Basic Linked List Operations

This lesson lists the various operations that can be performed on linked lists

We'll cover the following

- Operations
- isEmpty()
  - Explanation
  - Time Complexity

Operations

Following are the basic operations of a Singly Linked List:

Operations	Descriptions
`insertAtEnd(data)`	Inserts an element at the end of the linked list
`insertAtHead(data)`	Inserts an element at the start/head of the linked list
`delete(data)`	Deletes an element from the linked list
`deleteAtHead()`	Deletes the first element of the list
`deleteAtEnd()`	Deletes the last element of the list
`Search(data)`	Searches an element from the linked list

Operations	Descriptions
`isEmpty()`	Returns `True` if the linked list is empty, otherwise returns `False`

If you observe the list of methods mentioned above, the last function, `isEmpty()` is a helper function which will prove useful in defining all the other functions.

So let’s define it first.

isEmpty()

The basic condition for our list to be considered empty is that the **head** should be the only pointer in the list. This implies that the head points to `NULL`.

With that in mind, let’s write down the simple implementation for `isEmpty()`:

main.java

SinglyLinkedList.java

```java
public class SinglyLinkedList<T> {
    //Node inner class for SLL
    public class Node {
        public T data;
        public Node nextNode;

    }

    public Node headNode; //head node of the linked list
    public int size;      //size of the linked list

    //Constructor - initializes headNode and size
    public SinglyLinkedList() {
        headNode = null;
        size = 0;
    }

    //Helper Function that checks if List is empty or not
    public boolean isEmpty() {
        if (headNode == null) return true;
        return false;
    }
23  }
```

Explanation

Nothing tricky going on here. The crux of the code lies in the `if` condition on **line 20**. We merely check if the `head` is empty.

> Note: Even when a linked list is empty, the `head` Node must always exist.

Time Complexity

The time complexity of isEmpty() method is $O(1)$.

This is just the tip of the iceberg. We'll tackle each of the methods above in the following lessons and apply them to relevant problems.

In the next lesson, we'll begin our discussion on *linked list* insertion methods.

← Back

What is the Singly Linked List (SLL)?

Next →

Insertion in a Singly Linked List

☑ Mark as Completed

for-coding-interviews-in-java)