

geeksforgeeks.org/problems/common-elements1132/1

Three SD Ending Courses Tutorials Practice Jobs

Search... Problem Editorial Submissions Comments

Common in 3 Sorted Arrays

Difficulty: Easy Accuracy: 22.16% Submissions: 439K+ Points: 2

Given three sorted arrays in **non-decreasing** order, print all common elements in **non-decreasing** order across these arrays. If there are no such elements return an empty array. In this case, the output will be -1.

Note: can you handle the duplicates without using any additional Data Structure?

Examples :

Input: arr1 = [1, 5, 10, 20, 40, 80] , arr2 = [6, 7, 20, 80, 100] , arr3 = [3, 4, 15, 20, 30, 70, 80, 120]
Output: [20, 80]
Explanation: 20 and 80 are the only common elements in arr1, arr2 and arr3.

Input: arr1 = [1, 2, 3, 4, 5] , arr2 = [6, 7] , arr3 = [8,9,10]
Output: [-1]
Explanation: There are no common elements in arr1, arr2 and arr3.

Input: arr1 = [1, 1, 1, 2, 2, 2], arr2 = [1, 1, 2, 2, 2], arr3 = [1, 1, 1, 1, 2, 2, 2, 2]
Output: [1, 2]
Explanation: We do not need to consider duplicates

Constraints:
 $1 \leq \text{arr1.size(), arr2.size(), arr3.size()} \leq 10^5$
 $-10^5 \leq \text{arr1}_i, \text{arr2}_i, \text{arr3}_i \leq 10^5$

```

Java (21) Start Timer
1 // User Function Template for Java
2
3 class Solution {
4     //Function to find common elements in three arrays.
5     public List<Integer> commonElements(List<Integer> arr1, List<Integer> arr2,
6                                         List<Integer> arr3) {
7     }
8     // Code Here
9     int i = 0, j = 0, k = 0;
10    int s1 = arr1.size();
11    int s2 = arr2.size();
12    int s3 = arr3.size();
13    List<Integer> list = new ArrayList<>();
14
15    while(i < s1 && j < s2 && k < s3){
16        int a = arr1.get(i);
17        int b = arr2.get(j);
18        int c = arr3.get(k);
19        int ls = list.size();
20        if(a == b && b == c){
21            if(list.isEmpty() || list.get(ls-1) != a){
22                list.add(a);
23            }
24            i++;
25            j++;
26            k++;
27        }
28        else if(a < b){
29            i++;
30        }
31        else if(b < c){
32            j++;
33        }
34        else{
35            k++;
36        }
37    }
38    if(list.isEmpty()){
39        list.add(-1);
40    }
41
42    return list;
43
44}
45
46
47

```

Custom Input Compile & Run Submit

Problem List < > ✎ Submit ⌂ ⌂ Premium

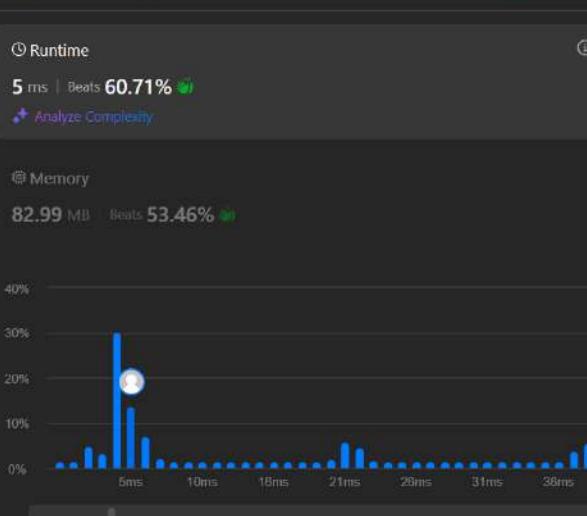
Description Accepted Editorial Solutions Submissions

All Submissions

Accepted 59 / 59 testcases passed sandeep submitted at Feb 03, 2026 21:04

Runtime 5 ms | Beats 60.71% Analyze Complexity

Memory 82.99 MB | Beats 53.46%



Code

Java Auto

```
1 class Solution {
2     public int findDuplicate(int[] nums) {
3         int slow = nums[0];
4         int fast = nums[0];
5
6         do {
7             slow = nums[slow];
8             fast = nums[nums[fast]];
9         } while (slow != fast);
10
11         slow = nums[0];
12         while (slow != fast) {
13             slow = nums[slow];
14             fast = nums[fast];
15         }
16
17         return slow;
18     }
19 }
```

Saved Ln 20, Col 1

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Get 90% Refund Courses Tutorials Practice Jobs

Search... Problem Editorial Submissions Comments

Output Window

Compilation Results Custom Input Y.O.G.I. (AI Bot)

Problem Solved Successfully ✓ Suggest Feedback

Test Cases Passed 1111 / 1111 Attempts : Correct / Total 1 / 2 Accuracy : 50%

Points Scored 4 / 4 Time Taken 2.85

Your Total Score: 29 ↑

Java (21) Start Timer

```
1 class Solution {  
2     public void mergeArrays(int a[], int b[]) {  
3         for(int i=0;i<b.length;i++){  
4             for(int j=0;j<a.length;j++){  
5                 if(b[i]<=a[j]){  
6                     int temp=b[i];  
7                     b[i]=a[j];  
8                     a[j]=temp;  
9                 }  
10            }  
11        }  
12        Arrays.sort(b);  
13    }  
14 }  
15 }  
16 }
```

Solve Next Custom Input Compile & Run Submit

Problem List < > ✎

Submit

Description Accepted Editorial Solutions Submissions

All Submissions: sandeep submitted at Feb 03, 2026 21:15

Accepted 172 / 172 testcases passed

Runtime 8 ms | Beats 90.00% Analyze Complexity

Memory 49.18 MB | Beats 51.43%

Runtime distribution chart showing a single peak at 8ms.

Code

Java Auto

```
1 class Solution {  
2     public int[][] merge(int[][] intervals) {  
3         if (intervals.length == 0) return new int[0][0];  
4  
5         Arrays.sort(intervals, (a, b) -> a[0] - b[0]);  
6  
7         List<int[]> res = new ArrayList<>();  
8         int s = intervals[0][0];  
9         int e = intervals[0][1];  
10  
11        for (int i = 1; i < intervals.length; i++) {  
12            if (intervals[i][0] <= e) {  
13                e = Math.max(e, intervals[i][1]);  
14            } else {  
15                res.add(new int[]{s, e});  
16                s = intervals[i][0];  
17                e = intervals[i][1];  
18            }  
19        }  
20    }
```

Saved Upgrade to Cloud Saving Ln 24, Col 2

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Get 90% Refund

Courses Tutorials Practice Jobs

Search...

Problem Editorial Submissions Comments

Output Window

Compilation Results Custom Input Y.O.G.I. (AI Bot)

Java (21) Start Timer

1 class Solution {
2 public boolean isSubset(int a[], int b[]) {
3 // Your code here
4 Arrays.sort(a);
5 Arrays.sort(b);
6 int ai = 0, bi = 0;
7 while(ai < a.length && bi < b.length){
8 if(a[ai] == b[bi])
9 bi++;
10 ai++;
11 }
12 if(bi == b.length)
13 return true;
14 return false;
15 }
16 }

Test Cases Passed: 1114 / 1114

Attempts : Correct / Total: You can see all your attempts in submission tab

Accuracy : 100%

Points Scored: You can see the score in submission tab

Time Taken: 0.51

Custom Input Compile & Run Submit

Get SWR Refund Courses Tutorials Practice Jobs

Search... Problem Editorial Submissions Comments

Java (21) Start Timer

```
1 class Solution {  
2     public boolean hasTripletSum(int arr[], int target) {  
3         Arrays.sort(arr);  
4  
5         for(int i=0;i<arr.length-2;i++){  
6             int left=i+1;  
7             int right=arr.length-1;  
8  
9             while(left<right){  
10                 int sum=arr[i]+arr[left]+arr[right];  
11                 if(sum==target){  
12                     return true;  
13                 }else if(sum<target){  
14                     left++;  
15                 }else{  
16                     right--;  
17                 }  
18             }  
19         }  
20         return false;  
21     }  
22 }  
23 }
```

Output Window

Compilation Results Custom Input Y.O.G.I. (AI Bot)

Problem Solved Successfully Suggest Feedback

Test Cases Passed Attempts : Correct / Total
1111 / 1111 1 / 1

Accuracy : 100%

Points Scored Time Taken
4 / 4 0.19

Your Total Score: 38

Solve Next Custom Input Compile & Run Submit

Get 90% Refund Courses Tutorials Practice Jobs

Search... Problem Editorial Submissions Comments Java (21) Start Timer

Output Window Compilation Results Custom Input Y.O.G.I. (AI Bot)

Problem Solved Successfully ✓ Suggest Feedback

Test Cases Passed Attempts : Correct / Total
1111 / 1111 **1 / 1**

Accuracy : 100%

Points Scored Points Scored
8 / 8

Your Total Score: **46** ↑ Time Taken
0.3

```
int rightMax[] = new int[n];
rightMax[n-1] = arr[n-1];
for(int i=n-2; i>=0; i--){
    rightMax[i] = Math.max( rightMax[i+1], arr[i] );
}

int trappedwater = 0;
for(int i=0; i<n; i++){
    int waterLevel = Math.min( leftMax[i], rightMax[i] );
    trappedwater += waterLevel - arr[i];
}
return trappedwater;

public static void main (String args[]){
    int arr[] = {3, 0, 1, 0, 4, 0, 2};
    Solution obj = new Solution();
    obj.maxWater( arr );
}
```

Solve Next Custom Input Compile & Run Submit