leetCode:  
There are two sorted arrays **nums1** and **nums2** of size m and n respectively.

Find the median of the two sorted arrays. The overall run time complexity should be O(log (m+n)).

**Example 1:**

nums1 = [1, 3]

nums2 = [2]

The median is 2.0

**Example 2:**

nums1 = [1, 2]

nums2 = [3, 4]

The median is (2 + 3)/2 = 2.5

第一次答案：迭代的方法

|  |
| --- |
| import java.util.List;  import java.util.ArrayList;  import java.util.Collections;  public class Solution {  public double findMedianSortedArrays(int[] nums1, int[] nums2) {  double result=0;  ArrayList<Integer> array = new ArrayList<Integer>();  for(int i:nums1){  array.add(i);  }  for(int j:nums2){  array.add(j);  }  Collections.sort(array);  if(array.size()%2==0){  int index = array.size()/2;  result = (array.get(index)+array.get(index-1))/2.0;  }else{  int index = array.size()/2;  result= array.get(index);  }  return result;  }  public static void main(String[] args){  int[] nums1 = {1,3,6};  int[] nums2 = {2,4,5};  double result;  Solution s = new Solution();  result = s.findMedianSortedArrays(nums1,nums2);  System.out.println(Math.min(3,2));  System.out.println(result);  }  } |

第二次用递归方法

k=length/2

保持p1+p2=k

|  |
| --- |
| public class Solution {  public double findMedianSortedArrays(int[] nums1, int[] nums2) {  int m = nums1.length, n = nums2.length;  int k = (m + n) / 2;  if((m+n)%2==0){  return (tools(nums1,nums2,0,0,m,n,k)+tools(nums1,nums2,0,0,m,n,k+1))/2;  }else{  return tools(nums1,nums2,0,0,m,n,k+1);  }  }  private double tools(int[] nums1, int[] nums2, int start1, int start2, int len1, int len2, int k){  if(len1>len2){  return tools(nums2,nums1,start2,start1,len2,len1,k);  }  if(len1==0){  return nums2[start2+k-1];  }  if(k==1){  return Math.min(nums1[start1],nums2[start2]);  }  int p1 = Math.min(k/2,len1);  int p2 = k-p1;  if(nums1[start1 + p1-1]<nums2[start2 + p2-1]){  return tools(nums1,nums2,start1 + p1,start2,len1-p1,len2,k-p1);  }else if(nums1[start1 + p1-1]>nums2[start2 + p2-1]){  return tools(nums1,nums2,start1,start2 + p2,len1,len2-p2,k-p2);  }else{  return nums1[start1+p1-1];  }    }  } |