1. Remove duplicates from Linked list
    a. 2 ways iterative and recursive
    b. https://leetcode.com/problems/remove-duplicates-from-sorted-list/submissions/
    c. https://leetcode.com/problems/remove-duplicates-from-sorted-list-ii/
2. Reverse Linked List
    a. https://leetcode.com/problems/reverse-linked-list-ii/solution/
3. Detect cycle in linked list
    a. Fast slow pointer concept
    b. Once they intersect put slow at head and inc both 1 step again when they intersect that is point where cycle is generated.
    c. https://leetcode.com/problems/linked-list-cycle-ii/solution/
    d. Do see complexity analysis https://leetcode.com/problems/linked-list-cycle/solution/
4. Copy list with random pointers
    a. https://leetcode.com/problems/copy-list-with-random-pointer/
        b. Traverse first list and create second clone list
        c. Now traverse the first list and store index <address, index> in map1.
        Traverse second list and store <index, address> in map2.
        d. Now traverse both list together and find random address index using map1 and using map2 assign its address to random pointer of current node.
    e. https://leetcode.com/problems/copy-list-with-random-pointer/discuss/43491/A-solution-with-constant-space-complexity-O(1)-and-linear-time-complexity-O(N)
        - Iterate the original list and duplicate each node. The duplicate
        of each node follows its original immediately.
        - Iterate the new list and assign the random pointer for each
        duplicated node.
        - Restore the original list and extract the duplicatenodes.