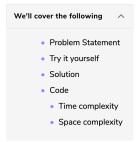


Grokking the Coding Interview: Patterns for **Coding Questions** 67% completed Q Search Course Pattern: Modified Binary Search Introduction Order-agnostic Binary Search (easy) Ceiling of a Number (medium) Next Letter (medium) Number Range (medium) Search in a Sorted Infinite Array (medium) Minimum Difference Element (medium) Bitonic Array Maximum (easy) Problem Challenge 1 Solution Review: Problem Challenge 1 Problem Challenge 2 Solution Review: Problem Challenge 2 Problem Challenge 3 Solution Review: Problem Challenge 3 Pattern: Bitwise XOR Introduction Single Number (easy) Two Single Numbers (medium) Complement of Base 10 Number (medium) Problem Challenge 1 Solution Review: Problem Challenge 1 Pattern: Top 'K' Elements Introduction Top 'K' Numbers (easy) Kth Smallest Number (easy) 'K' Closest Points to the Origin (easy) Connect Ropes (easy) Top 'K' Frequent Numbers (medium) Frequency Sort (medium) Kth Largest Number in a Stream (medium) 'K' Closest Numbers (medium) Maximum Distinct Elements (medium) Sum of Elements (medium) Rearrange String (hard) Problem Challenge 1 Solution Review: Problem Challenge 1 Problem Challenge 2 Solution Review: Problem Challenge 2 Problem Challenge 3 Solution Review: Problem Challenge 3

Pattern: K-way merge

Minimum Difference Element (medium)



Problem Statement

Given an array of numbers sorted in ascending order, find the element in the array that has the minimum difference with the given 'key'.

Example 1:

```
Input: [4, 6, 10], key = 7
Output: 6
Explanation: The difference between the key '7' and '6' is minimum than any other number in th
e array
```

Example 2:

```
Input: [4, 6, 10], key = 4
Output: 4
```

Example 3:

```
Input: [1, 3, 8, 10, 15], key = 12
Output: 10
```

Example 4:

```
Input: [4, 6, 10], key = 17
Output: 10
```

Try it yourself

Try solving this question here:

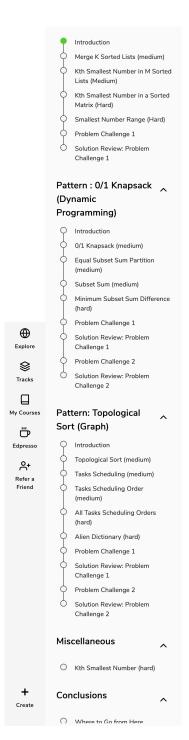
Solution

The problem follows the **Binary Search** pattern. Since Binary Search helps us find a number in a sorted array efficiently, we can use a modified version of the Binary Search to find the number that has the minimum difference with the given 'key'.

We can use a similar approach as discussed in Order-agnostic Binary Search. We will try to search for the 'key' in the given array. If we find the 'key' we will return it as the minimum difference number. If we can't find the 'key', (at the end of the loop) we can find the differences between the 'key' and the numbers pointed out by indices start and end, as these two numbers will be closest to the 'key'. The number that gives minimum difference will be our required number.

Code

Here is what our algorithm will look like:



Time complexity

Since, we are reducing the search range by half at every step, this means the time complexity of our algorithm will be O(log N) where 'N' is the total elements in the given array.

Space complexity

The algorithm runs in constant space O(1).

Interviewing soon? We've partnered with Hired so that companies apply to you instead of you applying to them. See how \odot

