Assignment #4 Data Structures

1. Design a stack that supports push, pop, top, and retrieving the minimum element in constant time.

   push(x) – Push element x onto stack.
   pop() – Removes the element on top of the stack.
   top() – Get the top element.
   getMin() – Retrieve the minimum element in the stack.

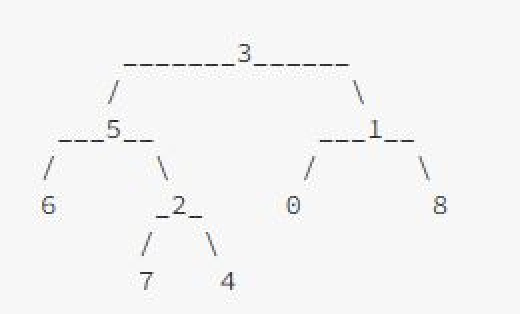
   Note : All the operations have to be constant time operations.
   Do nothing if the operation cannot be performed and return -1 in all such cases.

2. You are given two linked lists representing two non-negative numbers. The digits are stored in reverse order and each of their nodes contains a single digit. Add the two numbers and return it as a linked list.

   Input: (2 -> 4 -> 3) + (5 -> 6 -> 4)
   Output: 7 -> 0 -> 8

   342 + 465 = 807
   Note: Make sure there are no trailing zeros in the output list
   So, 7 -> 0 -> 8 -> 0 is not a valid response even though the value is still 807.

3. Find the lowest common ancestor in an unordered binary tree given two values in the tree.
   Note : Lowest common ancestor : the lowest common ancestor (LCA) of two nodes v and w in a tree or directed acyclic graph (DAG) is the lowest (i.e. deepest) node that has both v and w as descendants.



```
        _______3______
       /              \
    ___5__          ___1__
   /      \        /      \
   6      _2_      0       8
         /  \
         7   4
```

The LCA of 5 and 1 is 3. For nodes 5 and 4 is 5.

You are given 2 values. Find the lowest common ancestor of the two nodes represented by val1 and val2
No guarantee that val1 and val2 exist in the tree. If one value doesn’t exist in the tree then return -1.
There are no duplicate values.

You can use extra memory, helper functions, and can modify the node struct but, you can’t add a parent pointer.