Day 10 before class rev

19 November 2022 11:01

Common elements

Easy Accuracy: **22.16%** Submissions: **100k+** Points: **2**

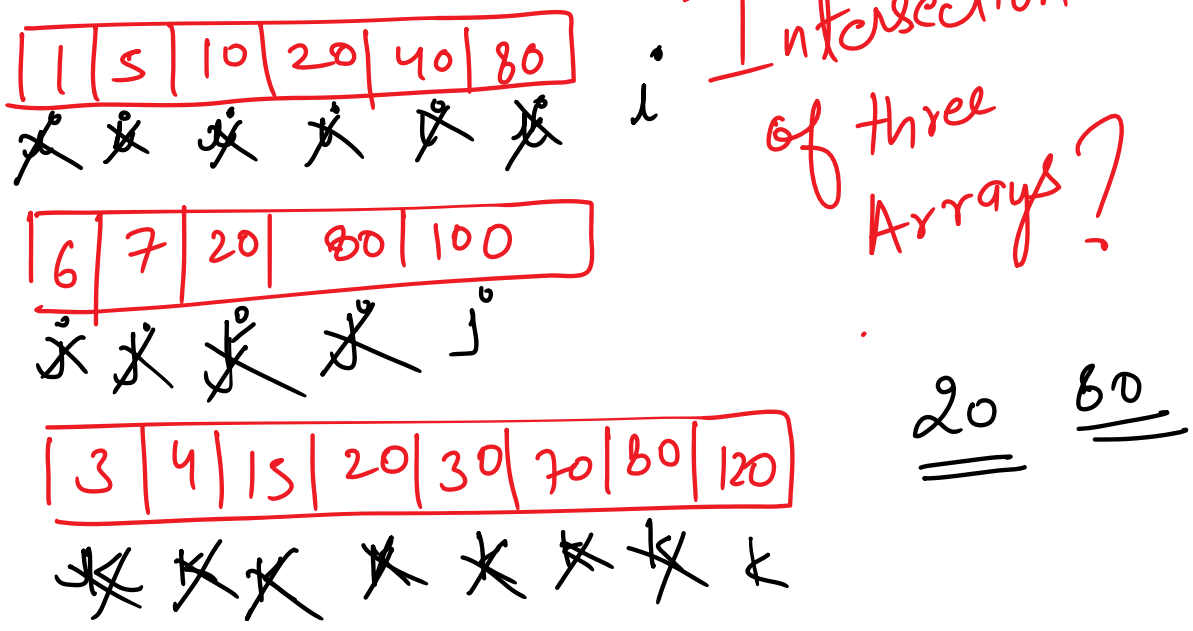
Given three arrays sorted in increasing order. Find the elements that are common in all three arrays.

Note: can you take care of the duplicates without using any additional Data Structure?

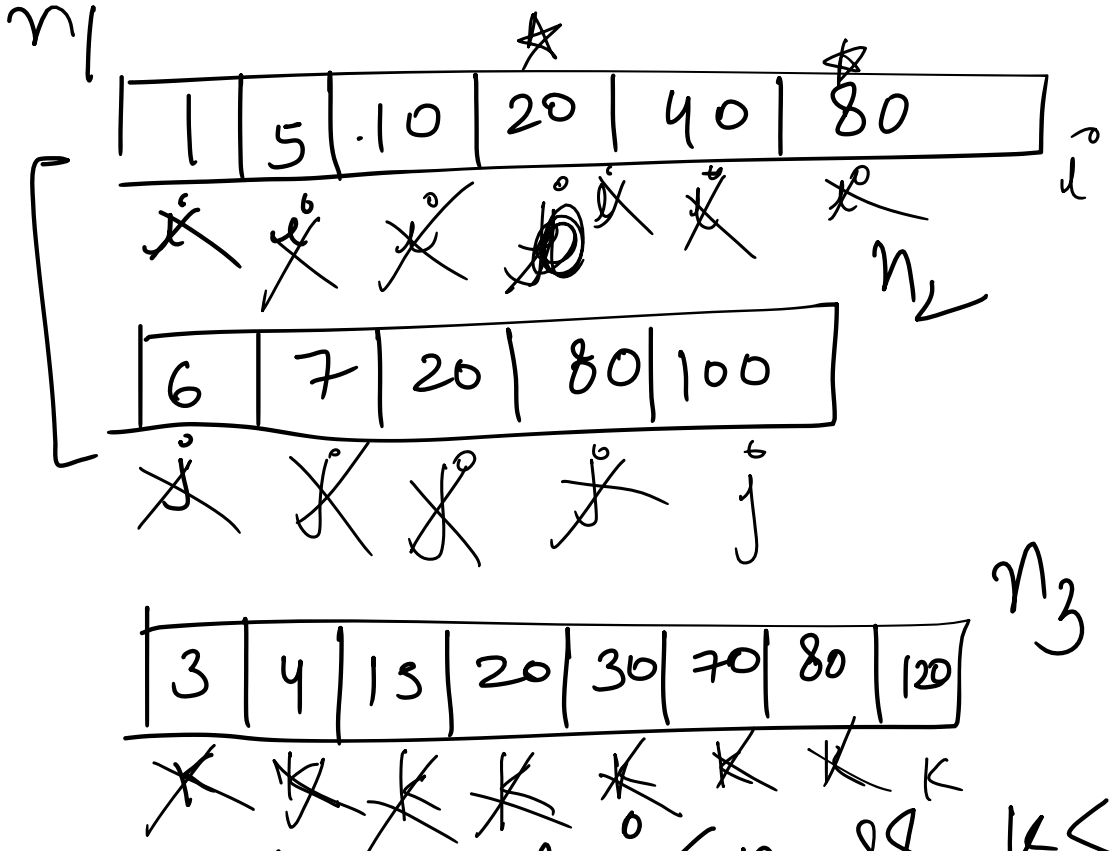
Example 1:

Input:
n1 = 6; A = {1, 5, 10, 20, 40, 80}
n2 = 5; B = {6, 7, 20, 80, 100}
n3 = 8; C = {3, 4, 15, 20, 30, 70, 80, 120}
Output: 20 80
Explanation: 20 and 80 are the only common elements in A, B and C.

Three Sorted Arrays
Find the elements that are common in all 3 arrays:



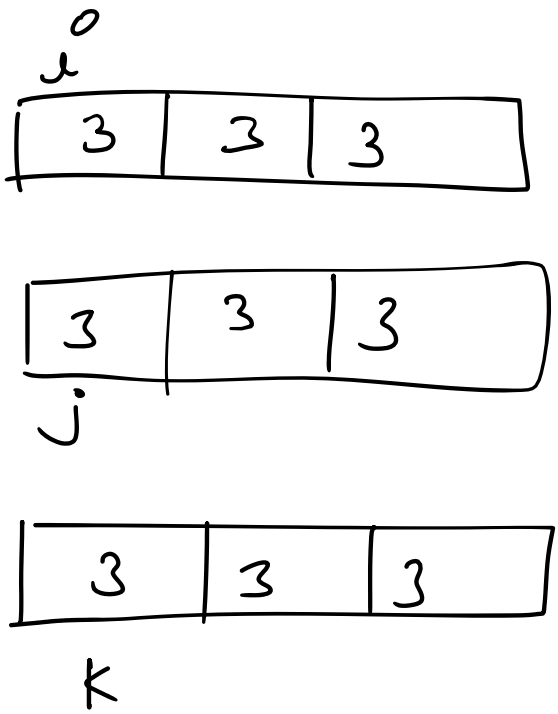
Common elements are 20 80



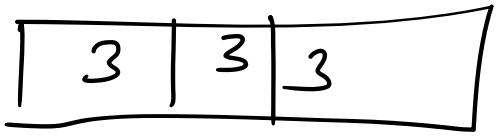

```
while( i<n1 && j<n2 && k<n3)
{  if(A1[i]==A2[j]==A3[k])
   {  v.push_back(A1[i]);
      i++, j++; k++;
```

```
else if ((A1[i] < A2[j])){
    i++;
}else if (A2[j] < A3[k]){
    j++;
}else{ k++;
}
```

Ex



Output



(According to Algorithm)

Expected.

output 1 3 ✓

So we think something to solve it.

small update in answer vector we have to make.

```
if(A[i] == B[j] && B[j] == C[k]){
            if(ans.size() != 0){
               if(ans.back() != A[i]){
                  ans.push_back(A[i]);
               }
            }else{
               ans.push_back(A[i]);
            }
            i++;j++;k++;
         }
```

Given an unsorted array **Arr** of **N** positive and negative numbers. Your task is to create an array of alternate positive and negative numbers without changing the relative order of positive and negative numbers.

Note: Array should start with a positive number.

Example 1:

Input:

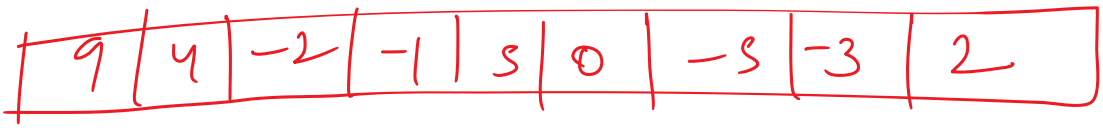
N = 9

Arr[] = {9, 4, -2, -1, 5, 0, -5, -3, 2}

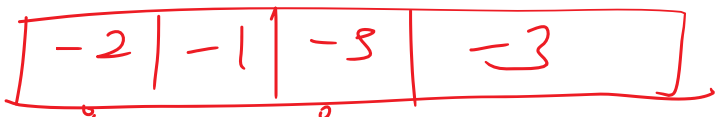
Output:

9 -2 4 -1 5 -5 0 -3 2

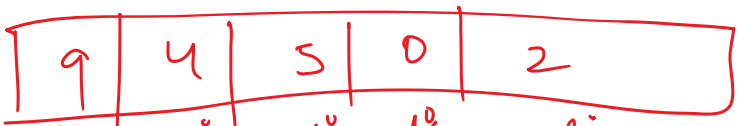
Naive Approach



Negative



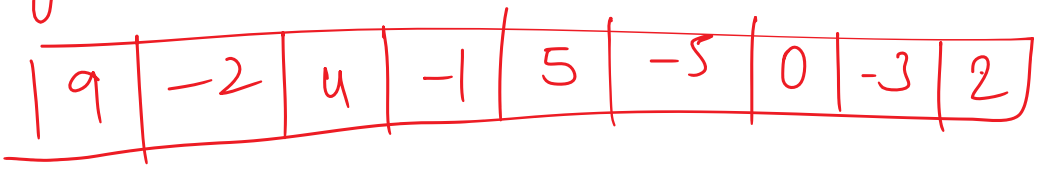
Positive



S.C

O(N)

original

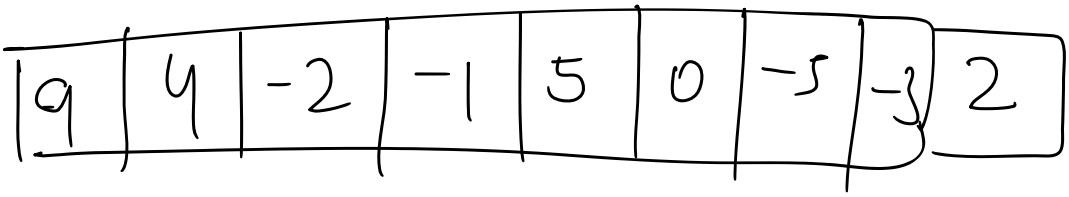


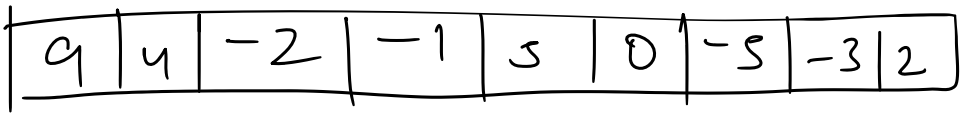
Ans

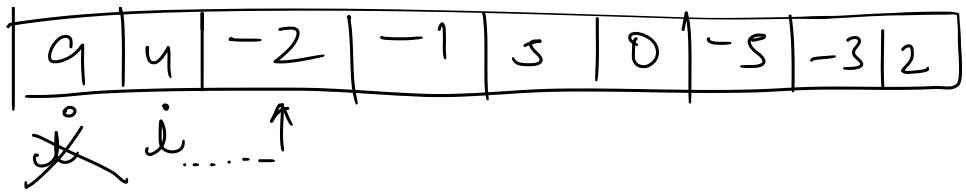
T.C - O(N)

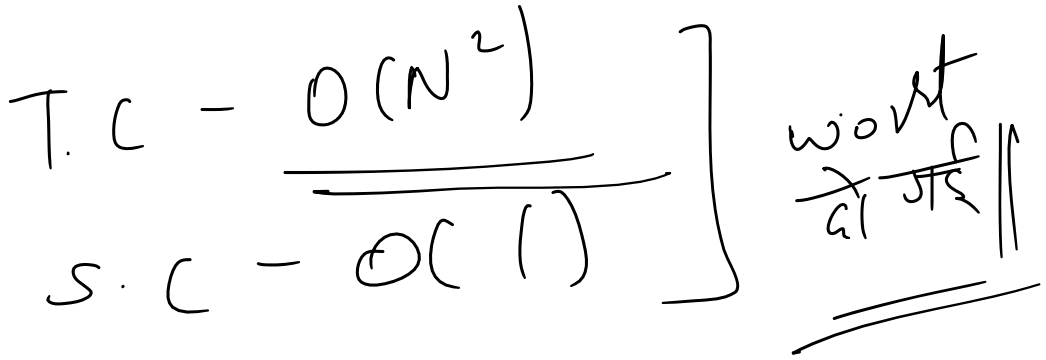
S.C - O(N)

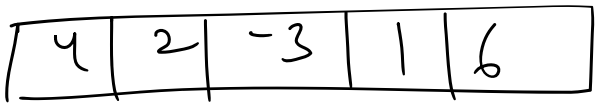
optimised O(1)











Given an array of positive and negative numbers. Find if there is a **subarray** (of size at-least one) with **0 sum**.

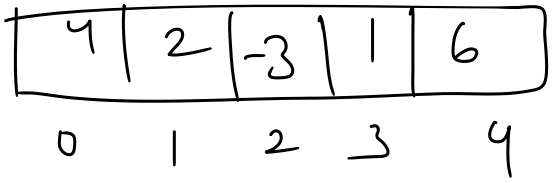
Example 1:

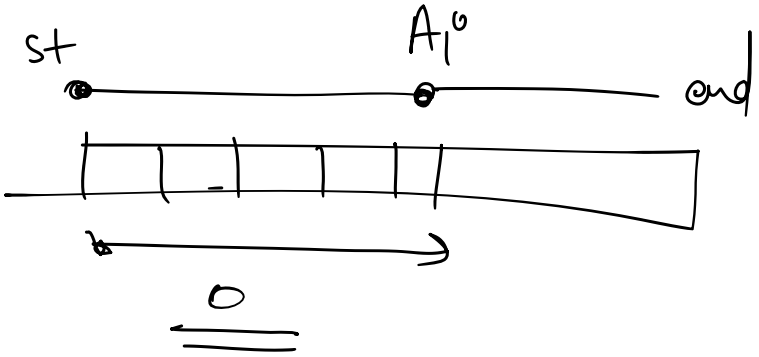
Input:
5
4 2 -3 1 6

Output:
Yes

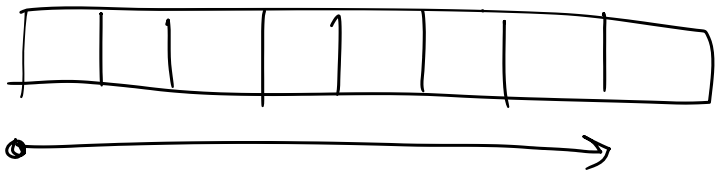
Explanation:
2, -3, 1 is the subarray

Explanation:
2, -3, 1 is the subarray
with sum 0.



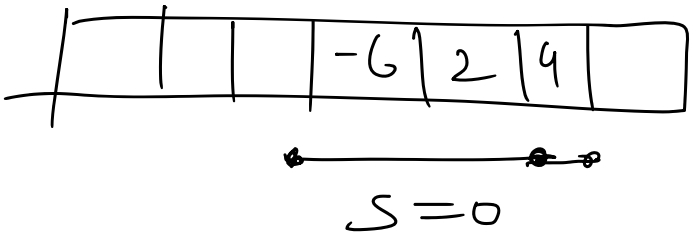


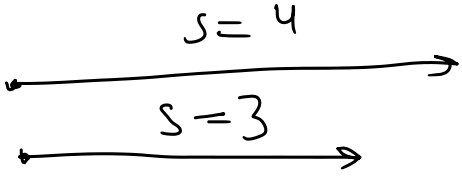
Simple

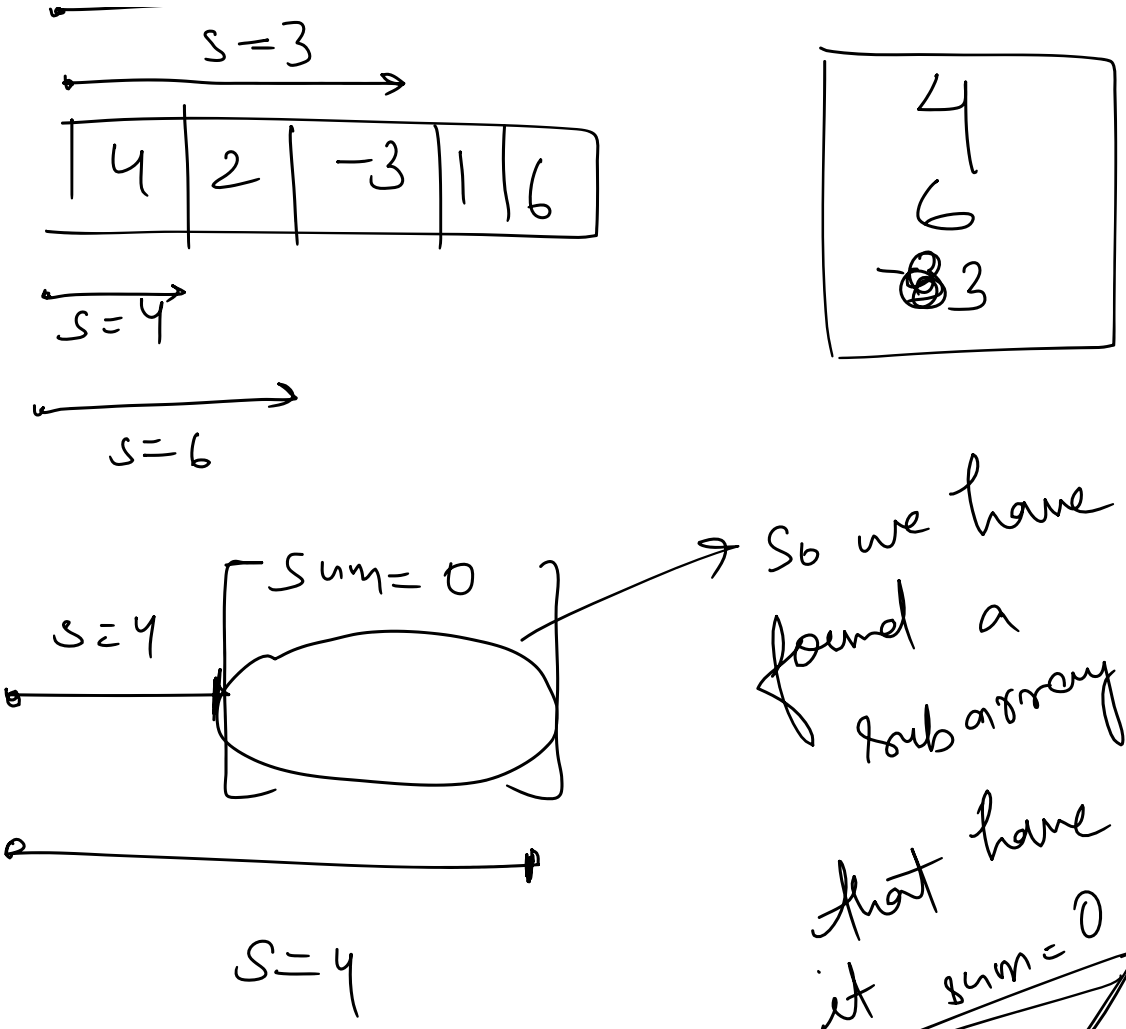


Case 2

Thoda Diffficult






s=3
4 | 2 | -3 | 1 | 6
s=4
s=6
4
6
-3
sum=0
s=4
s=4
So we have found a subarray that have it sum=0