AGENDA

Linked Lists (Contd.)

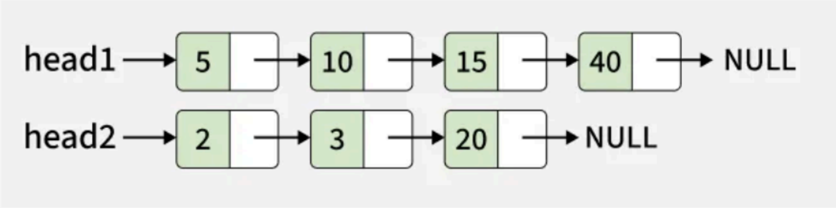
- Check if a linked list is a palindrome
- Detect and remove the loop from a Linked list (Floyd’s Cycle Detection Algorithm)
- Merge 2 Sorted Linked Lists
- K Reverse Linked List
- Remove duplicates from a sorted doubly linked list

Merge 2 Sorted Linked Lists

Given the **head** of two **sorted linked lists** consisting of nodes respectively. **Merge** both lists and return the **head** of the **sorted merged list**.

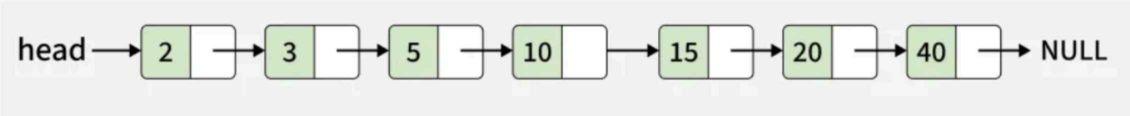
Examples:

Input:

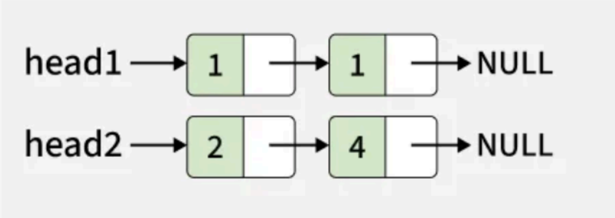


Output: 2 -> 3 -> 5 -> 10 -> 15 -> 20 -> 40

Explanation:

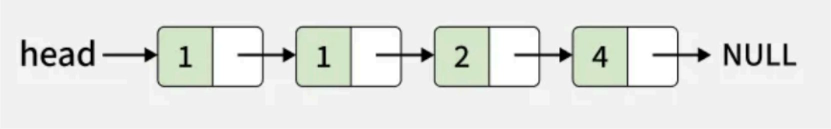


Input:

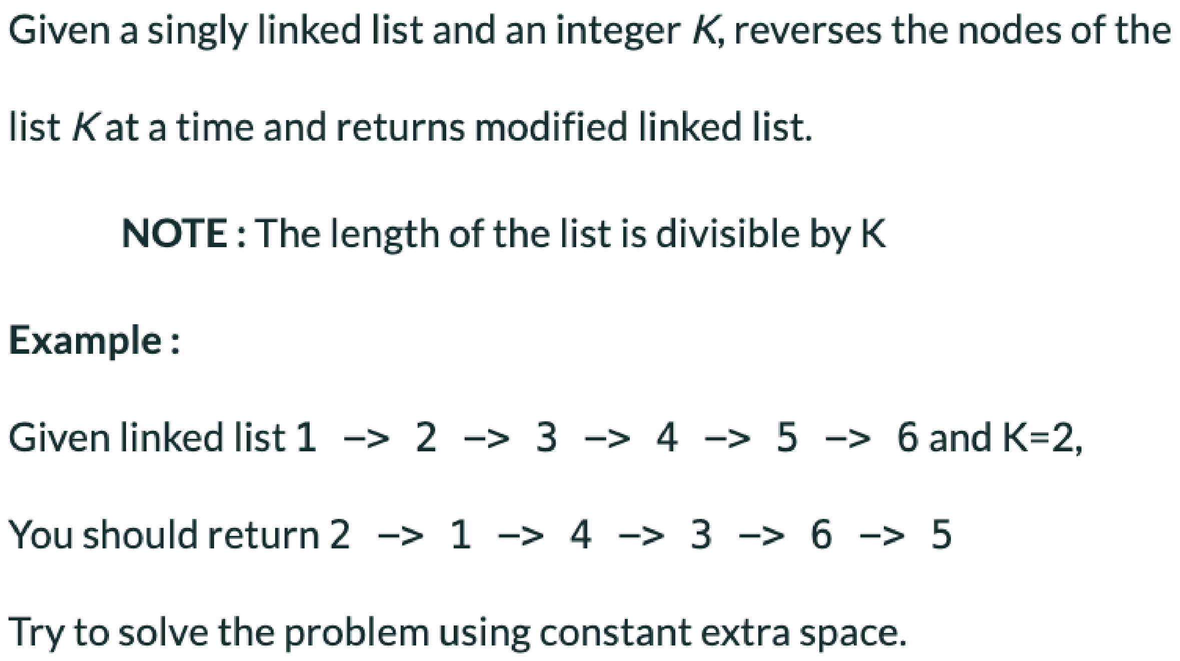


Output: 1 -> 1 -> 2 -> 4

Explanation:



K Reverse Linked List



Given a singly linked list and an integer *K*, reverses the nodes of the list *K* at a time and returns modified linked list.

> **NOTE :** The length of the list is divisible by K

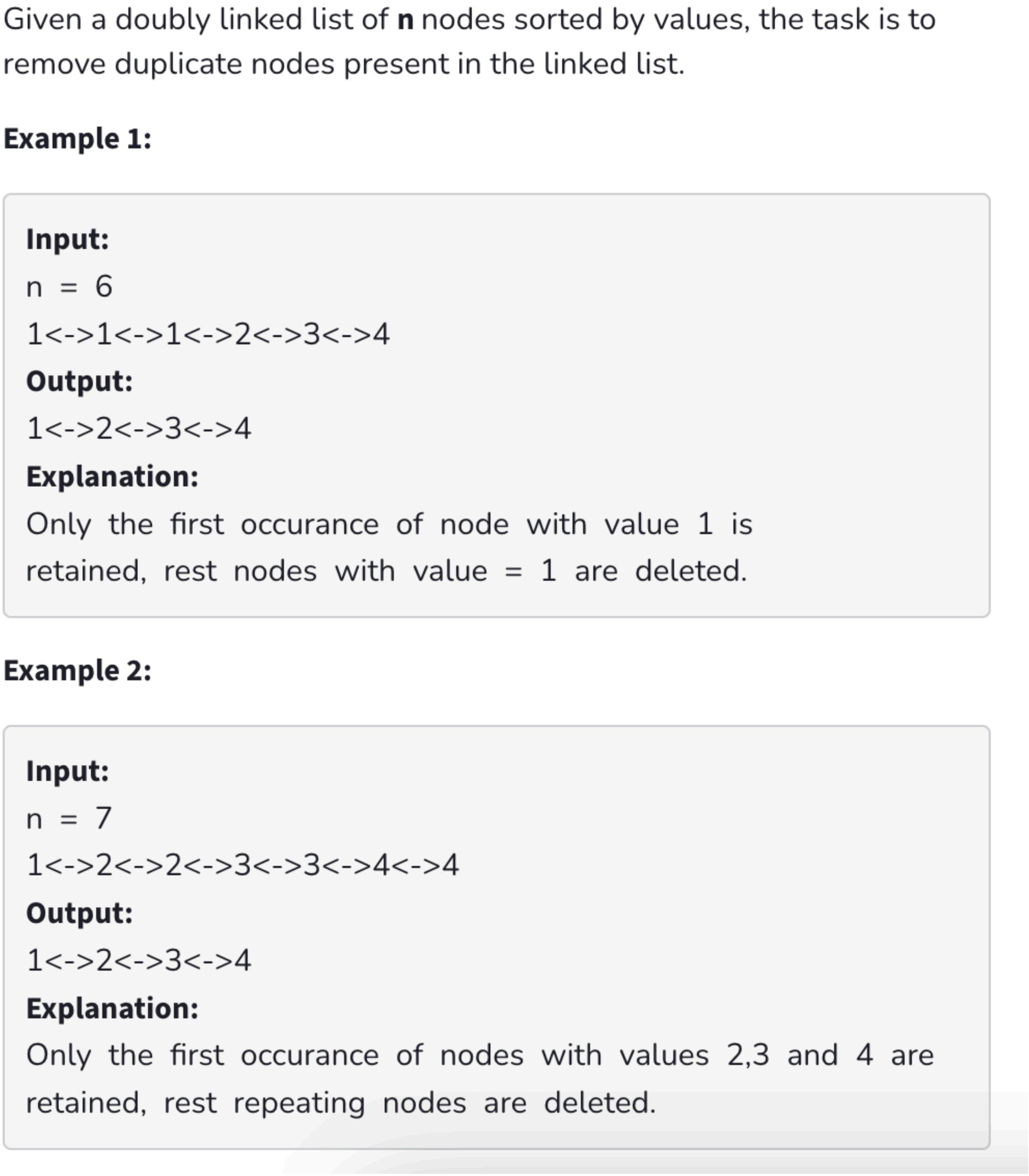
Example :

Given linked list 1 –> 2 –> 3 –> 4 –> 5 –> 6 and K=2,

You should return 2 –> 1 –> 4 –> 3 –> 6 –> 5

Try to solve the problem using constant extra space.

Remove duplicates from a sorted doubly linked list



Given a doubly linked list of **n** nodes sorted by values, the task is to remove duplicate nodes present in the linked list.

Example 1:

```
Input:
n = 6
1<->1<->1<->2<->3<->4
Output:
1<->2<->3<->4
Explanation:
Only the first occurance of node with value 1 is
retained, rest nodes with value = 1 are deleted.
```

Example 2:

```
Input:
n = 7
1<->2<->2<->3<->3<->4<->4
Output:
1<->2<->3<->4
Explanation:
Only the first occurance of nodes with values 2,3 and 4 are
retained, rest repeating nodes are deleted.
```