Problem 3281: Maximize Score of Numbers in Ranges

Problem Information

Difficulty: Medium
Acceptance Rate: 35.27%
Paid Only: No
Tags: Array, Binary Search, Greedy, Sorting

Problem Description

You are given an array of integers `start` and an integer `d`, representing `n` intervals `[start[i], start[i] + d]`.

You are asked to choose `n` integers where the `ith` integer must belong to the `ith` interval. The **score** of the chosen integers is defined as the **minimum** absolute difference between any two integers that have been chosen.

Return the **maximum** _possible score_ of the chosen integers.

Example 1:

Input: start = [6,0,3], d = 2

Output: 4

Explanation:

The maximum possible score can be obtained by choosing integers: 8, 0, and 4. The score of these chosen integers is `min(|8 - 0|, |8 - 4|, |0 - 4|)` which equals 4.

Example 2:

Input: start = [2,6,13,13], d = 5

Output: 5

Explanation:

The maximum possible score can be obtained by choosing integers: 2, 7, 13, and 18. The score of these chosen integers is `min(|2 - 7|, |2 - 13|, |2 - 18|, |7 - 13|, |7 - 18|, |13 - 18|)` which equals 5.

Constraints:

* `2 <= start.length <= 105` * `0 <= start[i] <= 109` * `0 <= d <= 109`

Code Snippets

C++:

```
class Solution {
public:
int maxPossibleScore(vector<int>& start, int d) {

}
};
```

Java:

```
class Solution {
public int maxPossibleScore(int[] start, int d) {

}
}
```

Python3:

```
class Solution:
def maxPossibleScore(self, start: List[int], d: int) -> int:
```