Started on	Wednesday, 20 November 2024, 6:32 PM
State	Finished
Completed on	Wednesday, 20 November 2024, 6:33 PM
Time taken	51 secs
Marks	1.00/1.00
Grade	**30.00** out of 30.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Find the intersection of two sorted arrays.

OR in other words,

Given 2 sorted arrays, find all the elements which occur in both the arrays.

Input Format

· The first line contains T, the number of test cases. Following T lines contain:

1. Line 1 contains N1, followed by N1 integers of the first array
2. Line 2 contains N2, followed by N2 integers of the second array

Output Format

The intersection of the arrays in a single line

Example

Input:

1

3 10 17 57

6 2 7 10 15 57 246

Output:

10 57

Input:

1

6 1 2 3 4 5 6

2 1 6

Output:

1 6

For example:

Input	Result
1 3 10 17 57 6 2 7 10 15 57 246	10 57

Answer: (penalty regime: 0 %)

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_SIZE 100000

void findIntersection(int arr1[], int n1, int arr2[], int n2) {
    int *hashTable = (int *)calloc(MAX_SIZE, sizeof(int));

    for (int i = 0; i < n1; i++) {
        hashTable[arr1[i]] = 1;
    }

    int found = 0;
    for (int j = 0; j < n2; j++) {
        if (hashTable[arr2[j]] == 1) {
            printf("%d ", arr2[j]);
```

```
            found = 1;
            hashTable[arr2[j]] = 0;
        }
    }

    if (!found) {
        printf("No intersection");
    }
    printf("\n");

    free(hashTable);
}
int main() {
    int T;
    scanf("%d", &T);

    while (T--) {
        int n1;
        scanf("%d", &n1);
        int arr1[n1];
        for (int i = 0; i < n1; i++) {
            scanf("%d", &arr1[i]);
        }

        int n2;
        scanf("%d", &n2);
        int arr2[n2];
        for (int i = 0; i < n2; i++) {
            scanf("%d", &arr2[i]);
        }

        findIntersection(arr1, n1, arr2, n2);
    }

    return 0;
}
```

	Input	Expected	Got	
✔	1 3 10 17 57 6 2 7 10 15 57 246	10 57	10 57	✔
✔	1 6 1 2 3 4 5 6 2 1 6	1 6	1 6	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 2-Finding Duplicates-O(n) Time Complexity,O(1) Space Complexity

Jump to...

4-Print Intersection of 2 sorted arrays-O(m+n)Time Complexity,O(1) Space Complexity ►