Competitive Programming Club

Prep Hour, Week 1

Linked Lists, Intervals

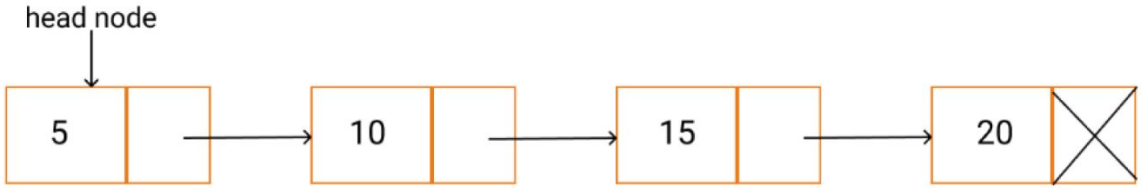
What & Why is Prep Hour?

- Organized lessons are a great way to practice
- Cover 1-2 topics/week
- Based on neetcode.io blind 75 problem set

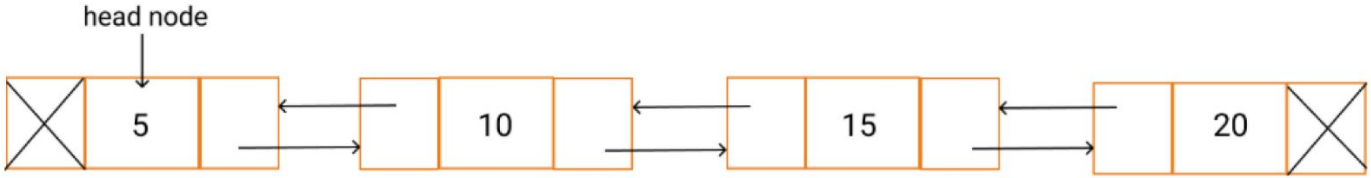
Linked Lists

- Nodes connected by pointers
- To access a previous node, you must maintain a pointer to it
- Practice problems involve accessing & modifying linked lists in interesting ways
- More on Linked Lists

Singly Linked List



Doubly Linked List



Intervals

- Pairs of start and end times/distances/whatever
- Practice problems revolve around finding overlap between intervals
- Mini-example: adding a new interval & merging to prevent overlap

Input: `intervals = [[1,3],[6,9]]`
`newInterval = [2,5]`

Output: `[[1,5],[6,9]]`

- *Because [1, 3] and [2, 5] overlap, they've been merged*
- *The other intervals remain unchanged*
- *More on Intervals*

Advanced Concept: Greedy Algorithms

- **Greedy Algorithms** are repetitive algorithms that pick the most attractive choice in a small scope at each step
- Problems that can be solved with greedy algorithms have **these traits**:
  - **Problem can be broken down into subproblems**, where a local optimal choice can be made at each step with the aim of finding a global optimum.
  - **Greedy Choice Property**: Determine if making a locally optimal choice at each step leads towards a global optimum solution. Essentially, you should be able to make a choice that seems best at the moment and follow that path without regretting past decisions.
  - **Optimal Substructure:** Verify if an optimal solution to the problem can be constructed from optimal solutions of its subproblems. This means that the solution to the problem includes solutions to subproblems.
  - **No Revisiting:** Once a choice is made, the algorithm never reconsiders it. If the problem requires going back and changing decisions, a greedy algorithm might not be appropriate.