# 4. Median of Two Sorted Arrays

Given two sorted arrays nums1 and nums2 of size m and n respectively, return the median of the two sorted arrays.

The overall run time complexity should be O(log (m+n)).

## SOLUTION IN JAVA

class Solution {

public double findMedianSortedArrays(int[] nums1, int[] nums2) {

if(nums1.length == 0 && nums2.length != 0){

int size = nums2.length / 2;

if(nums2.length % 2 == 0){

return (nums2[size - 1] + nums2[size]) / 2.0d;

}else{

return nums2[size];

}

}

if(nums2.length == 0 && nums1.length != 0){

int size = nums1.length / 2;

if(nums1.length % 2 == 0){

return (nums1[size - 1] + nums1[size]) / 2.0d;

}else{

return nums1[size];

}

}

if(nums2.length == 0 && nums1.length == 0){

return 0.0d;

}

List<Integer> list = new ArrayList<>();

int index1 = 0;

int index2 = 0;

int complete = 0;

while(index1 < nums1.length && index2 < nums2.length){

if(nums1[index1] <= nums2[index2]){

list.add(new Integer(nums1[index1]));

index1++;

}else{

list.add(new Integer(nums2[index2]));

index2++;

}

if(index1 == nums1.length){

complete = 1;

}else if(index2 == nums2.length){

complete = 2;

}

}

if(complete == 1){

for(; index2 < nums2.length; index2++){

list.add(new Integer(nums2[index2]));

}

}else{

for(; index1 < nums1.length; index1++){

list.add(new Integer(nums1[index1]));

}

}

int size = list.size();

if(size % 2 == 0){

return (list.get(size/2 - 1) + list.get(size/2)) / 2.0d;

}else{

return list.get(size/2);

}

}

}