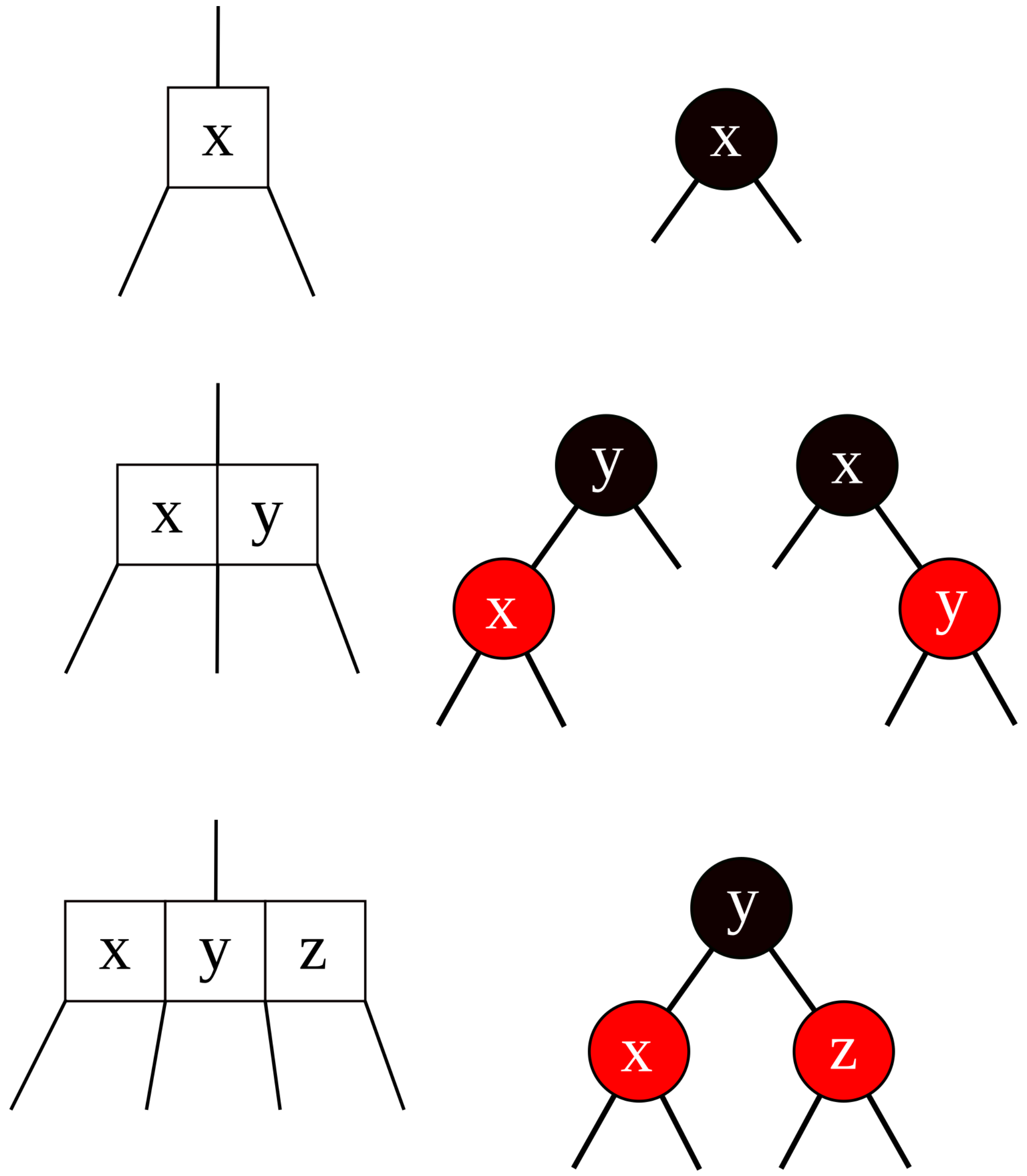


Height of red-black trees

- Stay tuned for proofs in CS140!
- The height is between $\log(n+1) \leq h \leq 2log(n+1)$.
- This implies that search (and insertion/deletion) is $O(\log n)$!

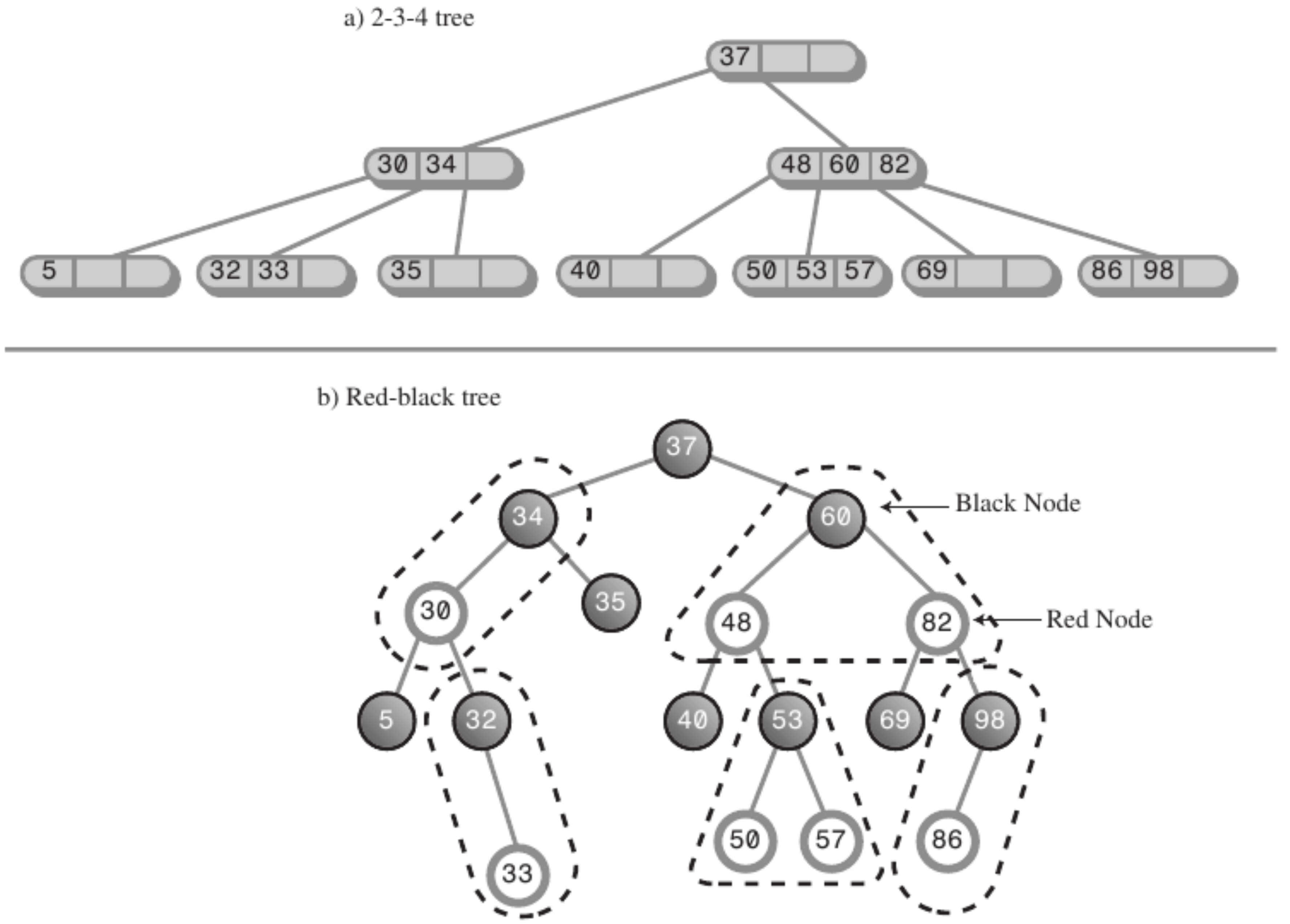
Red-black trees and 2-3-4 search trees

- There is a (not 1-1) correspondence between 2-3-4 search trees and red-black trees.



https://en.m.wikipedia.org/wiki/Red%E2%80%93black_tree#/media/File%3ARed-black_trees_and_2%E2%80%933%E2%80%934_trees.svg

2-3-4 search tree and red-black tree correspondence



Lafore - Data Structures and Algorithms in Java

Other balanced search trees

- You might also hear of splay trees or AVL trees, all offering amortized $O(\log n)$ performance for dictionary operations.

Summary for dictionary operations

	Worst case			Average case		
	Search	Insert	Delete	Search	Insert	Delete
BST	n	n	n	$\log n$	$\log n$	$\sqrt{n}$
Balanced search trees	$\log n$	$\log n$	$\log n$	$\log n$	$\log n$	$\log n$

Readings:

- Recommended Textbook: Chapter 3.3 (Pages 424-447)
- Website:
  - https://algs4.cs.princeton.edu/33balanced/
- Visualization:
  - https://yongdanielliang.github.io/animation/web/24Tree.html

Worksheet:

- Lecture worksheet

Problem 1

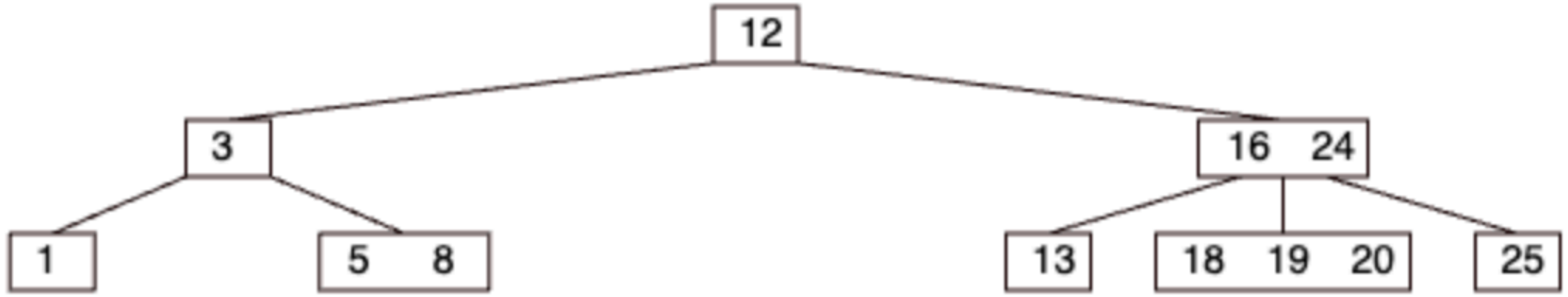
- Draw the 2-3-4 tree that results when you insert the keys 25, 12, 16, 13, 24, 8, 3, 18, 1, 5, 19, 20 in that order into an initially empty tree.

Problem 2

- Find an insertion order for the keys 19, 5, 1, 18, 3, 8, 24, 13 that leads to a 2-3-4 tree of height 1.

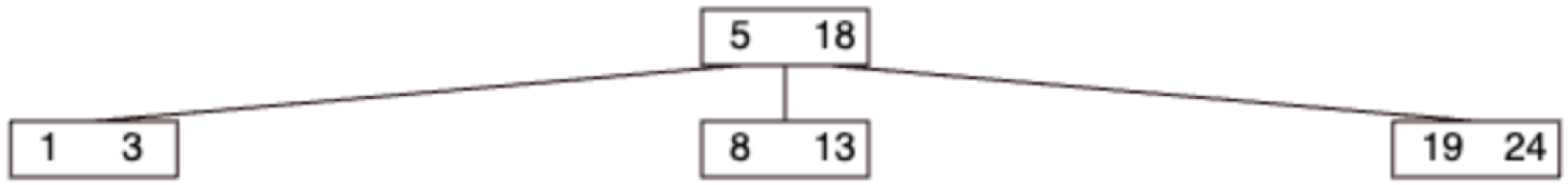
ANSWER 1

- Draw the 2-3-4 tree that results when you insert the keys 25, 12, 16, 13, 24, 8, 3, 18, 1, 5, 19, 20 in that order into an initially empty tree.



ANSWER 2

- Find an insertion order for the keys 19, 5, 1, 18, 3, 8, 24, 13 that leads to a 2-3-4 tree of height 1.
- Example of insertion order: 5 1 13 24 18 3 8 19


5 18
1 3
8 13
19 24