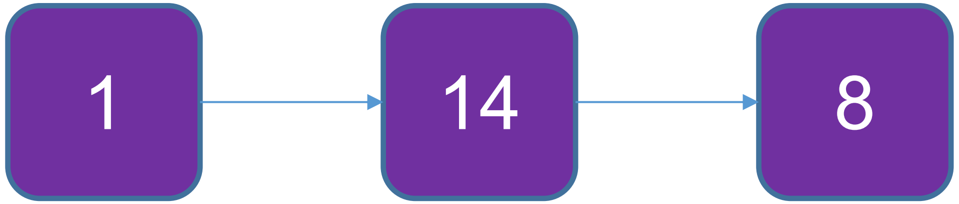
Linked Lists

18th August, 2022

ANJALI
VIRAMGAMA

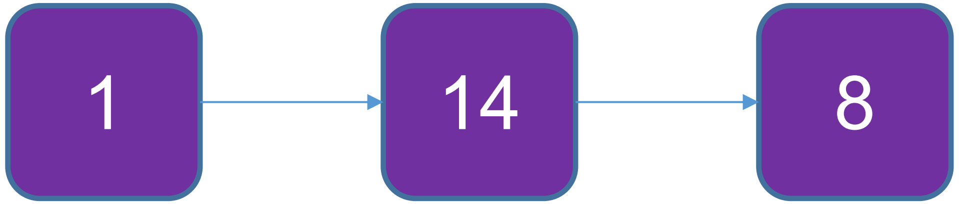
What is a LinkedList?

- A collection of **nodes**, where each node consists of data and next pointer.
- The first node in a linked list is called the *head* node.
- The final node in a linked list, called the *tail*, points to null.



Linked Lists vs Arrays

- Linked Lists are dynamically sized, while the size of an array cannot be changed as easily
- Accessing the **k**th element in an array takes constant time, while it takes **k** time for a linked list
- Both take up O(n) space to store n elements.
- Linked Lists take slightly more space, however, as they need to store pointers to each element



Declaring a Linked List - Python

Declaration

```
class Node:
    def __init__(self, dataval=None):
        self.dataval = dataval
        self.nextval = None
```

Implementation

```
my_linked_list = Node(“first element”)
my_linked_list.next = Node(“second element”)
my_linked_list.next.next = Node(“third element”)
```

Declaring a Linked List - Java

Declaration

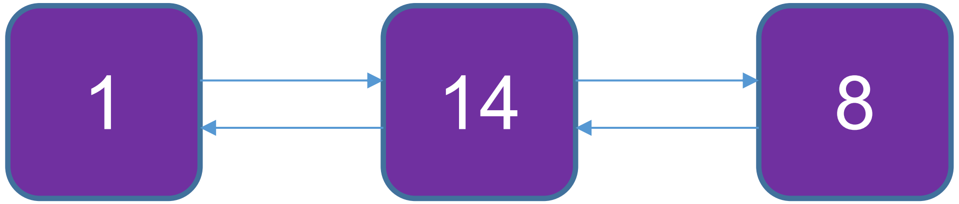
```
class LinkedList {
    Node head; // head of the list
    /* Linked list Node*/
    class Node {
        int data;
        Node next;
        // Constructor to create a new node, leave
      next as null
        Node(int d) { data = d; }
    }
}
```

Implementation

```
/* Start with the empty list. */
LinkedList list = new LinkedList();
 list.head = new Node(1);
```

Types of LinkedLists

- Singly LinkedList
- Doubly LinkedList (has prev and next)
- Circular LinkedList (all nodes are connected to form a circle
- Some LinkedLists have tail node. Useful to append to the list in O(1) time
- Some lists have partial cycles and could result in an endless loop if we are not careful.



Time Complexity Table

	Singly Linked	Doubly Linked
Search		
Insertion to Sorted		
Insertion to Unsorted		
Deletion		

Time Complexity Table

	Singly Linked	Doubly Linked
Search	O(n)	O(n)
Insertion to Sorted	O(n)	O(n)
Insertion to Unsorted	O(1)	O(1)
Deletion	O(n)	O(n)*

*O(1) with reference to node

In class Assignment

Easy: Merge 2 sorted Lists

Easy: Linked List Cycle

Easy: Reverse Linked List

Medium: Add 2 numbers

Medium: Copy List with Random Pointer

Hard: Merge k sorted Lists

Homework

Easy: Remove Duplicates from Sorted List

Medium: Reverse Linked List 2.

Medium: Remove Nth from end of list

Medium: Rotate List

Medium: Swap nodes in pairs

Medium: LinkedList Cycle 2

Disclaimer

All content and material is copyrighted material belonging to Anjali Viramgama and is purely for the dissemination of education. You are permitted to access print and download extracts from this site purely for your own education only and on the following basis:-

- You can download this document from the provided link for self use only.
- Any copies of this document, in part or full, saved to disc or to any other storage medium may only be used for subsequent, self viewing purposes or to print an individual extract or copy for non commercial personal use only.
- Any further dissemination, distribution, reproduction, copying of the content of the document herein or the uploading thereof on other websites or use of content for any other commercial/unauthorized purposes in any way which could infringe the intellectual property rights of its contributors, is strictly prohibited.
- No graphics, images or photographs from any accompanying text in this document will be used separately for unauthorised purposes.
- No material in this document will be modified, adapted or altered in any way.
- No part of this document may be reproduced or stored in any other web site or included in any public or private electronic retrieval system or service the contributors' prior written permission.
- Any rights not expressly granted in these terms are reserved.