Problem 3357: Minimize the Maximum Adjacent Element Difference

Problem Information

Difficulty: Hard
Acceptance Rate: 17.28%
Paid Only: No
Tags: Array, Binary Search, Greedy

Problem Description

You are given an array of integers `nums`. Some values in `nums` are **missing** and are denoted by -1.

You must choose a pair of **positive** integers `(x, y)` **exactly once** and replace each **missing** element with _either_ `x` or `y`.

You need to **minimize****** the**maximum** **absolute difference** between _adjacent_ elements of `nums` after replacements.

Return the **minimum** possible difference.

Example 1:

Input: nums = [1,2,-1,10,8]

Output: 4

Explanation:

By choosing the pair as `(6, 7)`, nums can be changed to `[1, 2, 6, 10, 8]`.

The absolute differences between adjacent elements are:

* `|1 - 2| == 1` * `|2 - 6| == 4` * `|6 - 10| == 4` * `|10 - 8| == 2`

Example 2:

Input: nums = [-1,-1,-1]

Output: 0

Explanation:

By choosing the pair as `(4, 4)`, nums can be changed to `[4, 4, 4]`.

Example 3:

Input: nums = [-1,10,-1,8]

Output: 1

Explanation:

By choosing the pair as `(11, 9)`, nums can be changed to `[11, 10, 9, 8]`.

Constraints:

* `2 <= nums.length <= 105` * `nums[i]` is either -1 or in the range `[1, 109]`.

Code Snippets

C++:

```
class Solution {
public:
int minDifference(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int minDifference(int[] nums) {
```

```
}
}
```

Python3:

```
class Solution:
def minDifference(self, nums: List[int]) -> int:
```