LinkedLists

By group 2 😎

Sponsored by: Sanrio

Jaylen Byrd, Patrick Nelson, Pablo Hernandez, Robin Kuhn, Kemberly Montina, Hector Rios

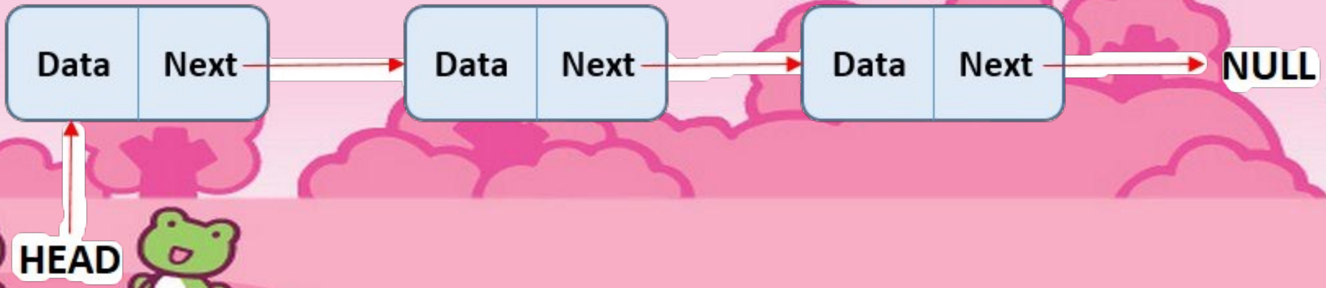
What is a LinkedList?

- A collection of elements, which are usually of the same type, and it stores its items within nodes.
- Each node is linked to one another.
- Linear Data Structure

Different ways to implement:

- There are three ways we can implement LinkedLists:

1. The first is a Singly Linked list where the node “points” to the next node in the list.
2. The second is a Doubly Linked list where the node “points” to both the next node and the previous node.
3. The third is a Circular Linked list where the last node “points” to the first node.



What Are LinkedLists Useful For?

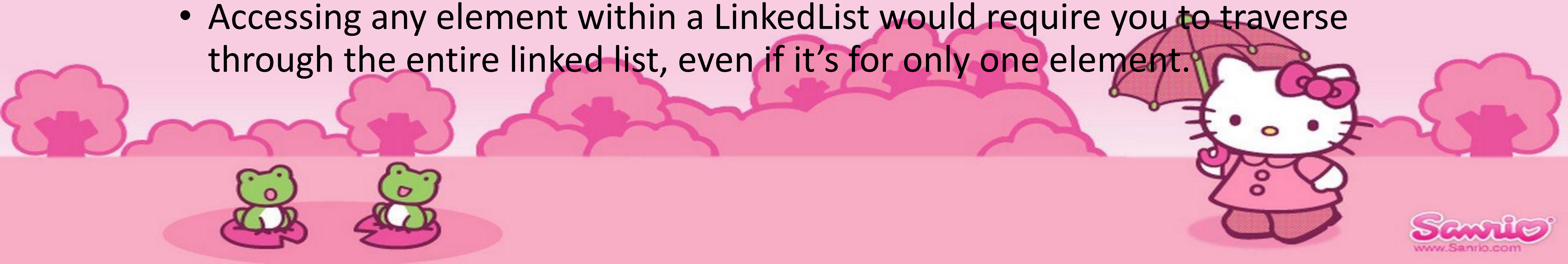
- The manipulation of data.
- Able to grow and shrink at runtime—mutable.
- Memory management between the element and its key.

Where Should We Use LinkedLists

- Useful where modifications to a collection of data are going to be made frequently
  - Addition/Subtraction of elements
  - LinkedLists are faster in this regard
- When you need to constantly add elements past a list's initialization
  - Size of the list automatically changes as you add and remove items.

Where Is It Not Useful?

- When trying to access data directly
  - Must start from the "Head" and follow through the link to reach a desired node
- When you're tight on memory
  - LinkedLists take up more memory because they allocate for not only the elements inside each node, but also the address neighboring nodes
- Accessing specific elements
  - Accessing any element within a LinkedList would require you to traverse through the entire linked list, even if it's for only one element.



Implementation!

Pls direct ur attention to Jaylen and Patrick