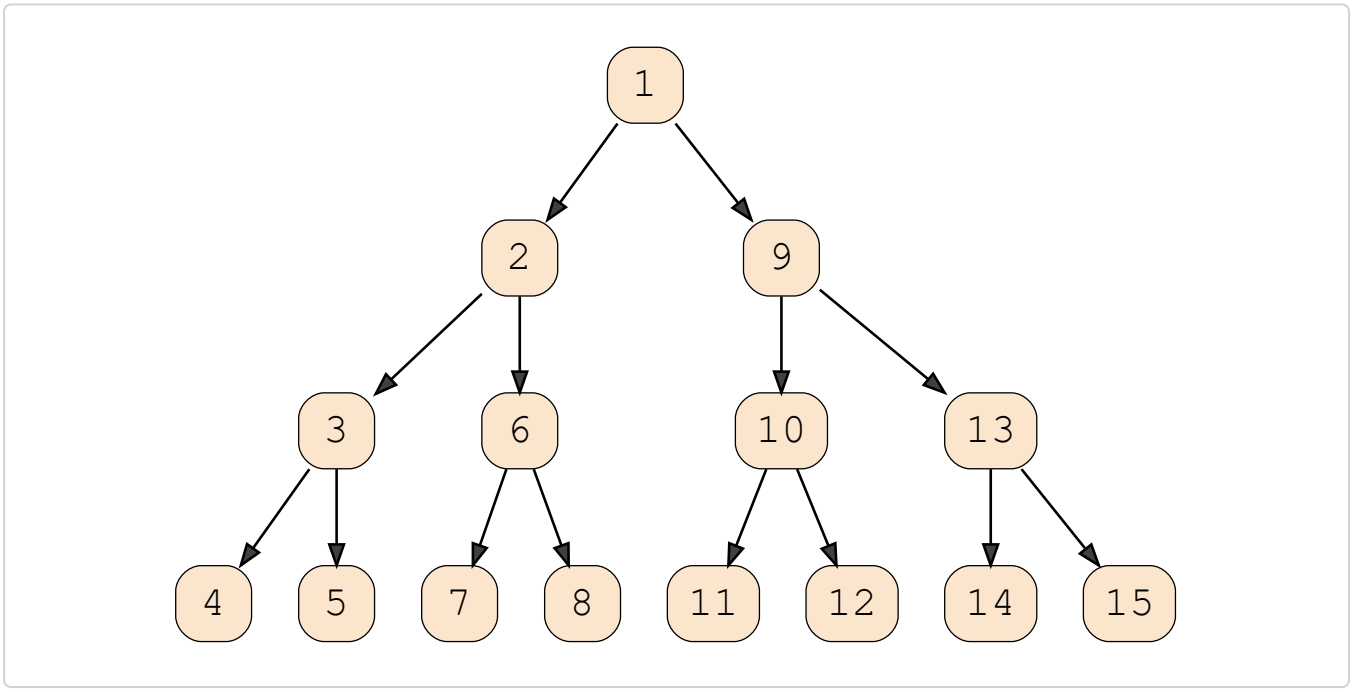
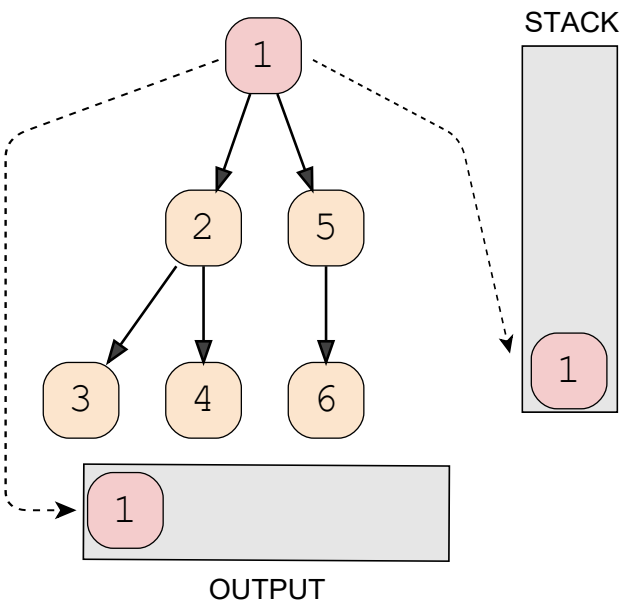
Depth First Traversal

As the name suggests, depth-first traversal involves traversing a tree from top to bottom. (Reading time: 2 minutes)

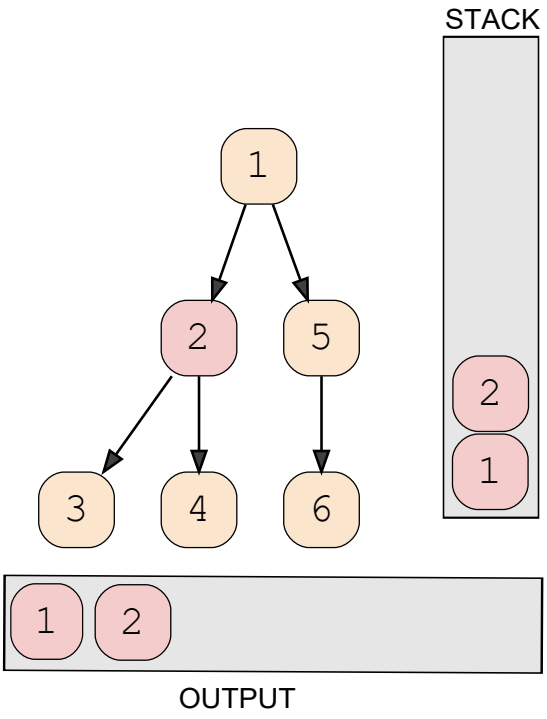


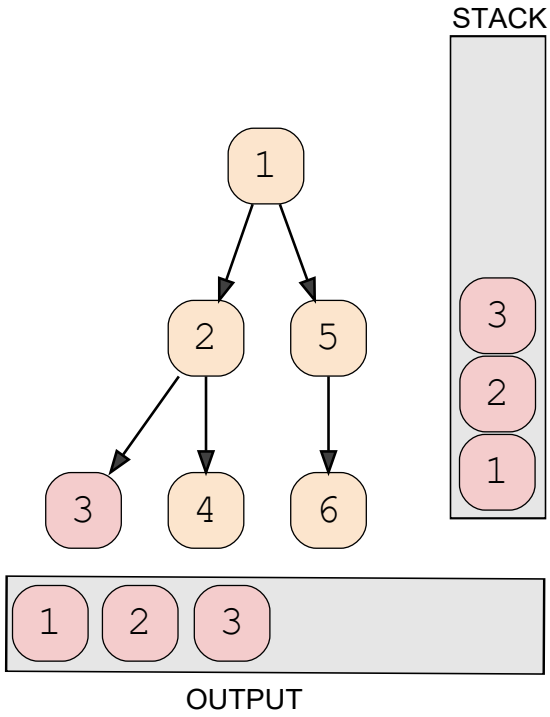
In depth-first traversal, we first traverse through the **left subtrees**, and then the **right subtrees**! The values within the nodes in this example show the order, not their values!

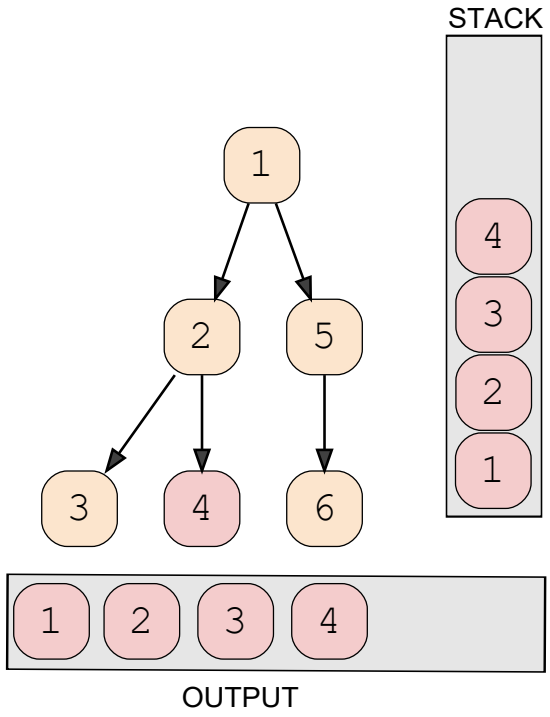
Depth-first traversal uses a **stack** data structure. The stack keeps track of all the visited nodes! However, the stack is implemented implicitly.

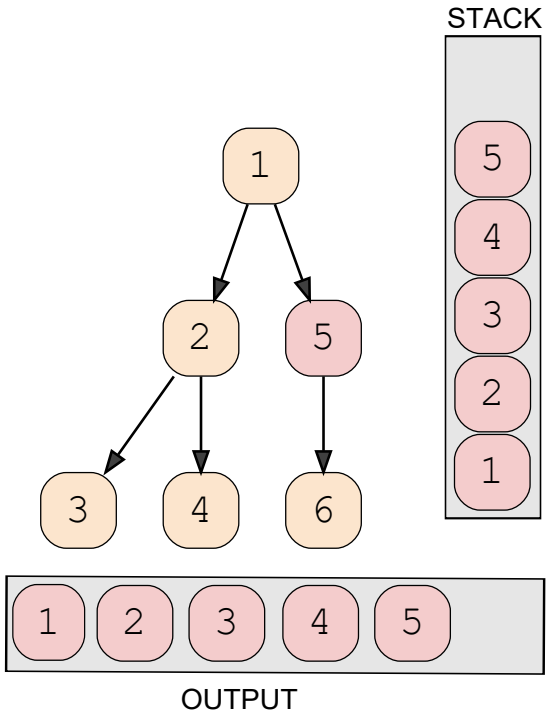


We visit the first node. The node has the value **1**, so that value gets pushed to the stack and added to the output sequence. This node has other nodes, so that value doesn't get popped off the stack just yet. Instead, we go to the next value.

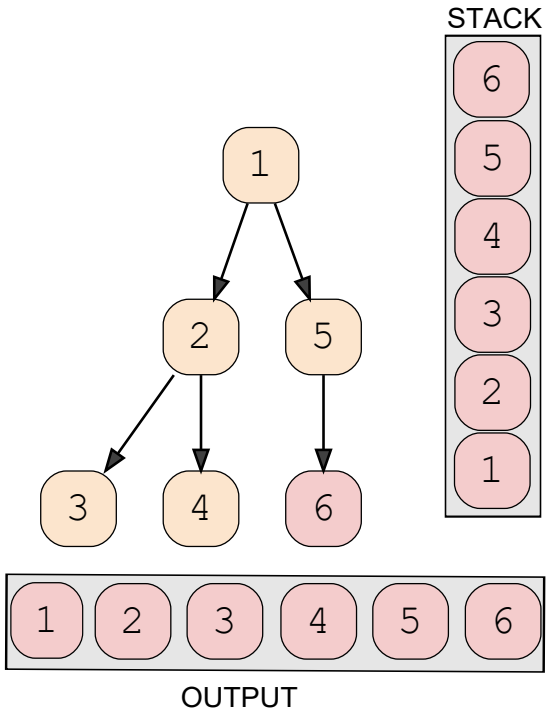



STACK
1
2
5
3
4
6
3
2
1
1
2
3
OUTPUT


STACK
1
2
5
3
4
6
4
3
2
1
1
2
3
4
OUTPUT



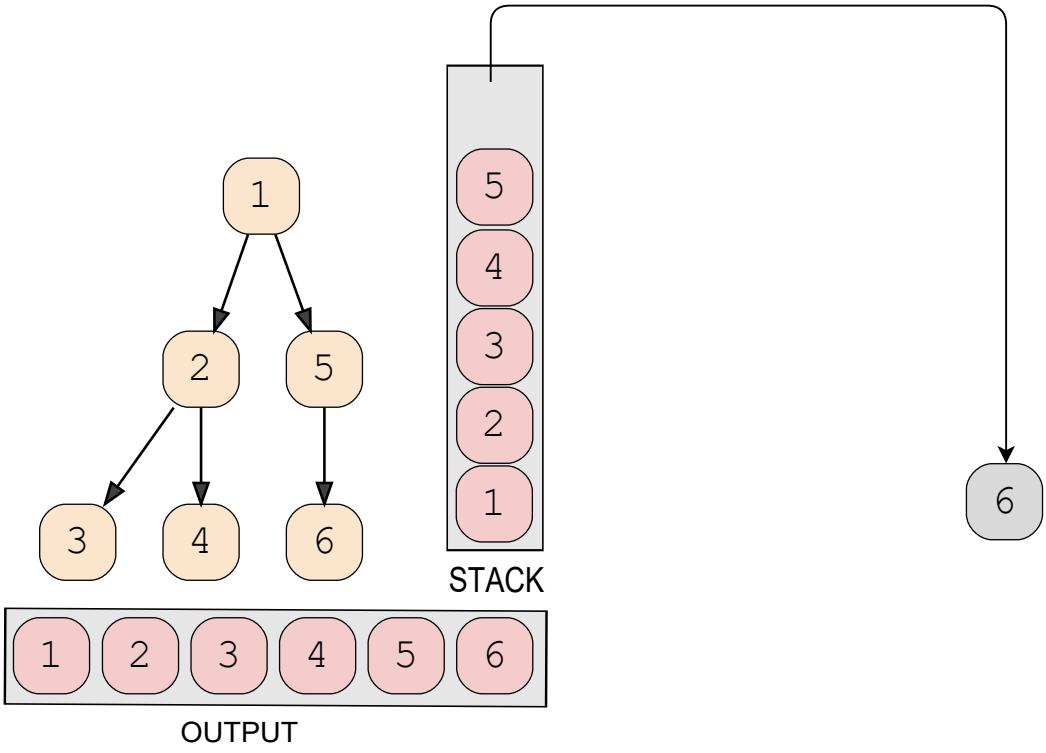
4 of 5



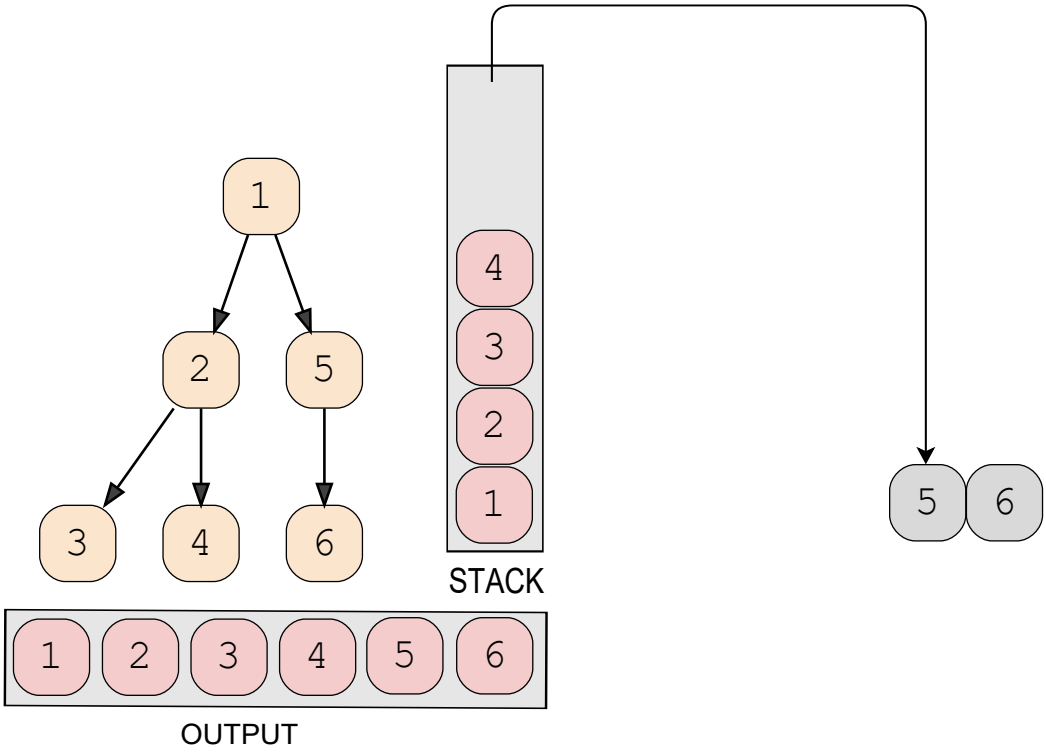
5 of 5

We reached the end of the tree: there are no other nodes to visit! Now, the values will be popped off the stack until the stack is **empty**. This is the sign

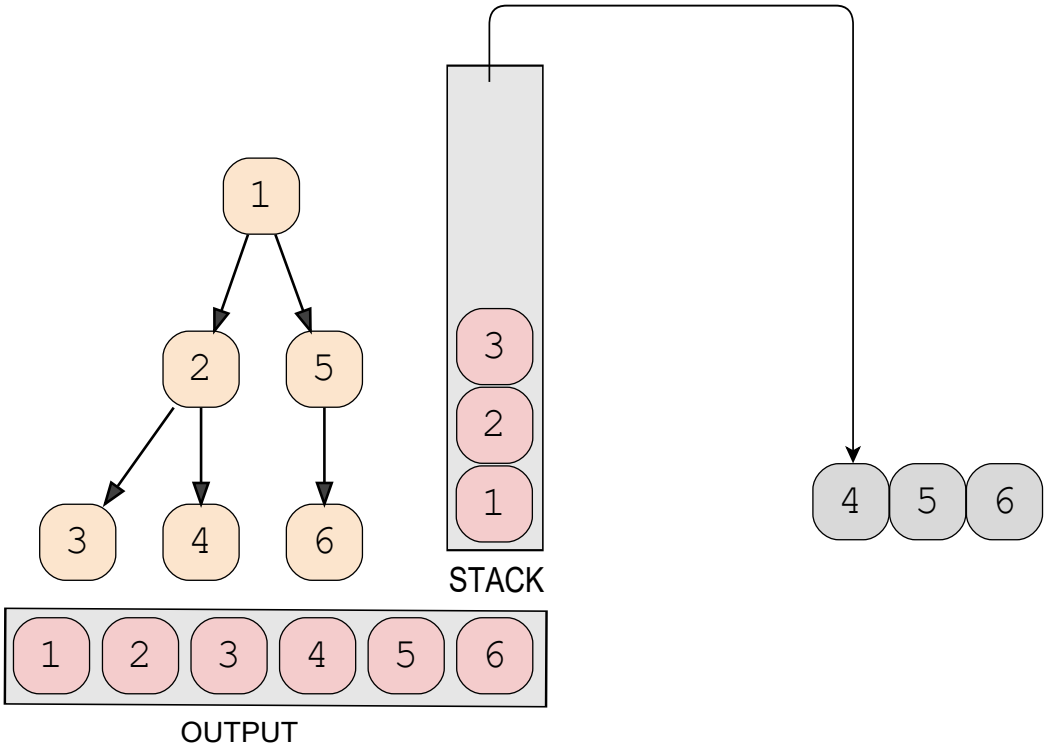
that we have finished traversing through the tree, and we have the complete output!

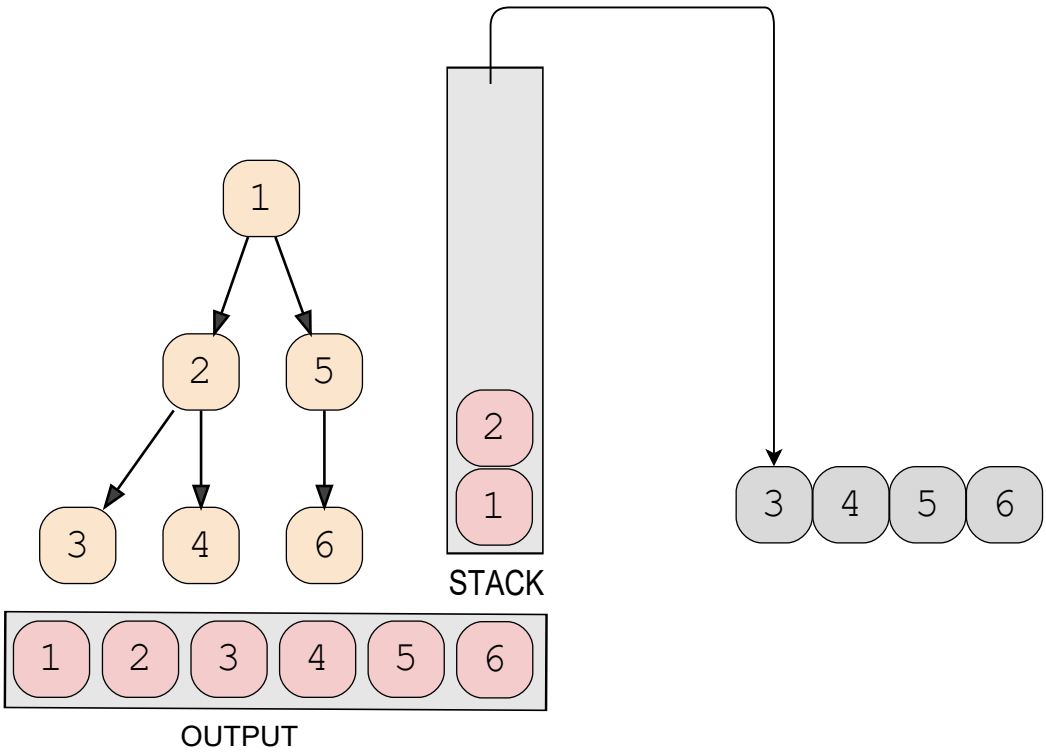


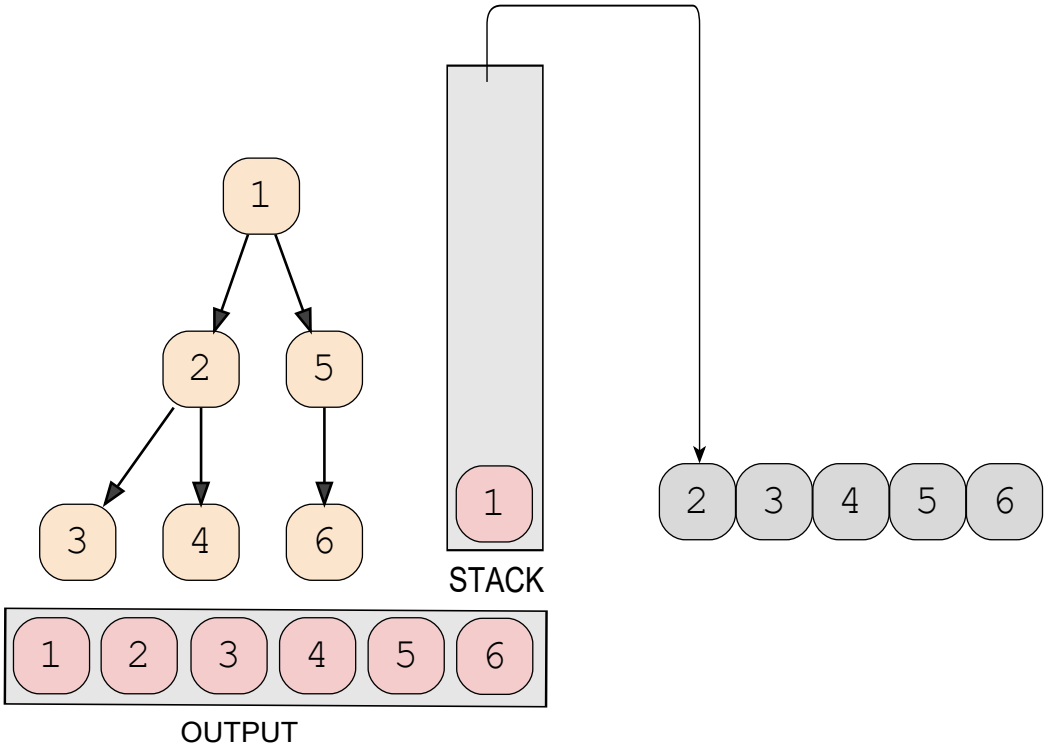
1 of 6



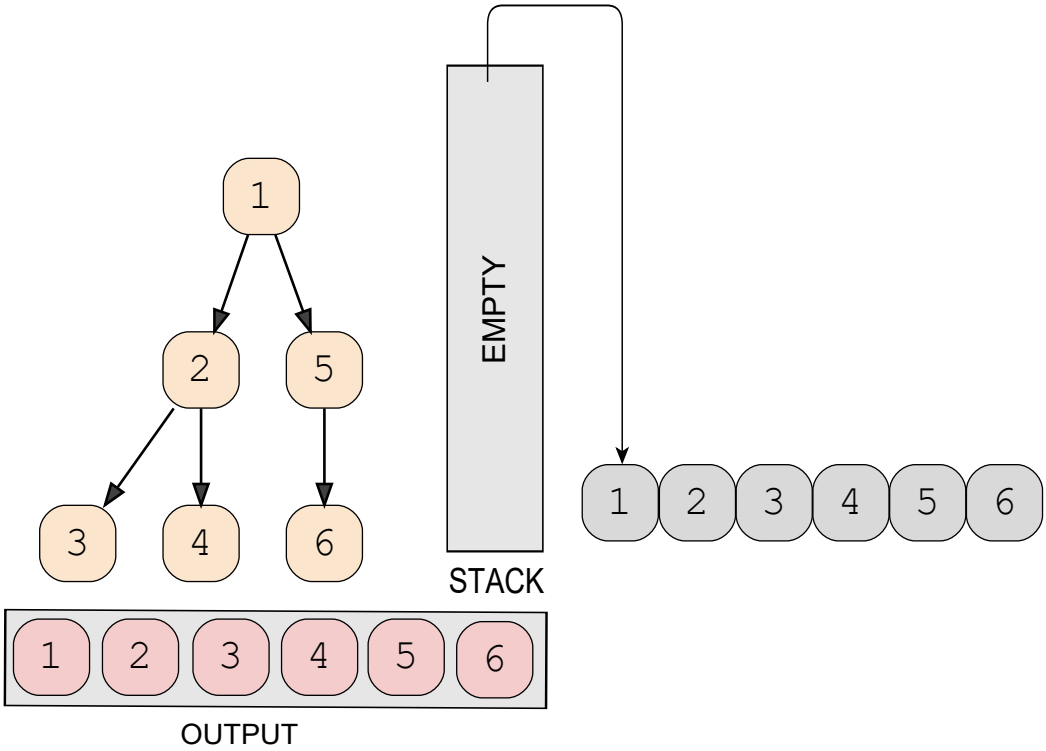
2 of 6


1
2
5
3
4
6
3
2
1
STACK
4
5
6
1
2
3
4
5
6
OUTPUT


1
2
5
3
4
6
2
1
STACK
3
4
5
6
1
2
3
4
5
6
OUTPUT



5 of 6



6 of 6

Now, let's move on to breadth-first traversal.