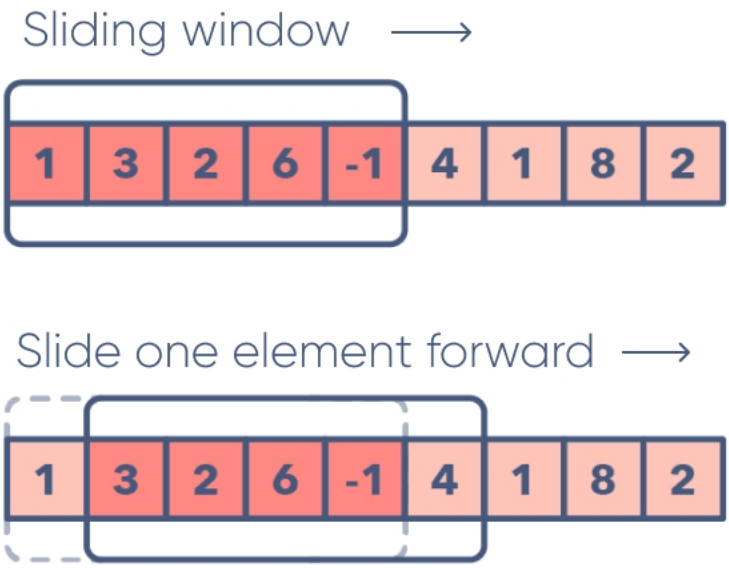
Swipe >>>

Algorithm Pattern

Coding Pattern Questions

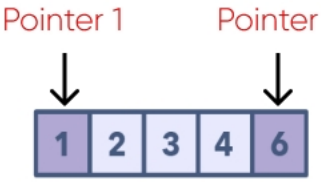
#1. Sliding Window



This pattern is used to find max/min k-subarray, XOR, product, sum, etc.In some cases, the window size remains constant and in other cases the sizes grows or shrinks.

Coding Pattern Questions

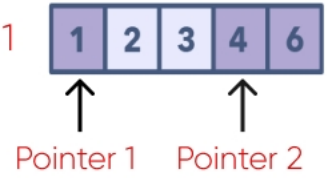
#2. Two Pointers or Iterators

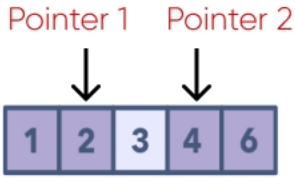


Target sum = 6

1+6 > target sum, therefore let's decrement Pointer 2

1+4 < target sum, therefore let's increment Pointer 1



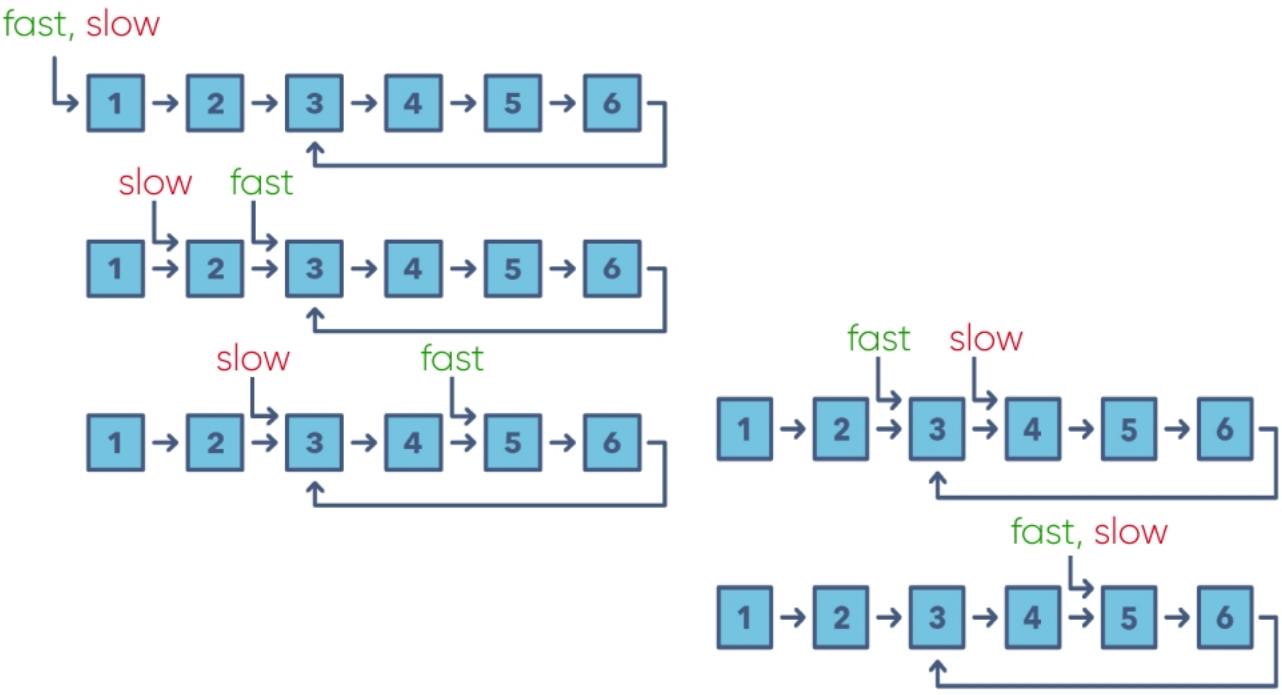


2+4 == target sum, we have found our pair!

It is a pattern where two pointers iterate through the data structure until one or both of the pointers hit a certain condition.It is often useful when searching pairs in a sorted array or linked list.

Coding Pattern Questions

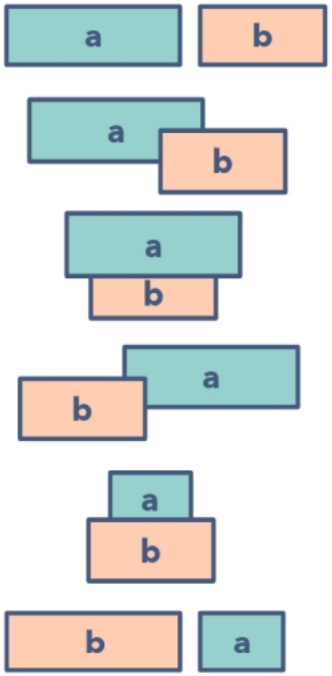
#3. Fast and Slow pointers



It is a pointer algorithm that uses two pointers which move through the array (or sequence/linked list) at different speeds. This approach is quite useful when dealing with cyclic linked lists or arrays.

Coding Pattern Questions

#4. Merge Intervals



'a' and 'b' do not overlap

'a' and 'b' overlap, 'b' ends after a

'a' completely overlaps 'b'

'a' and 'b' overlap, 'a' ends after b

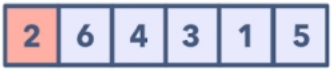
'b' completely overlaps 'a'

'a' and 'b' do not overlap

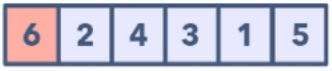
It is an efficient technique to deal with overlapping intervals. In a lot of problems involving intervals, you either need to find overlapping intervals or merge intervals if they overlap.

Coding Pattern Questions

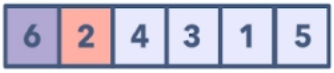
#5. Cyclic sort



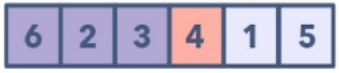
Number '2' is not at it's correct place, let's swap it with the correct index



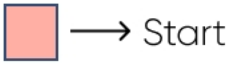
After the swap, number '2' is placed at it's correct index



Number '4' is not at it's correct place, let's swap it with the correct index



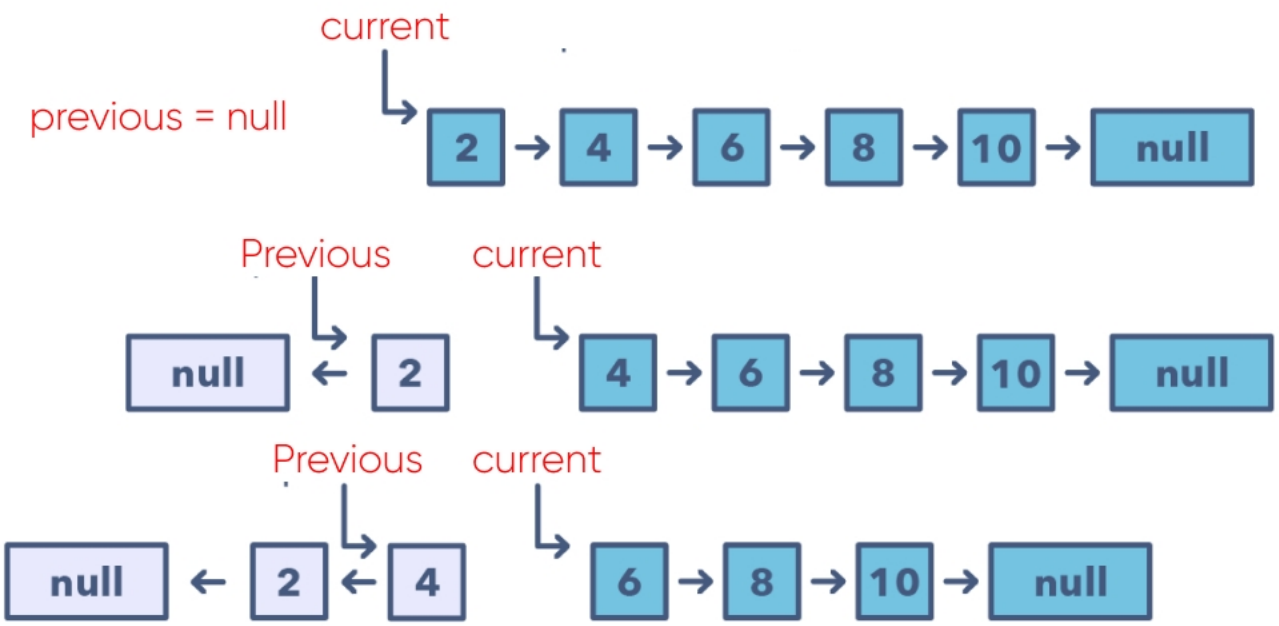
Number '4' is at it's correct place,



It iterates over the array one number at a time, and if the current number you are iterating is not at the correct index, you swap it with the number at its correct index.

Problem Solving Patterns

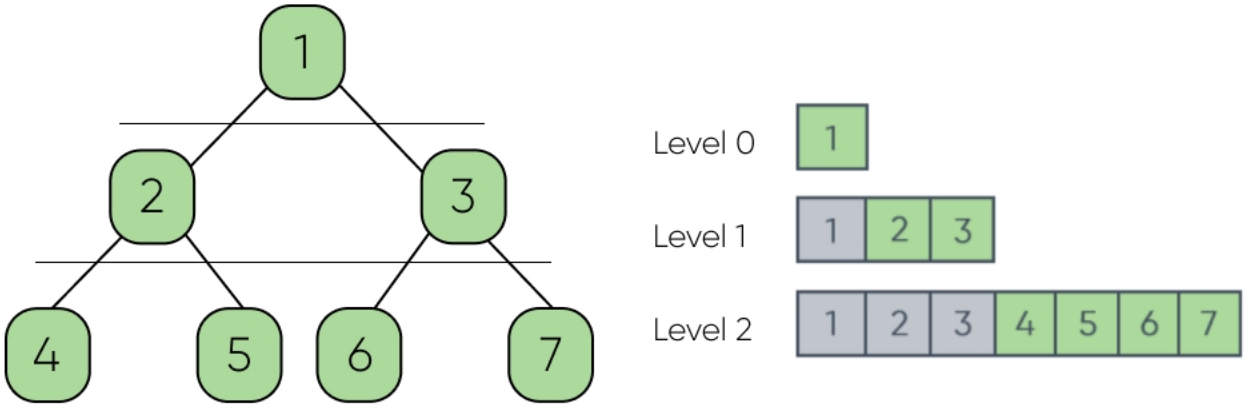
#6. In-place reversal of linked list



This pattern reverses one node at a time starting with one variable pointing to the head of the linked list, and one variable will point to the previous node that you have processed.

Problem Solving Patterns

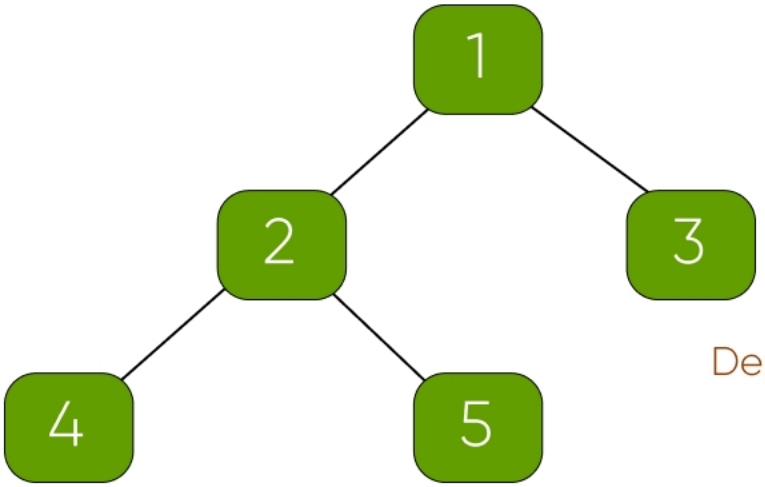
#7. Tree BFS



This pattern is used to traverse a tree and uses a queue to keep track of all the nodes of a level before jumping onto the next level.

Problem Solving Patterns

#8. Tree DFS

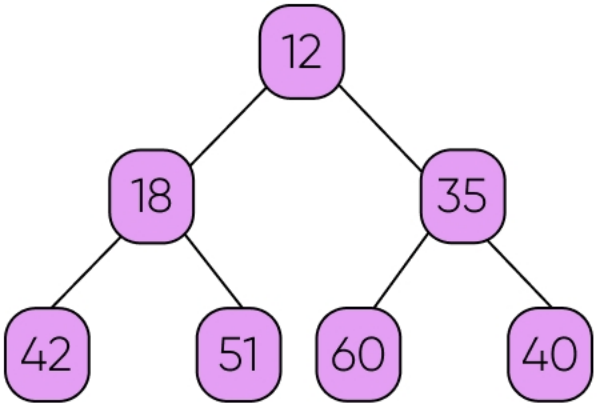


Depth First Traversals:
Preorder Traversal : 1 2 4 5 3
Inorder Traversal : 4 2 5 1 3
Postorder Traversal : 4 5 2 3 1

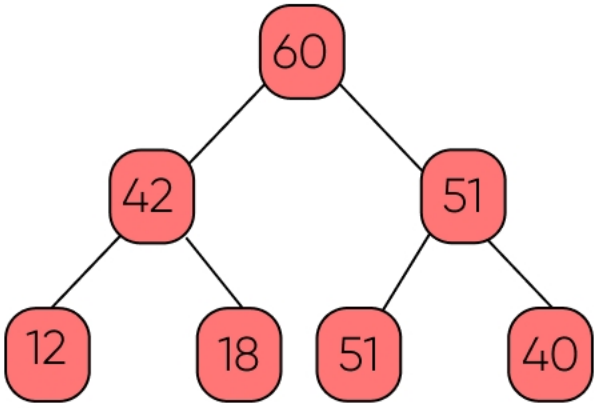
This pattern works by starting at the root of the tree, if the node is not a leaf you need to do three things : Preorder Traversal, Inorder Traversal, Postorder Traversal.

Problem Solving Patterns

#9. Two heaps



MIN HEAP

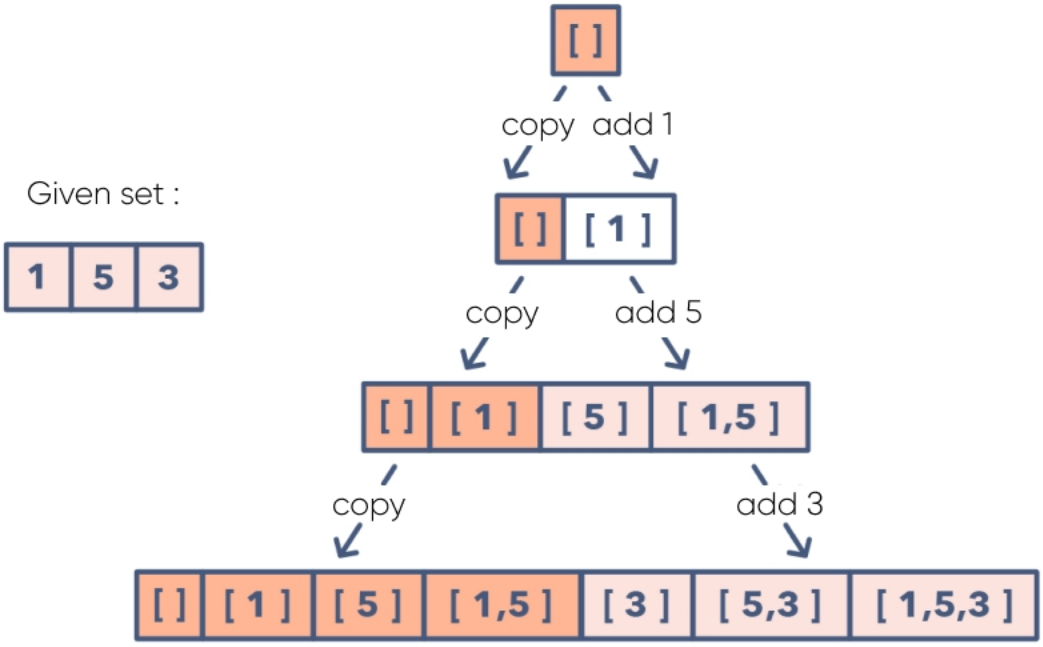


MAX HEAP

This pattern uses two heaps; A Min Heap to find the smallest element and a Max Heap to find the biggest element.

Problem Solving Patterns

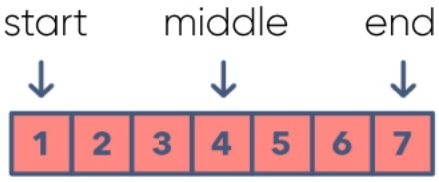
#10. Subsets



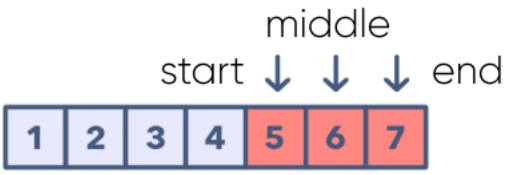
The pattern Subsets describes an efficient Breadth First Search (BFS) approach to handle all these problems.

Problem Solving Patterns

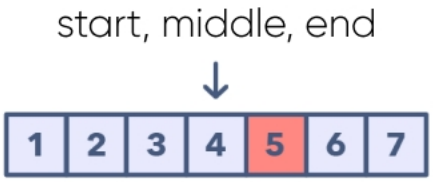
#11. Modified binary search



A **key > arr[middle]**, therefore **start = middle +1**



A **key < arr[middle]**, therefore **end = middle - 1**

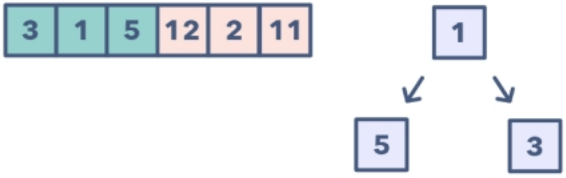


A **key == arr[middle]**, return **middle**

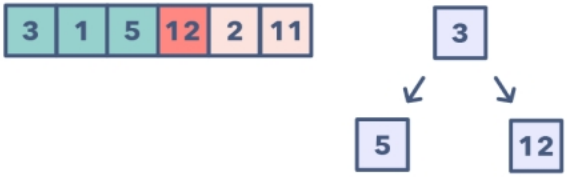
Whenever you are given a sorted array, linked list, or matrix, and are asked to find a certain element, the best algorithm you can use is the Binary Search.

Problem Solving Patterns

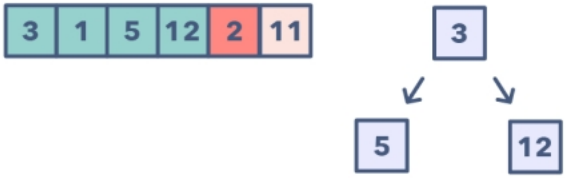
#12. Top K elements



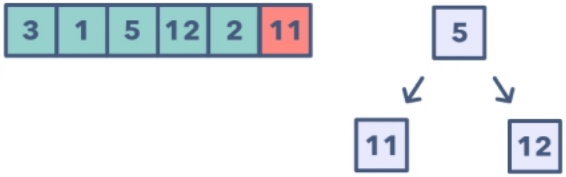
Insert the first three numbers in the heap



The root is smaller than '12' so take '1' out and insert '12'



Skip '2', as it is not bigger than the root '3'

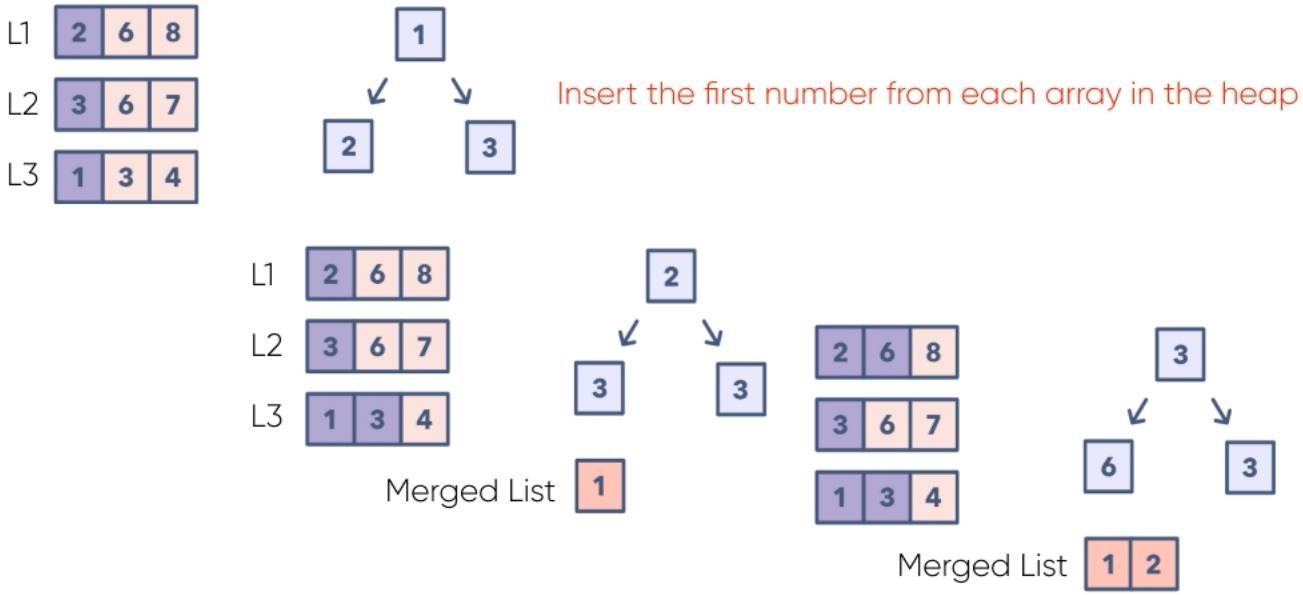


The root is smaller than '12' so take '5' out and insert '12'

Any problem that asks us to find the top/smallest/frequent 'K' elements among a given set falls under this pattern.

Problem Solving Patterns

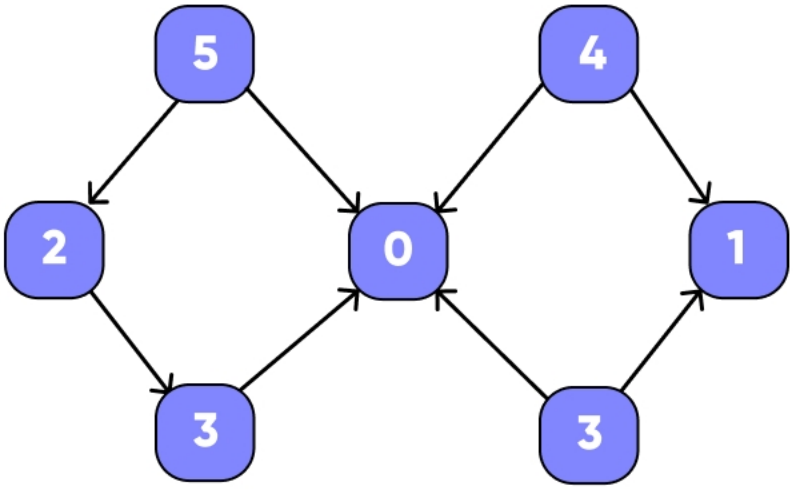
#13. K-way Merge



It helps you solve problems that involve a set of sorted arrays. Push the smallest element of each array in a Min Heap to get the overall minimum then push the next element from the same array to the heap. Then, repeat this process to make a sorted traversal of all elements.

Problem Solving Patterns

#14. Topological sort



Topological order :
1, 2, 5, 4, 3, 6, 7
2, 1, 5, 4, 7, 3, 6
2, 5, 1, 4, 7, 3, 6
Etc

- There are often many possible topological sorts of a given DAG
- Topological sorting for Directed Acyclic Graph (DAG) is a linear ordering of vertices such that for every directed edge u v, vertex u comes before v in the ordering.

cs mock

About us :

CS Mock aims to smoothen the placement journey of college graduates by polishing their skills, highlighting the points of improvement and boosting their confidence. We provide 1-on-1 live mock interview and mentorship with industry professionals who are part of hiring team.

+91 9956127183

support@csmock.com

www.csmock.com