**Find the closest pair from two arrays: -**

**Easy** Accuracy: **18.01%** Submissions: **32K+** Points: **2**

Given two **sorted**arrays **arr** and **brr** and a number**x**, find the pair whose **sum** is closest to **x** and the pair has an element from**each** array. In the case of multiple closest pairs return any one of them.  
Note: Can return the two numbers in any manner.

**Example 1:**

**Input :** N = 4, M = 4  
arr[ ] = {1, 4, 5, 7}

brr[ ] = {10, 20, 30, 40}

X = 32

**Output :** 1

**Explanation:**

The closest pair whose sum is closest

to 32 is {1, 30} = 31.

**Example 2:**

**Input :** N = 4, M = 4  
arr[ ] = {1, 4, 5, 7}

brr[ ] = {10, 20, 30, 40}

X = 50

**Output :** 3

**Explanation:**

The closest pair whose sum is closest

to 50 is {7, 40} = 47.

**Your Task:**  
You only need to complete the function **printClosest()** that takes an array **(arr)**, another array **(brr)**, size of array arr **(N),**size of array brr **(M),** and return the array of two integers whose sum is closest to **X**. The driver code takes care of the printing of the closest difference.

**Expected Time Complexity:** O(N+M).  
**Expected Auxiliary Space:** O(1).

**Constraints:**  
1 ≤ N, M ≤ 105  
1 ≤ A[i], B[i] ≤ 109

**Code: -**

//{ Driver Code Starts

#include <bits/stdc++.h>

using namespace std;

// } Driver Code Ends

//User function teamplate for C++

class Solution{

public:

vector<int> printClosest(int arr[], int brr[], int n, int m, int x) {

//code here

int left = 0, right = m-1, dif = INT\_MAX, item1, item2;

while(left < n and right >= 0){

if(abs(arr[left] + brr[right] - x) < dif){

dif = abs(arr[left] + brr[right] - x);

item1 = arr[left];

item2 = brr[right];

}

if(arr[left] + brr[right] < x)

++left;

else

--right;

}

return {item1, item2};

}

};

//{ Driver Code Starts.

int main()

{

int t;

cin>>t;

int m,n,x;

while(t--)

{

cin>>n>>m;

int a[n],b[m];

for(int i=0;i<n;i++)

cin>>a[i];

for(int i=0;i<m;i++)

cin>>b[i];

cin>>x;

vector<int> ans;

Solution ob;

ans = ob.printClosest(a, b, n, m, x);

cout << abs(ans[0] + ans[1] - x) << endl;

}

return 0;

}

// } Driver Code Ends

**T.C: - O(N + M)**

**S.C: - O(1)**