Sort an array of 0s, 1s and 2s

Easy Accuracy: 50.58% Submissions: 489K+ Points: 2

Given an array of size **N** containing only 0s, 1s, and 2s; sort the array in ascending order.

Example 1:

Input:
N = 5
arr[]= {0 2 1 2 0}
Output:
0 0 1 2 2
Explanation:
0s 1s and 2s are segregated into ascending order.

Example 2:

C++ (g++ 5.4) Average Time: 10m Start Timer

```
// } Driver Code Ends
class Solution
{
    public:
    void sort012(int arr[], int n)
    {
        int low = 0;
        int high = n-1;
        int mid = 0;

        while(mid <= high){
            if(arr[mid] == 0){
                swap(arr[mid] , arr[low]);
                mid++;
                low++;
            }else if(arr[mid] == 1){
                mid++;
            }else{
                swap(arr[mid] , arr[high]);
                high--;
            }

        }
    }

};
// } Driver Code Ends
```

Custom Input | Compile & Run | Submit

Median of 2 Sorted Arrays of Different Sizes

Hard Accuracy: 28.4% Submissions: 58K+ Points: 8

Given two sorted arrays array1 and array2 of size **m** and **n** respectively. Find the median of the two sorted arrays.

Example 1:

Input:
m = 3, n = 4
array1[] = {1,5,9}
array2[] = {2,3,6,7}
Output: 5
Explanation: The middle element for {1,2,3,5,6,7,9} is 5

Example 2:

Input:

C++ (g++ 5.4) | Average Time: 35m | Start Timer

```
// } Driver Code Ends
//User function Template for C++

class Solution{
    public:
    double MedianOfArrays(vector<int>& array1, vector<int>& array2)
    {
        // Your code goes here
        vector<int > v;
        for(int i = 0 ; i<array1.size() ; i++){
            v.push_back(array1[i]);
        }
        for(int i = 0 ; i<array2.size();i++){
            v.push_back(array2[i]);
        }

        sort(v.begin() , v.end());
        int size = v.size();

        int mid = v.size()/2;

        if((size%2) != 0){
            double res = v[mid];
            return res;
        }
        double ans = v[mid] + v[mid-1];
        return ans/2;



    }
};
// } Driver Code Ends
```

Custom Input | Compile & Run | Submit

Description Discussion (149) Solutions (9.9K) Submissions

4. Median of Two Sorted Arrays

Hard 22.1K 2.5K

Companies

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))`.

Example 1:

```
Input: nums1 = [1,3], nums2 = [2]
Output: 2.00000
Explanation: merged array = [1,2,3] and median is 2.
```

Example 2:

```
Input: nums1 = [1,2], nums2 = [3,4]
Output: 2.50000
Explanation: merged array = [1,2,3,4] and median is (2 + 3) / 2 = 2.5.
```

C++ Auto

```cpp
class Solution {
public:
    double findMedianSortedArrays(vector<int>& nums1, vector<int>& nums2) {
        vector<int > v;
        for(int i = 0 ; i<nums1.size() ; i++){
            v.push_back(nums1[i]);
        }
        for(int i = 0 ; i<nums2.size();i++){
            v.push_back(nums2[i]);
        }

        sort(v.begin() , v.end());
        int size = v.size();

        int mid = v.size()/2;

        if((size%2) != 0){
            double res = v[mid];
            return res;
        }
        double ans = v[mid] + v[mid-1];
        return ans/2;
    }
};
```