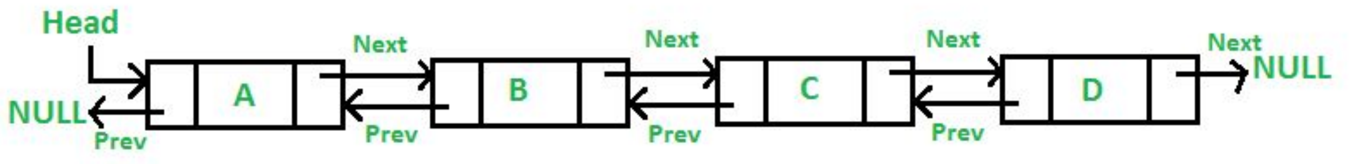
Linked Lists

Topics to cover: Insertion, Searching, Common Mistakes, All of this for doubly linked lists. Exercise at the end

What is a Linked List?

- A linked list is a sequence of data structures, which are connected via links
- Linked lists are the second most-used data structure after arrays
- Important terms
  - **Node** - Each node of a LL stores data and pointers to adjacent nodes
  - **Links**
    - **Singly** linked lists, have 1 link that points to the next node's memory address.
    - **Doubly** linked lists have 2 links, one forwards and one backwards
  - **Linked List** - To access a LL you only remember the "head" node, from there you traverse through the list linearly. The head node's previous link is NULL, as is the last node's next



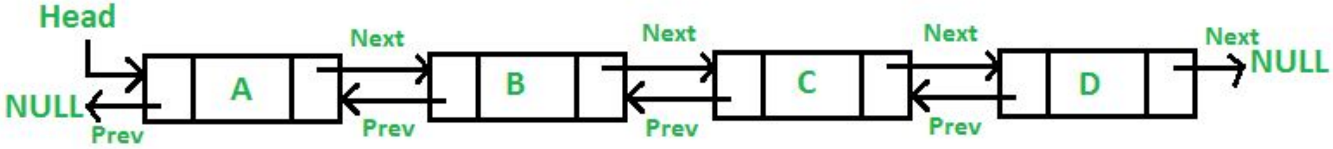
One Node

```
struct exampleNode{
        char key;
        exampleNode *next;
        exampleNode *prev;
} ;
```

Searching

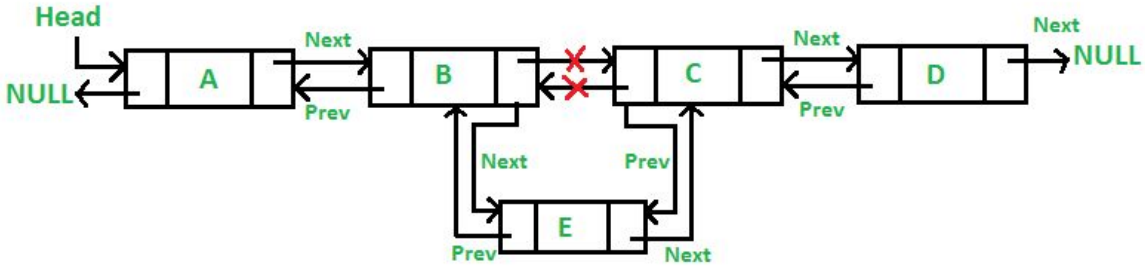
- exampleNode* head = A;
- How do you return the node with key == ‘C’?
- How do you return the node with key == ‘A’?
- What do you do when asked for key == ‘Z’?

```
exampleNode* temp = head;
while(temp.next != null){
        if(temp.key == ‘ourkey’)
                return temp;
        else
                temp = temp.next;
}
return error;
```



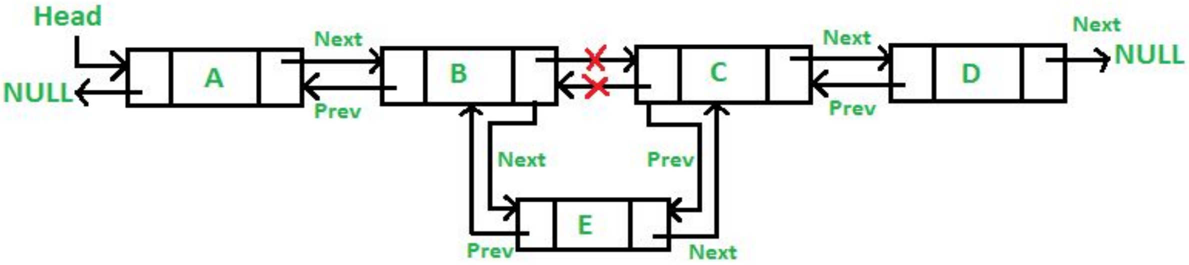
Insert in Doubly Linked List:

1. Create the node we're inserting (key = E)
2. Starting from head, loop through until we are at the B node because we're inserting after that
3. B.next is currently C. C.prev is currently B. We need to ensure when we add node E we do not end up with nodes pointing to nothing



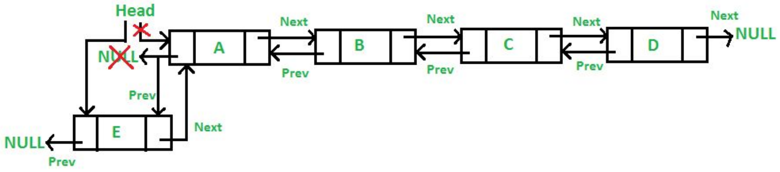
Insert in Doubly Linked List:

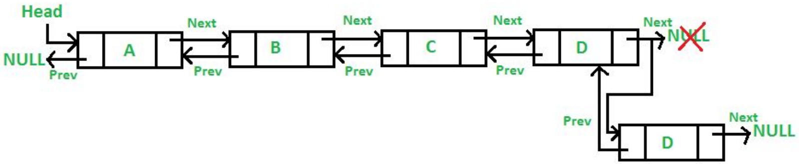
4. Set E.next equal to B.next
5. Set B.next equal to E
6. Set E.prev equal to C.prev
7. Check if E.next is null. If not then set E.next.prev equal to E
8. Set C.prev equal to E



Questions for the class

- How would inserting a node before the head work?
- How would inserting a node at the tail work?


Head
Next
Next
Next
Next
NULL
A
B
C
D
NULL
Prev
Prev
Prev
Prev
E
Next
NULL
Prev


Head
Next
Next
Next
Next
NULL
NULL
A
B
C
D
Prev
Prev
Prev
Prev
Prev
Next
D
NULL