Problem 1713: Minimum Operations to Make a Subsequence

Problem Information

Difficulty: Hard
Acceptance Rate: 49.31%
Paid Only: No
Tags: Array, Hash Table, Binary Search, Greedy

Problem Description

You are given an array `target` that consists of **distinct** integers and another integer array `arr` that **can** have duplicates.

In one operation, you can insert any integer at any position in `arr`. For example, if `arr = [1,4,1,2]`, you can add `3` in the middle and make it `[1,4,_3_ ,1,2]`. Note that you can insert the integer at the very beginning or end of the array.

Return _the**minimum** number of operations needed to make _`target` _a**subsequence** of _`arr` _._

A **subsequence** of an array is a new array generated from the original array by deleting some elements (possibly none) without changing the remaining elements' relative order. For example, `[2,7,4]` is a subsequence of `[4,_2_ ,3,_7_ ,2,1,_4_]` (the underlined elements), while `[2,4,2]` is not.

Example 1:

Input: target = [5,1,3], arr = [9,4,2,3,4] **Output:** 2 **Explanation:** You can add 5 and 1 in such a way that makes arr = [_5_ ,9,4,_1_ ,2,3,4], then target will be a subsequence of arr.

Example 2:

Input: target = [6,4,8,1,3,2], arr = [4,7,6,2,3,8,6,1] **Output:** 3

Constraints:

* `1 <= target.length, arr.length <= 105` * `1 <= target[i], arr[i] <= 109` * `target` contains no duplicates.

Code Snippets

C++:

```
class Solution {
public:
int minOperations(vector<int>& target, vector<int>& arr) {

}
};
```

Java:

```
class Solution {
public int minOperations(int[] target, int[] arr) {

}
}
```

Python3:

```
class Solution:
def minOperations(self, target: List[int], arr: List[int]) -> int:
```