719. Find K-th Smallest Pair Distance

DescriptionHintsSubmissionsDiscussSolution

Given an integer array, return the k-th smallest **distance** among all the pairs. The distance of a pair (A, B) is defined as the absolute difference between A and B.

Example 1:

```
Input:
nums = [1,3,1]
k = 1
Output: 0
Explanation:
Here are all the pairs:
(1,3) -> 2
(1,1) -> 0
(3,1) -> 2
Then the 1st smallest distance pair is (1,1), and its distance is 0.
```

Note:

1. `2 <= len(nums) <= 10000`.
2. `0 <= nums[i] < 1000000`.
3. `1 <= k <= len(nums) * (len(nums) - 1) / 2`.

Seen this question in a real interview before?

- Difficulty:Hard
- Total Accepted:1.6K
- Total Submissions:6.8K
- Contributor: fallcreek
-
- Subscribe to see which companies asked this question.

  Related Topics Array, Binary Search

  Similar Questions

  Find K Pairs with Smallest Sums Kth Smallest Element in a Sorted Matrix Find K Closest Elements Kth Smallest Number in Multiplication Table

```
#include<stdio.h>
#include<iostream>
#include<vector>
#include<algorithm>
using namespace std;
```

```
int countPairs(vector<int> &nums, int mid)
{
     int res = 0;
     for(int i=0;i<(int)nums.size();i++)
     {
          int j=i;
          while(j<(int)nums.size() && nums[j]-nums[i]<=mid) j++;
          res+=j-i-1;
     }
     return res;
}
int smallestDistancePair(vector<int>& nums, int k) {
     int n = nums.size();
     sort(nums.begin(),nums.end());
     int left = nums[1]-nums[0];
     for(int i=1;i<(int)nums.size();i++)
     {
          left = min(left,nums[i]-nums[i-1]);
     }
     int right = nums[n-1]-nums[0];
     int mid;
     while(left<=right)
     {
          mid = (left+right)/2;
          int tmp = countPairs(nums,mid);
          if(tmp>=k)
          {
               right=mid-1;
          }else{
               left=mid+1;
          }
     }
     return left;
}
int main(int argc, char *argv[])
{
     vector<int> test = {1,3,1};
     int ans = smallestDistancePair(test,1);
     cout<<ans;
     return 0;
}
```