**2.1 Introduction to Linked List | Need of Linked List**

|  |  |  |  |
| --- | --- | --- | --- |
| **Easy Level Problems** | | | |
| 1. Print the Middle of a given linked list |  | 1. Circular Linked List Traversal |  |
| 1. Reverse a Linked List |  | 1. Deletion from a Circular Linked List |  |
| 1. Reverse a Doubly Linked List |  | 1. Delete without head pointer |  |
| 1. Rotate a linked list. |  | 1. Implement Queue using Linked List |  |
| 1. Nth node from end of linked list |  | 1. Implement a stack using singly linked list |  |
| 1. Delete last occurrence of an item from linked list |  | 1. Remove every k-th node of the linked list |  |
| 1. Delete middle of linked list |  | 1. Pairwise swap of a Linked list |  |
| 1. Remove duplicate elements from sorted linked list |  | 1. Occurence of an integer in a Linked List |  |
| 1. Detect Loop in linked list |  | 1. Given a Linked list of 0s, 1s and 2s, sort it |  |
| 1. Delete N nodes after M nodes of a linked list |  | 1. Deletion in Linked List |  |
| 1. Merge a linked list into another linked list at alternate positions |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

| **Medium Level Problems on Linked List Data Structure** |
| --- |
| Convert singly linked list into circular linked list |
| Reverse a linked list in groups of given size |
| Merge two sorted linked lists |
| Remove loop in Linked List |
| Function to check if a singly linked list is palindrome |
| Remove all occurrences of one Linked list in another Linked list |
| Intersection point in Y shaped Linked lists |
| Intersection of two Sorted Linked Lists |
| Split a Circular Linked List into two halves |
| Find pairs with given sum in doubly linked list |
| Remove duplicates from an unsorted doubly linked list |
| Intersection point of two Linked Lists. |
| Add two numbers represented by Linked lists |
| Multiply two numbers represented by Linked Lists |
| Swap Kth node from beginning with Kth node from end in a Linked List |
| Sort a k sorted doubly linked list |
| Rotate Doubly linked list by N nodes |
| Convert a Binary Tree into Doubly Linked List in spiral fashion |
| Convert a given Binary Tree to Doubly Linked List |
| Construct a linked list from 2D matrix |
| Reverse a doubly linked list in groups of given size |

| **Hard Level Problems on Linked List Data Structure** |
| --- |
| Reverse a sublist of linked list |
| Rearrange a given linked list in-place. |
| Reverse Alternate K Nodes in a Singly Linked List: |
| Reverse a Linked List in groups of given size |
| Merge k Sorted Linked Lists |
| Flattening a Linked List |
| Partition a linked list around a given value |
| Clone a linked list with random pointers |