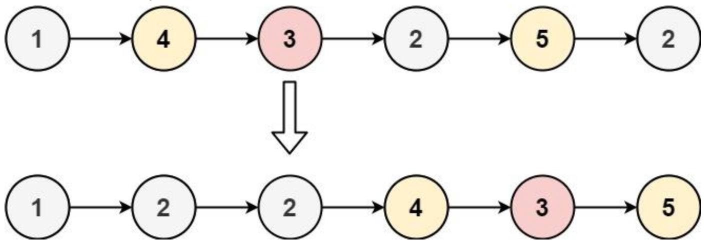
PSG COLLEGE OF TECHNOLOGY
DEPARTMENT OF COMPUTER APPLICATIONS
23MX17- DATA STRUCTURES LABORATORY
PROBLEM SHEET ON LINKED LIST

1) Given the head of a linked list and a value x, partition it such that all nodes less than x come before nodes greater than or equal to x. You should preserve the original relative order of the nodes in each of the two partitions.

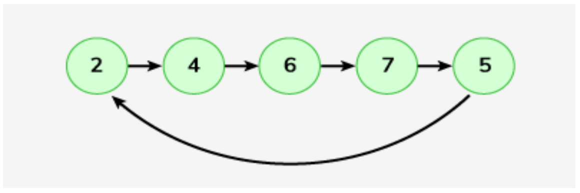


Input: head = [1,4,3,2,5,2], x = 3
Output: [1,2,2,4,3,5]
Example 2:

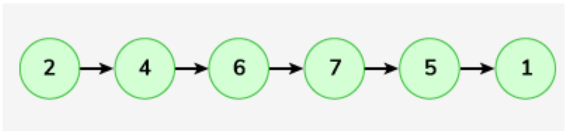
Input: head = [2,1], x = 2
Output: [1,2]

2) Given the head, the head of a singly linked list, Returns true if the linked list is circular & false if it is not circular. A linked list is called circular if it is not NULL terminated and all nodes are connected in the form of a cycle. Note: The linked list does not contain any inner loop.

Examples:
Input:

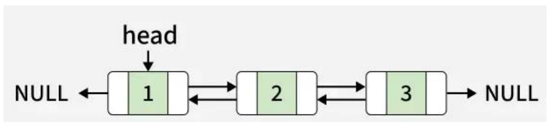


Output: true
Explanation: As shown in figure the first and last node is connected, i.e. 5 --> 2
Input:



Output: false
Explanation: As shown in figure this is not a circular linked list.

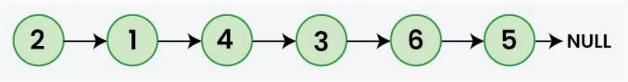
3) Given a reference to the head of a doubly-linked list and an integer, *data*, create a new DoublyLinkedListNode object having data value *data* and insert it at the proper location to maintain the sort.
Example
head refers to the list 1 ↔ 2↔4 ↔null
data=3
Return a reference to the new list: 1 ↔ 2↔3 ↔4 ↔null.

4) Given the head of a Doubly Linked List, reverse the list in-place so that the first node becomes the last, the second node becomes the second last, and so on. Return the new head of the reversed list.
Examples:
Input: 

Output: 3 <-> 2 <-> 1 -> NULL

5) Given a singly linked list, the task is to swap linked list elements pairwise.

Input : *1 -> 2 -> 3 -> 4 -> 5 -> 6 -> NULL*
Output : *2 -> 1 -> 4 -> 3 -> 6 -> 5 -> NULL*



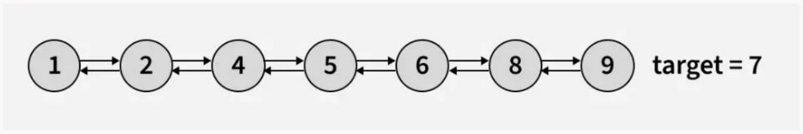
Input : *1 -> 2 -> 3 -> 4 -> 5 -> NULL*
Output : *2 -> 1 -> 4 -> 3 -> 5 -> NULL*

6) Given a singly linked list of 0s, 1s and 2s, The task is to sort the list in non-decreasing order.
Examples:
Input: 1 -> 1 -> 2 -> 0 -> 2 -> 0 -> 1 -> NULL
Output: 0 -> 0 -> 1 -> 1 -> 1 -> 2 -> 2 -> NULL
Input: 1 -> 1 -> 2 -> 1 -> 0 -> NULL
Output: 0 -> 1 -> 1 -> 1 -> 2 -> NULL

7) Given a sorted doubly linked list of positive distinct elements, the task is to find pairs in a doubly-linked list whose sum is equal to the given value x in sorted order.

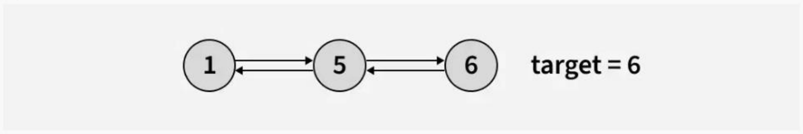
Examples:

Input:



Output: *(1, 6), (2,5)*
Explanation: *We can see that there are two pairs (1, 6) and (2, 5) with sum 7.*

Input:

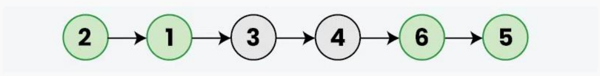


Output: *(1,5)*
Explanation: *We can see that there is one pair (1, 5) with a sum of 6.*

8) Given a linked list, The task is to reverse alternate k nodes. If the number of nodes left at the end of the list is fewer than k, reverse these remaining nodes or leave them in their original order, depending on the alternation pattern.

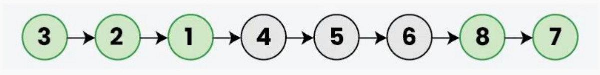
Example:

Input: *1 -> 2 -> 3 -> 4 -> 5 -> 6 -> NULL, k = 2*
Output: *2 -> 1 -> 3 -> 4 -> 6 -> 5 -> NULL.*



Explanation :*The nodes are reversed alternatively after 2 nodes.*

Input: *1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> NULL, k = 3*
Output: *3 -> 2 -> 1 -> 4 -> 5 -> 6 -> 8 -> 7-> NULL.*

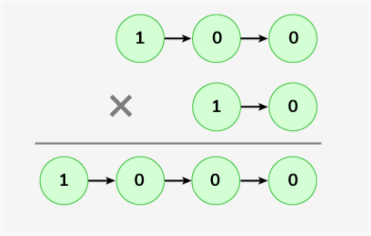


Explanation :*The nodes are reversed alternatively after 3 nodes.*

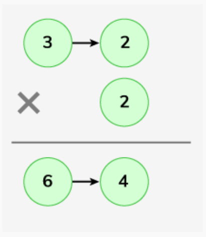
9) Given two numbers represented by linked lists, The task is to return the multiplication of these two linked lists.

Examples:

***Input** : **head1** : 1->0->0 , **head2** : 1->0*
***Output**: 1000*
***Explanation**: head1 represents **100** and head2 represents the number **10**, 100 x 10 = 1000*



***Input** : **head1** : 3->2, **head2** : 2*
***Output**: 64*
***Explanation**: head1 represents **32** and head2 represents the number **2**, 32 x 2= 64*

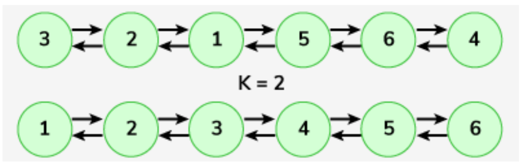


10) Given a doubly linked list, each node is at most k-indices away from its target position. The problem is to sort the given doubly linked list. The distance can be assumed in either of the directions (left and right).

Examples :

Input: Doubly Linked List : 3 <-> 2 <-> 1 <-> 5 <-> 6 <-> 4 , k = 2

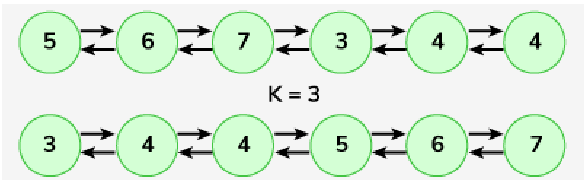
Output: 1 <-> 2 <-> 3 <-> 4 <-> 5 <-> 6



Explanation: After sorting the given 2-sorted list is 1 <-> 2 <-> 3 <-> 4 <-> 5 <-> 6.

Input: Doubly Linked List : 5 <-> 6 <-> 7 <-> 3 <-> 4 <-> 4 , k = 3

Output: 3 <-> 4 <-> 4 <-> 5 <-> 6 <-> 7



Explanation: After sorting the given 3-sorted list is 3 <-> 4 <-> 4 <-> 5 <-> 6 <-> 7.