Problem 2555: Maximize Win From Two Segments

Problem Information

Difficulty: Medium
Acceptance Rate: 36.54%
Paid Only: No
Tags: Array, Binary Search, Sliding Window

Problem Description

There are some prizes on the **X-axis**. You are given an integer array `prizePositions` that is **sorted in non-decreasing order** , where `prizePositions[i]` is the position of the `ith` prize. There could be different prizes at the same position on the line. You are also given an integer `k`.

You are allowed to select two segments with integer endpoints. The length of each segment must be `k`. You will collect all prizes whose position falls within at least one of the two selected segments (including the endpoints of the segments). The two selected segments may intersect.

* For example if `k = 2`, you can choose segments `[1, 3]` and `[2, 4]`, and you will win any prize i that satisfies `1 <= prizePositions[i] <= 3` or `2 <= prizePositions[i] <= 4`.

Return _the**maximum** number of prizes you can win if you choose the two segments optimally_.

Example 1:

Input: prizePositions = [1,1,2,2,3,3,5], k = 2 **Output:** 7 **Explanation:** In this example, you can win all 7 prizes by selecting two segments [1, 3] and [3, 5].

Example 2:

Input: prizePositions = [1,2,3,4], k = 0 **Output:** 2 **Explanation:** For this example, **one choice** for the segments is [3, 3] and [4, 4], and you will be able to get 2 prizes.

Constraints:

* `1 <= prizePositions.length <= 105` * `1 <= prizePositions[i] <= 109` * `0 <= k <= 109 ` * `prizePositions` is sorted in non-decreasing order.

Code Snippets

C++:

```
class Solution {
public:
int maximizeWin(vector<int>& prizePositions, int k) {

}
};
```

Java:

```
class Solution {
public int maximizeWin(int[] prizePositions, int k) {

}
}
```

Python3:

```
class Solution:
def maximizeWin(self, prizePositions: List[int], k: int) -> int:
```