upGrad

TechM Full-Stack Software Development

upGrad

Lecture On: Singly Linked list

Instructor: Arkoprovo Dey

In Last Class, we covered….

- Basic Sorting Algorithms

Homework Discussion

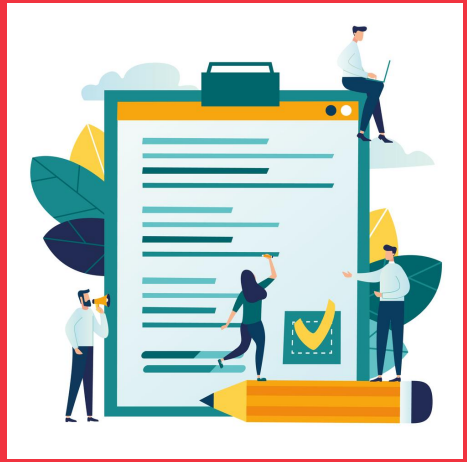
1. Sort an array such that the first element is largest and then the smallest element. It has to be followed by second largest and second smallest element in array.

   e.g. [6,1,4,2,9,3] -> [9,1,6,2,4,3]

Today's Agenda

1 Introduction to Linked List

2 Common problems on singly Linked List



Poll 1 (15 Sec.)

An array can be used in the fragmented memory space.

1. True
2. False

Poll 1 (Answer)

An array can be used in the fragmented memory space.

1. True
2. **False**

Poll 2 (15 Sec.)

When should we start to look beyond an array data structure?

1. When we need very fast access to elements
2. When we need dynamic sizing
3. When we need to store data in an ordered way
4. None of these

Poll 2 (Answer)

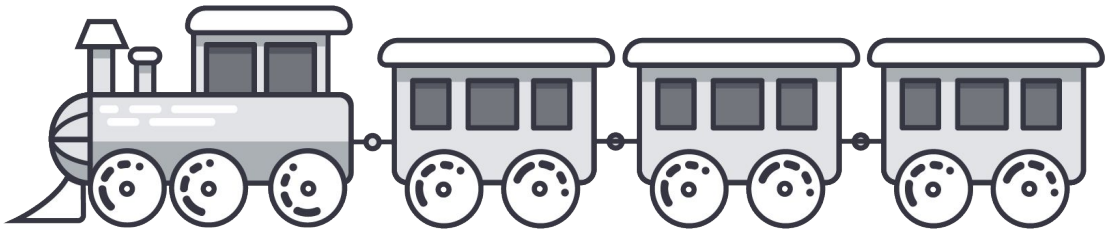
When should we start to look beyond an array data structure?

1. When we need very fast access to elements
2. **When we need dynamic sizing**
3. When we need to store data in an ordered way
4. None of these

Linked List

Now, we know how arrays, a linear data structure, works. But if we want a kind of data structure where the size of the list is not fixed at the beginning, we will need to be a bit more innovative.

Let's see how trains work today!

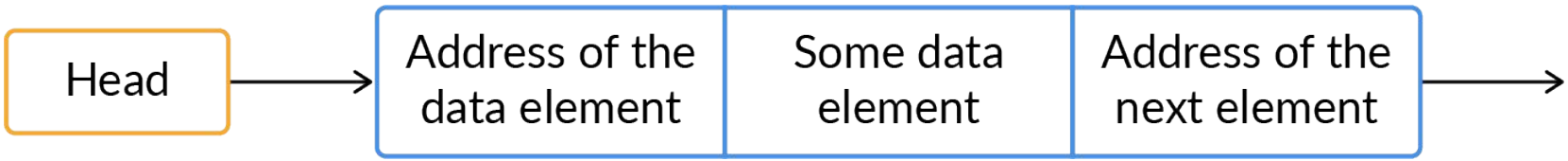


🤔 What if we were able to model a data structure along similar lines?

Linked List

A linked list can be thought of as a sequence of data elements joined together with a 'link'. Each element of this list (similar to a bogie in a train) consists of the data we want to store in the list, and a 'pointer' to the next element in the list.

To demarcate the start of a Linked List, we can have an element (of the same type as that of the other data elements) whose pointer points to the starting node of the linked list. This element is called the **'head'** and can be thought of as the engine of the train.



Linked List

We can see a stark difference in the implementation of an array. The memory address of the subsequent data elements do not need to be contiguous. The pointer will contain the address to wherever the data is stored in the system memory.

To implement a node in Java, we define a class, traditionally called a ‘Node’. This class contains data members corresponding to the data we want in our Linked List. Along with these, it contains another data member of the type ‘Node’ that contains the reference to the next list item.

The ‘pointer’ in the last element of the list contains a ‘null’ value.

Poll 3 (15 Sec.)

Which of the following is true about linked list?

1. It supports dynamic sizing
2. It can be used even with fragmented memory space
3. Adding a new data node is very easy
4. All of these

Poll 3 (Answer)

Which of the following is true about linked list?

1. It supports dynamic sizing
2. It can be used even with fragmented memory space
3. Adding a new data node is very easy
4. **All of these**

Linked List

We can manipulate Linked Lists in whichever way we want. We should always know how to perform a few of the following basic operations on a Linked List:

- Creating a node in a Linked List (at the beginning, end, or anywhere in the middle)
- Deleting a node (at the beginning, end, or anywhere in the middle)
- Searching for a node

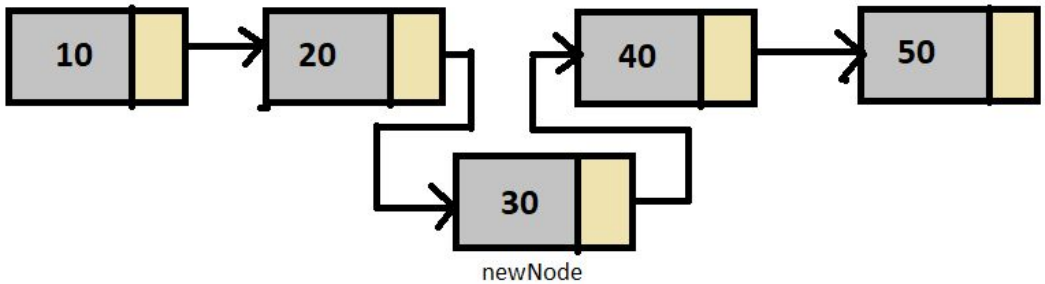
Linked List

Let's look at some pseudo code for creating a node at the beginning of a Singly Linked List with string data 'upGrad'.

```
1: begin
2: create newNode
3: newNode.data = "upGrad"
/* points to the previous first node */
4: newNode = next.head
5: head = newNode    /*updating the head */
6: end
```

Linked List

Similarly, we can also insert some data at a specific position in the middle of the list.



Poll 4 (15 Sec.)

Which of the following is true about singly linked list?

1. Can traverse in both directions
2. Can traverse in a single direction
3. Can get direct access to the random node
4. None of the above

Poll 4 (Answer)

Which of the following is true about singly linked list?

1. Can traverse in both directions
2. **Can traverse in a single direction**
3. Can get direct access to the random node
4. None of the above

Poll 5 (15 Sec.)

Which of the following information is contained in each node of a linked list?

1. Data
2. Pointer to the next node
3. Both 1 and 2
4. Neither 1 nor 2

Poll 5 (Answer)

Which of the following information is contained in each node of a linked list?

1. Data
2. Pointer to the next node
3. **Both 1 and 2**
4. Neither 1 nor 2

Hands-on Coding

- Now let us see how we can perform some of the operations on linked list in Java:
  - Traverse the linked list
  - Find the length of singly linked list
  - Search for a node in LL
  - Modify/update value of a node
  - Find the middle element of LL
  - Return the nth element

Poll 6 (15 Sec.)

What is the worst case time complexity of traversing a singly linked list?

1. O(n)
2. O(nlogn)
3. O(1)
4. $O(n^2)$

Poll 6 (Answer)

What is the worst case time complexity of traversing a singly linked list?

1. **O(n)**
2. O(nlogn)
3. O(1)
4. $O(n^2)$

Poll 7 (15 Sec.)

What is the time complexity of searching an element in linked list at the k^{th} index?

1. O(1)
2. O(n)
3. $O(n^2)$
4. $O(n^3)$

Poll 7 (Answer)

What is the time complexity of searching an element in linked list at the k^{th} index?

1. O(1)
2. **O(n)**
3. $O(n^2)$
4. $O(n^3)$

Poll 8 (15 Sec.)

In the famously fast and slow pointer approach to finding the midpoint of the linked list, how many times do we need to traverse the linked list?

1. 1
2. 2
3. 3
4. 4

Poll 8 (Answer)

In the famously fast and slow pointer approach to finding the midpoint of the linked list, how many times do we need to traverse the linked list?

1. **1**
2. 2
3. 3
4. 4

Hands-on Coding

- Some more practice on LL:
  - Insert a new node at the start of LL
  - Insert a new node at the end of LL
  - Insert a new node in the middle of LL
  - Delete first node
  - Delete last node
  - Delete a node somewhere in between

Poll 9 (15 Sec.)

What is the worst case time complexity of inserting a new node at the beginning of the singly linked list?

1. O(n)
2. O(nlogn)
3. O(1)
4. $O(n^2)$

Poll 9 (Answer)

What is the worst case time complexity of inserting a new node at the beginning of the singly linked list?

1. O(n)
2. O(nlogn)
3. **O(1)**
4. $O(n^2)$

Poll 10(15 Sec.)

What is the worst case time complexity of deleting a node at k^{th} position in singly linked list?

1. O(n)
2. O(nlogn)
3. O(1)
4. $O(n^2)$

Poll 10(Answer)

What is the worst case time complexity of deleting a node at k^{th} position in singly linked list?

1. **O(n)**
2. O(nlogn)
3. O(1)
4. $O(n^2)$

Coding Practice

- Reverse a singly LL using iterative method.
- Sort a singly linked list.
- Detect loop in a LL.
- Find starting point of the loop in LL.

Homework

1. Remove loop in a LL.

2. Check for duplicate nodes in LL and delete if any.

upGrad

Tasks to complete after the session

Homework Questions
MCQs
Coding Questions

In the next class...

- Doubly Linked List
- Circular Linked List

upGrad

#RahoAmbitious

Thank You!