Linked List Pattern

1. Creating a linked list :

here we have to implement a linked list or variation of a linked list like singly , doubly , circular etc

It can be done with creating a node which have a value and a pointer to the next node

Question :

- Remove Linked List Elements
- Delete Node in a Linked List
- Design Linked List

Trick :

When working with linked list its better to create a dummy node as head and tail it helps us to insert in between these nodes and these will help us to not handle different edge cases like head or tail as null

2. Combine 2 linked list or Sort Linked list :

This pattern allow us to combine 2 list in a single list like merge sort used to solve the linked list also

Question :

- Merge Two Sorted Lists
- Sort List
- Insertion Sort List
- Merge k Sorted Lists

3. Traversing : time:

Here we will traverse a linked list form the head to tail visiting every element

Question :

- Convert Binary Number in a Linked List to Integer

Note : never lost the head always move with a separate pointer to keep the structure of linked list and head

4. Reverse a linked list : time:O(n) space :O(1):

This pattern is used very frequently in various singly linked list question. This helps us to reverse the linked list in place or creating a new one reversed

Use full when we have to traverse a linked list in reverse :

Questions :

- Convert Binary Number in a Linked List to Integer
- Reverse Linked List
- Add Two Numbers
- Add Two Numbers II

Variation of reverse linked list :

Here we can reverese an entire linked list but it can be use to reverse a small portion of the linked list :

Questions :

- Palindrome Linked List
- Reverse Linked List II

5. **Intersection of Linked list : time:O(n) space :O(1):**

Here we find intersection of 2 linked list we can use 2 pointers to do in 2 pass

Questions :

- Intersection of Two Linked Lists

6. **Cycle in linked lists :**

Here we have to detect the cycle of the linked list
some times we have to find the length of the cycle in the linked list some times we have to find the starting point of the linked list we can use 2 pointer to solve;

Questions :

- Linked List Cycle
- Linked List Cycle II
- Middle of the Linked List
- Find first node of loop in a linked list

7. **Array based linked lists :**

These questions can be given in array forms or a linked list both approch remain same just traversal changes

Questions :

- Remove Duplicates from Sorted List
- Find All Numbers Disappeared in an Array
- Remove Zero Sum Consecutive Nodes from Linked List leetcode
- Next Greater Node In Linked List
- Find the Minimum and Maximum Number of Nodes Between Critical Points

8. **Change order of original linked list :**

Here we have to change the linked list structure in some way and we cannot create a new linked list

Questions :

- Reorder List
- Remove Nth Node From End of List
- Flatten a Multilevel Doubly Linked List
- Remove Duplicates from Sorted List II
- Merge In Between Linked Lists
- Odd Even Linked List

9. Swapping nodes :

These questions want us to swap node of the linked list means we cannot create a new node or swap just values of the linked list we have to remove and re- insert the nodes to swap

Questions :

- Swapping Nodes in a Linked List
- Swap Nodes in Pairs

10. Group nodes operations :

Here we have to apply a operation on a certain groups of nodes

Questions :

- Rotate List
- Partition list
- Swap Nodes in Pairs
- Split Linked List in Parts
- Reverse Nodes in k-Group

11. Copy Linked List :

Here we create a new exact same linked list but the not a single node is of the original linked list

Questions :

- Copy List with Random Pointer
- Clone a linked list with next and random pointer

12. Copy Linked List ::

These questions are question which use to utilize the property of the linked list how can we use the linked list and why its better then other data structure in these cases

Questions :

- Design Twitter
- Design Linked List
- LRU Cache
- Design Browser History
- LFU Cache
- All O'one Data Structure
- Design Skiplist