Lecture 1
September 16

Binary search
Linked lists
Two sum problem

BinarySearch

Different algorithm techniques:

1. Brute force
2. Divide and conquer (best)
3. Greedy
4. Backtracking

It’s a divide and conquer approach.

Recursive vs iterative version.

How to approach a coding problem?

---

1. Draw an example.
2. Before writing every line of code, test your code on that example.

Coding is a precise business

Be precise:

a. With the boolean conditions.
b. With the start and end of your for loops.
c. With the boundary conditions in your recursive calls

Be reproducible:

Write the same code every time, use the same variable names, same function names. Repeat small patterns.

Sample problems using binary search

1. Finding the pivot point in a sorted rotated array
2. Finding the start and end point of a repeated number in a sorted array.

Linked lists: Finding a node

Linked lists: Deleting a node

Linked lists: Reversing a linked list

Obviously, need to remember this algorithm.

Need to remember the algorithm, not the code.

Take 3 pointers: previous, current, next.

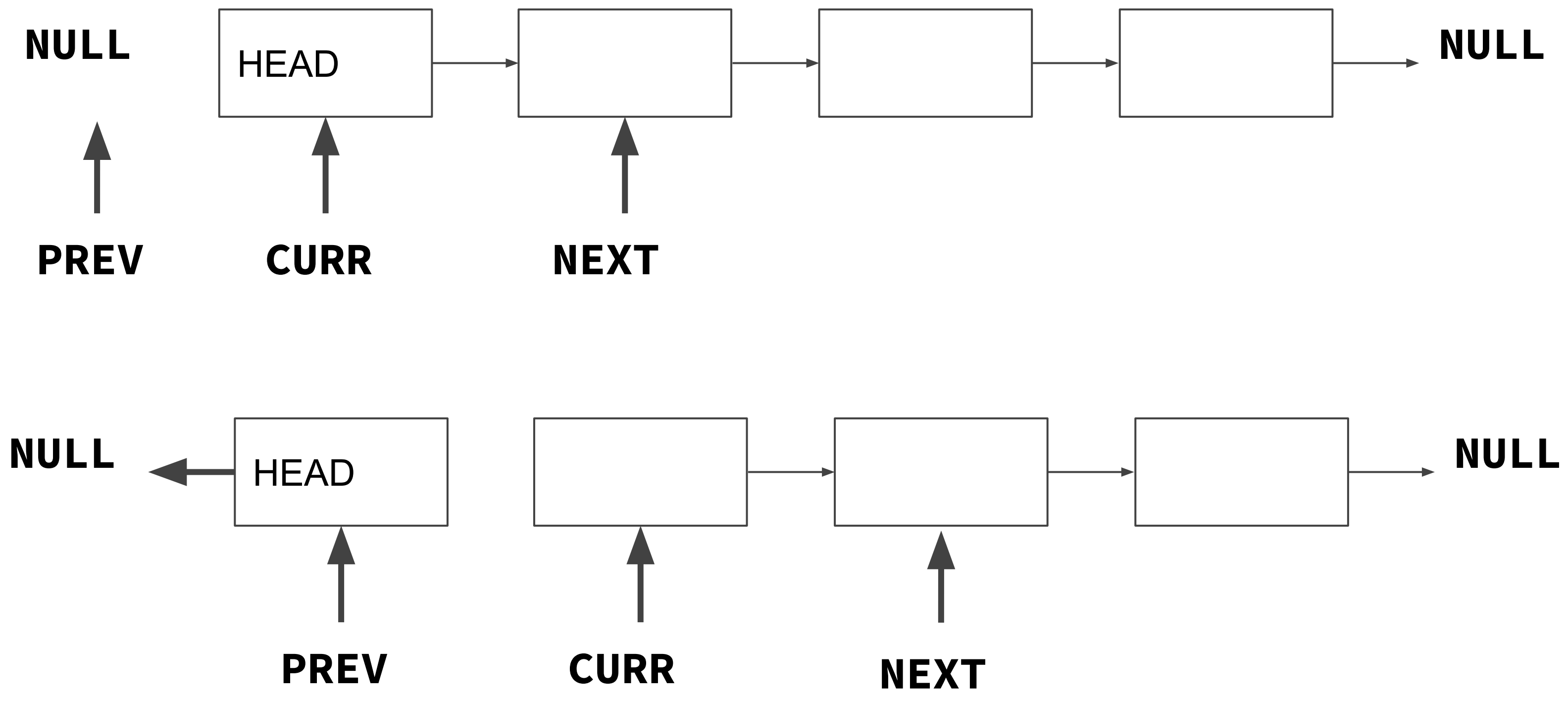
Traverse the linked list node by node.

Reverse the node->next pointer for every node.

Return the new head.

Obviously: Test every line of code, be aware of boundary conditions.

Linked lists: Reversing a linked list

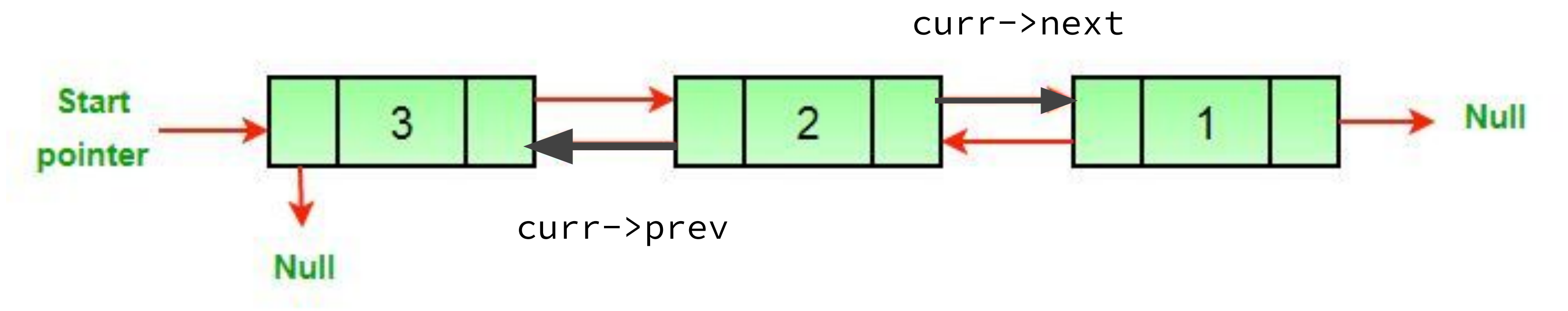


```
How did curr move to the next node when the link was
broken?
```

Linked lists: Reversing a doubly linked list.

Home work.

Give overview in the class.

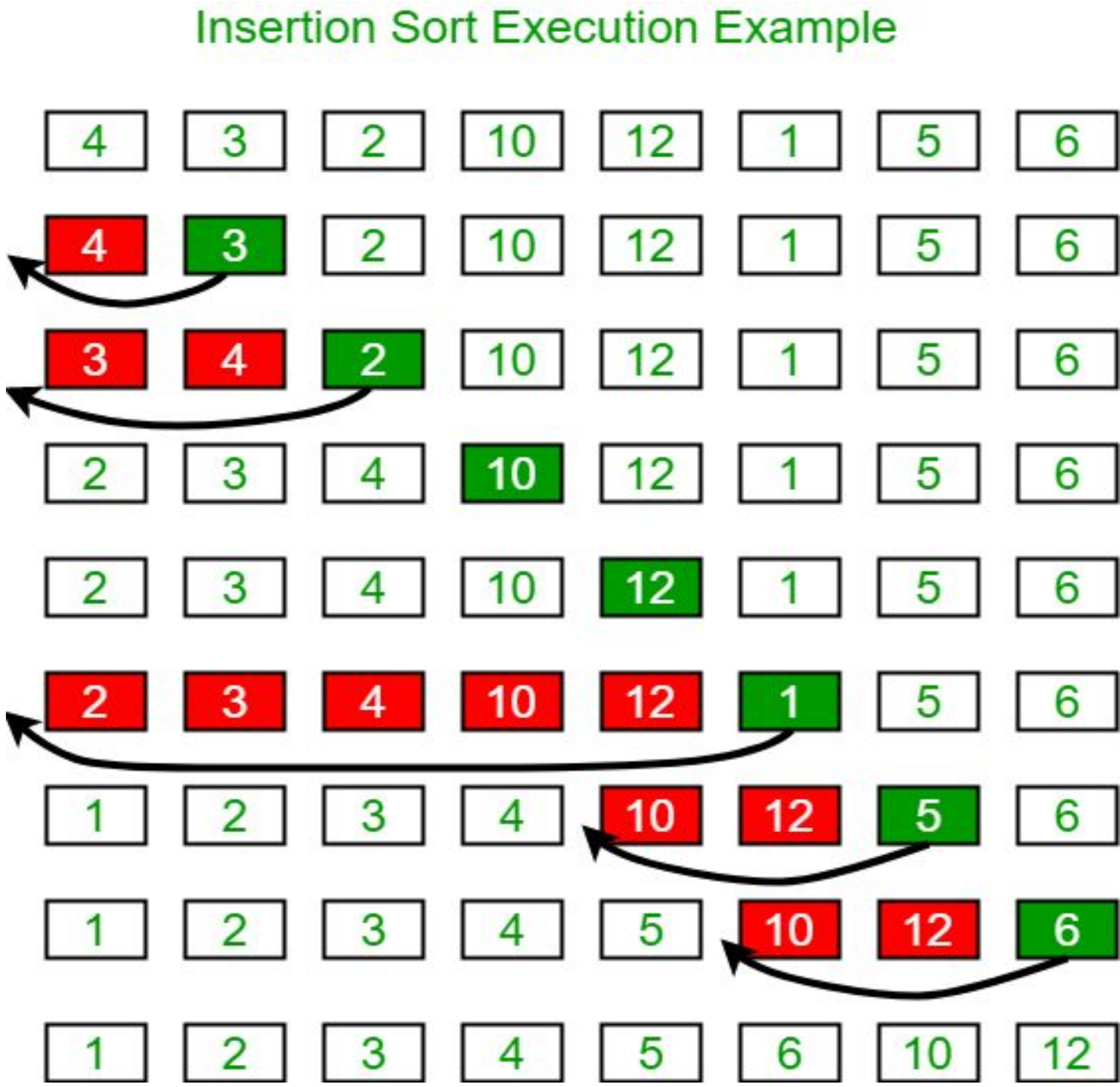


Array traversal: Two pointer

Example: 2 sum problem

In a sorted array, find two numbers that sum to a given number.

Array traversal: Insertion sort


Insertion Sort Execution Example
4 3 2 10 12 1 5 6
4 3 2 10 12 1 5 6
3 4 2 10 12 1 5 6
2 3 4 10 12 1 5 6
2 3 4 10 12 1 5 6
2 3 4 10 12 1 5 6
1 2 3 4 10 12 5 6
1 2 3 4 5 10 12 6
1 2 3 4 5 6 10 12