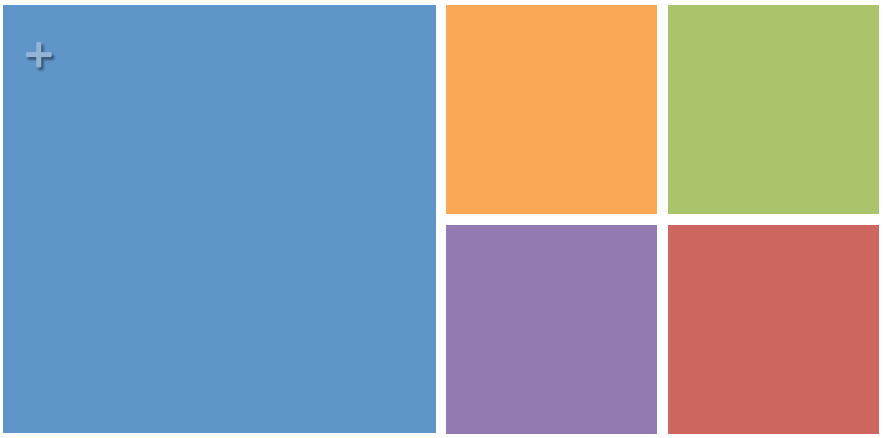
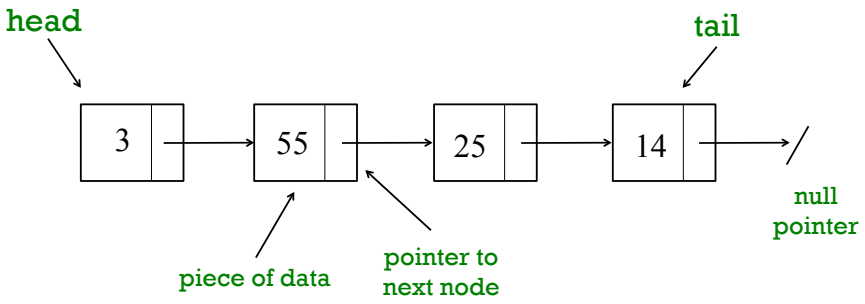


Lecture 11: More Linked Lists

Today

- Reading
  - JS Ch. 9 (Linked Lists)
- Objectives
  - Operations on singly linked lists
  - (Doubly linked list)

Singly Linked List



Node Class

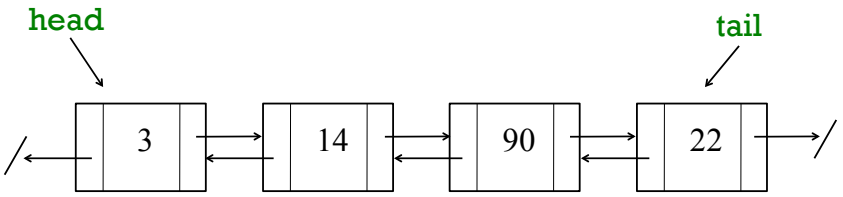
```
public class Node<E> {
      E data;           // piece of data
      Node<E> next;     // ptr. to next node in list

      public Node(E data, Node<E> next) {
            this.data = data;
            this.next = next;
      }

      public Node(E data) {
            this(data, null);
      }

      public void setNext( Node<E> next){ this.next = next; }

      public void setData( E data) { this.data = data; }

      public E data() { return data; }

      public Node<E> next() { return next; }

}
```

Singly Linked List Operations

- public void addFirst(E value)
- public E removeFirst()
- public void addLast(E value)
- public E removeLast()
- public E remove(E value)
- public boolean contains(E value)
- public E get(int i)
- public E set(int i, E value)

What is the worst case complexity of each?

Doubly Linked List



Doubly Linked List

- Useful if you need to traverse in both directions
- Must change twice as many links when adding or removing!
- Node<E> class now contains pointer to previous node
- java.util.LinkedList provides a doubly linked list
- We'll use DoublyLinkedList from Bailey