**Introduction - Doubly Linked List**

[Report Issue](https://github.com/LeetCode-Feedback/LeetCode-Feedback/issues)

We have introduced the singly linked list in previous chapters.

A node in a singly linked list has the value field, and a "next" reference field to link nodes sequentially.

In this article, we will introduce another type of linked list: Doubly Linked List.

*Definition*

The doubly linked list works in a similar way but has one more reference field which is known as the "prev" field. With this extra field, you are able to know the previous node of the current node.

Let's take a look at an example:

A number on a white background

AI-generated content may be incorrect.

The green arrows indicate how our "prev" field works.

*Node Structure*

Here is a typical definition of the node structure in a doubly linked list:

A white background with blue and black dots

AI-generated content may be incorrect.

Similar to the singly linked list, we will use the head node to represent the whole list.

*Operations*

Similar to a singly linked list, we will introduce how to access data, insert a new node or delete an existing node in a doubly linked list.

We can access data in the same exact way as in a singly linked list:

1. We are not able to access a random position in constant time.
2. We have to traverse from the head to get the i-th node we want.
3. The time complexity in the worse case will be O(N), where N is the length of the linked list.

For addition and deletion, it will be a little more complicated since we need to take care of the "prev" field as well. We will go through these two operations in next two articles.

After that, we provide an exercise for you to redesign the linked list using doubly linked list.