Add Two Numbers (Linked List)

Solving the Add Two Numbers (Linked List) problem – a common and fundamental problem involving linked list manipulation and digit wise addition.

Key Concept:

- Each linked list node represents a digit.
- Digits are in reverse order, so the first node is the least significant digit.
- You need to simulate element-wise addition, just like how you'd add numbers manually – keeping track of the carry.

Strategy:

1. Create a dummy node to build the result list.
2. Use a pointer curr to track the current node in the result.
3. Use a variable carry = 0 to hold the carry from each addition.
4. Traverse both lists until both are exhausted and carry is 0:
   - Extract digit from l_1 (if exists), else 0.
   - Extract digit from l_2 (if exists), else 0.
   - Compute: sum = $val_1 + val_2$ + carry
     carry = sum / 10
     digit = sum % 10
   - Create a new node with digit, attach to result.
   - Advance l_1, l_2 and curr.

Time Complexity:

- O(max(m, n)), where m and n are the lengths of the input lists.

Example:

For $l_1 = [2,4,3]$ and $l_2 = [5,6,4]$:
342 + 465 = 807 → return [7,0,8]