perplexity

give me all pattern in linkedlist that is related to one question to anthor

There are several key **Linked List patterns** in LeetCode that often reappear across different problems, with each pattern enabling you to tackle a variety of related questions. Here is a comprehensive list of major patterns and examples tying one question (problem) to another:

1. Fast & Slow Pointers (Tortoise and Hare)

- **Pattern:** Use two pointers, one moving faster than the other, to find cycles, detect palindromes, or locate the middle of the list.
- **Problems:**
  - Detect cycle in a linked list.
  - Find the starting node of a cycle.
  - Check if a linked list is a palindrome.
  - Find the middle node of a linked list[1] [2] [3] [4].

2. Linked List In-place Reversal

- **Pattern:** Reverse the entire linked list or part of it using pointer manipulation—no extra space.
- **Problems:**
  - Reverse a linked list.
  - Reverse nodes in k-group.
  - Reverse a sublist from position m to n.
  - Reorder list in a specific pattern[1] [5] [4].

3. Two Pointer on Two Lists

- **Pattern:** When dealing with two different linked lists, advance pointers to compare, merge, or find intersections.
- **Problems:**
  - Merge two sorted linked lists.
  - Find the intersection node of two linked lists.
  - Add numbers represented by two linked lists[2] [4] [3].

4. Dummy Node Pattern

- **Pattern:** Use a dummy (sentinel) node as a temporary head to simplify edge cases in operations like insertions or deletions.
- **Problems:**
  - Remove nth node from end.
  - Merge lists.
  - Add two numbers (when result requires a new head/dynamic length).
  - Delete duplicates in a sorted list[2] [4].

5. Cycle Detection and Related Problems

- **Pattern:** Identifying cycles and their properties using hashing or runner technique.
- **Problems:**
  - Detect cycle.
  - Find entrance to the cycle.
  - Find length of cycle[4] [1] [3].

6. Ad-hoc/Logical Patterns

- **Pattern:** Custom logic for specialized questions, such as cloning a linked list with random pointers, flattening a linked list, or rotating the list.
- **Problems:**
  - Clone a linked list with random pointers.
  - Flatten a multilevel linked list.
  - Rotate list (left/right by k positions).
  - Remove duplicates, group odd/even nodes[2] [4].

7. Divide-and-Conquer / Merging K Lists

- **Pattern:** Use divide-and-conquer (or a heap/priority queue) to merge multiple sorted linked lists.
- **Problems:**
  - Merge k sorted linked lists[4].

Example Table: Pattern and Shared Logic

Pattern	Problems Connected by Same Pattern
Fast & Slow Pointer	Find middle, find cycle, check palindrome, remove nth
In-place Reversal	Reverse list, reverse k-group, reorder list
Merge/Two Lists	Merge sorted, intersection node, add numbers

Pattern	Problems Connected by Same Pattern
Dummy Node	Remove nodes, add numbers, insertions, deletions
Cycle Problems	Detect cycle, locate cycle start, calculate cycle length
Special/Ad-hoc	Clone with random, rotate, flatten, odd-even grouping

Summary

- Many linked list LeetCode problems can be solved and connected by **advancing common patterns** like two pointers, dummy nodes, and in-place reversal.
- **When learning a pattern, always look for how its technique applies to other linked list problems:** e.g., finding middle vs. finding cycle (both use fast/slow pointers); merging lists vs. intersection (pointers across two lists).
- Mastering these lets you transfer logic seamlessly between different, but related, problems[4] [1] [2] [3].

If you want code templates or question links for a specific pattern, let me know!

⁂

1. https://blog.algomaster.io/p/15-leetcode-patterns
2. https://www.youtube.com/watch?v=bcRe6Qnx40w
3. https://www.architectalgos.com/linked-list-challenges-identifying-and-solving-key-problem-patterns-e2c674de37c2
4. https://github.com/AswinBarath/Linked-lists
5. https://www.geeksforgeeks.org/dsa/top-50-linked-list-interview-question/