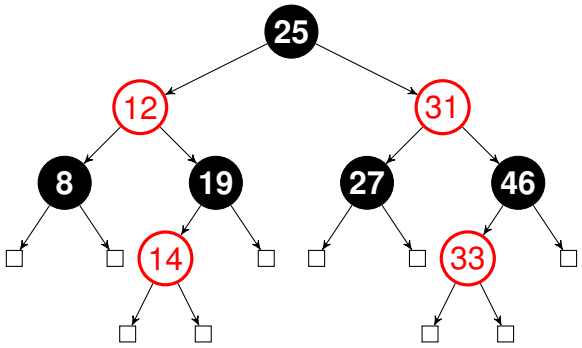
1 Balanced Search Trees

(a) Convert the red-black tree into a 2-4 tree. Solid nodes are black.



(b) Insert the keys 13 and 17 into the resulting 2-4 tree. Assume that, if a node has 4 keys, we choose to push up the left of the 2 middle keys (so the 2nd key from the left).

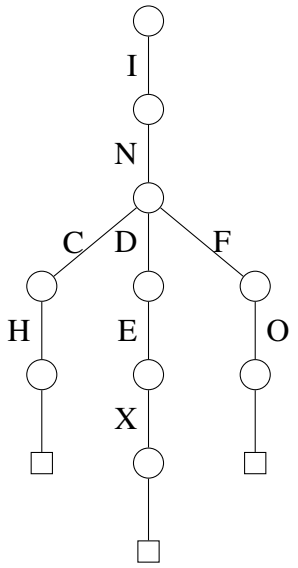
(c) Convert the resulting 2-4 tree into a valid left-leaning red-black tree.

(d) Given a 2-4 tree containing N keys, describe how you can obtain the keys in sorted order in worst case $O(N)$ time.

(e) If a 2-4 tree has depth H (that is, the leaves are at a distance of H from the root), what is the maximum number of comparisons done in the corresponding red-black tree to find whether a certain key is present in the tree?

2 Tries

List the words encoded by the following trie, then draw the resulting trie after inserting the words *indent*, *inches*, and *trie*.



3 Skip Lists

Draw the resulting skip list after adding the following numbers at the specified random heights. Highlight the links traversed to find 148.

Number	40	41	43	48	54	59	77	128	131	139	148	161	170	179	189
Height	2	1	3	1	3	1	4	2	2	1	1	3	2	1	1