Lab Assignment: Rotate a Linked List

Objective

To implement a function in C to rotate a singly linked list to the left by `k` positions.

Problem Statement

Given a singly linked list, rotate the list to the left by `k` places. For example:

$$10 \to 20 \to 30 \to 40 \to 50, \quad k = 4$$

Output:

$$50 \to 10 \to 20 \to 30 \to 40$$

Function Signature

```
struct Node* rotate(struct Node* head, int k);
```

Details

- First, compute the length `n` of the linked list.
- Normalize `k` using `k = k % n`. This ensures we do not perform redundant rotations.
- If `k == 0`, return the original head (no rotation required).
- Traverse `k` nodes from the head.
- The node at position `k` becomes the new head.
- The node just before it (at position `k-1`) becomes the new tail.
- Connect the old tail to the old head.
- Set the new tail's next pointer to `NULL`.

Example Walkthrough

For input:

$$10 \to 20 \to 30 \to 40 \to 50, \quad k = 4$$

1. Length $n = 5$.
2. $k = 4\%5 = 4$.
3. Traverse 4 nodes: New head = node with value 50.
4. New tail = node with value 40.
5. Connect old tail (50) to old head (10).
6. Break the link after new tail (40).
7. Final list:

$$50 \to 10 \to 20 \to 30 \to 40$$

Expected Output

The program should correctly rotate the linked list to the left by `k` places and return the new head pointer.