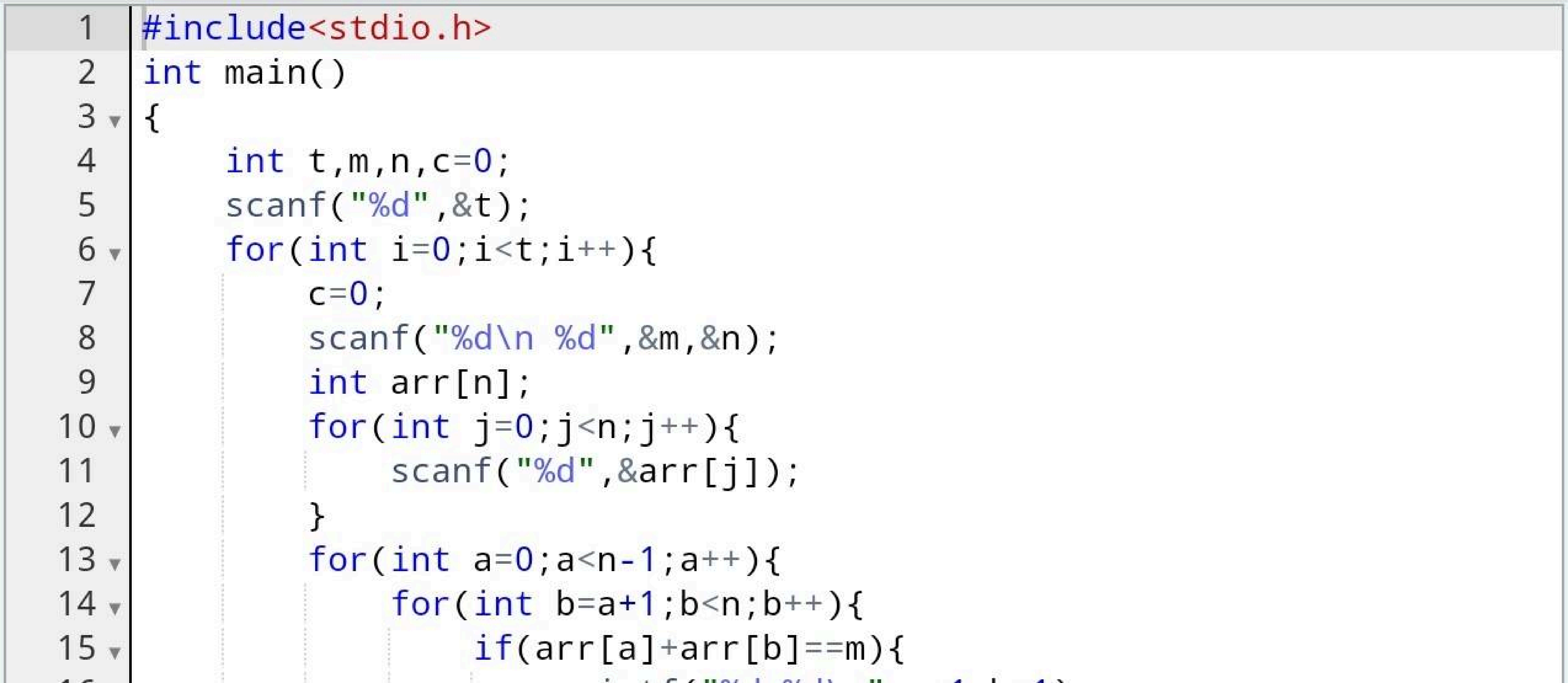
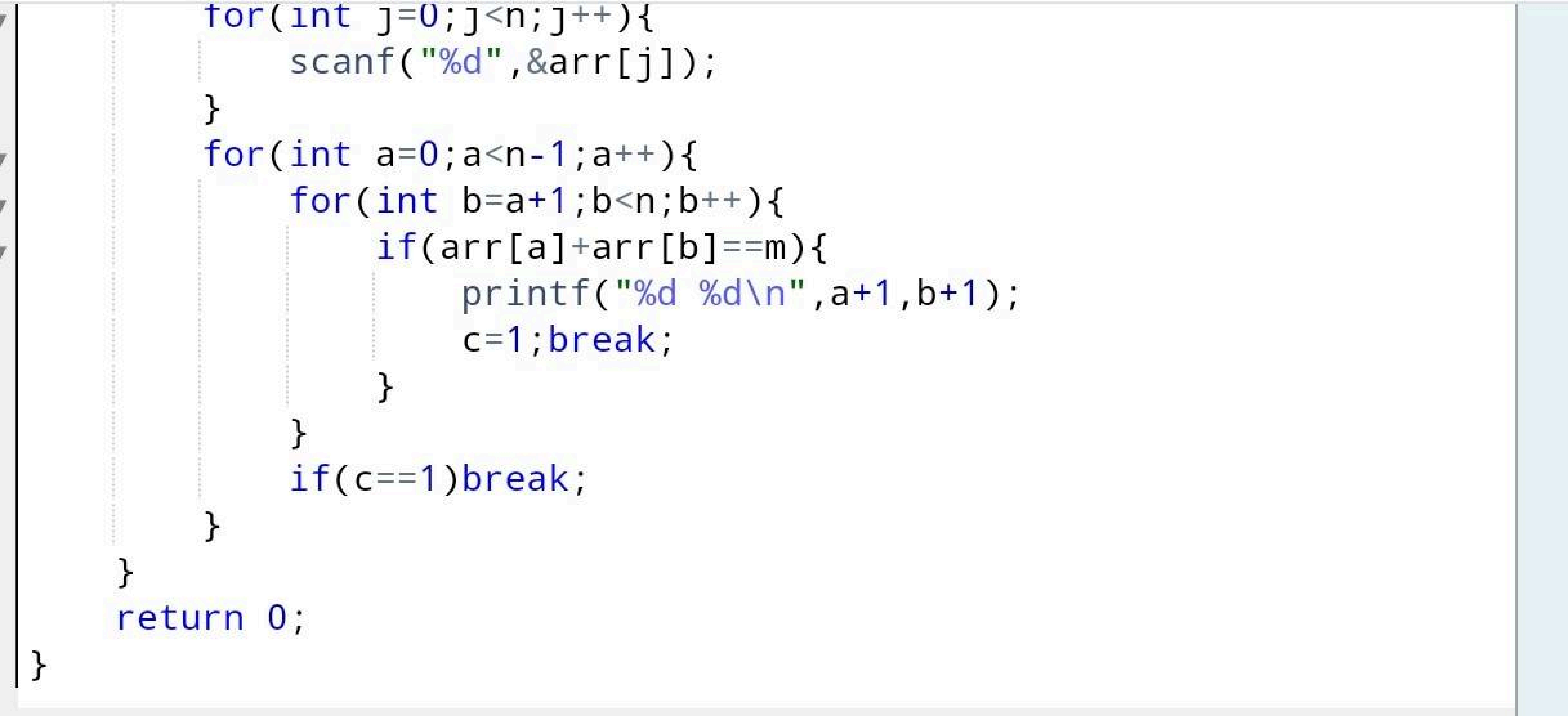
Sunny and Johnny like to pool their money and go to the ice cream parlor. Johnny never buys the same flavor that Sunny does. The only other rule they have is that they spend all of their money.

Given a list of prices for the flavors of ice cream, select the two that will cost all of the money they have.

For example, they have m = 6 to spend and there are flavors costing cost = [1, 2, 3, 4, 5, 6]. The two flavors costing 1 and 5 meet the criteria. Using 1-based indexing, they are at indices 1 and 4.



```c
#include<stdio.h>
int main()
{
    int t,m,n,c=0;
    scanf("%d",&t);
    for(int i=0;i<t;i++){
        c=0;
        scanf("%d\n %d",&m,&n);
        int arr[n];
        for(int j=0;j<n;j++){
            scanf("%d",&arr[j]);
        }
        for(int a=0;a<n-1;a++){
            for(int b=a+1;b<n;b++){
                if(arr[a]+arr[b]==m){
```



```c
        for(int j=0;j<n;j++){
            scanf("%d",&arr[j]);
        }
        for(int a=0;a<n-1;a++){
            for(int b=a+1;b<n;b++){
                if(arr[a]+arr[b]==m){
                    printf("%d %d\n",a+1,b+1);
                    c=1;break;
                }
            }
            if(c==1)break;
        }
    }
    return 0;
}
```

	Input	Expected	Got	
✓	2 4 5 1 4 5 3 2 4 4 2 2 4 3	1 4 1 2	1 4 1 2	✓

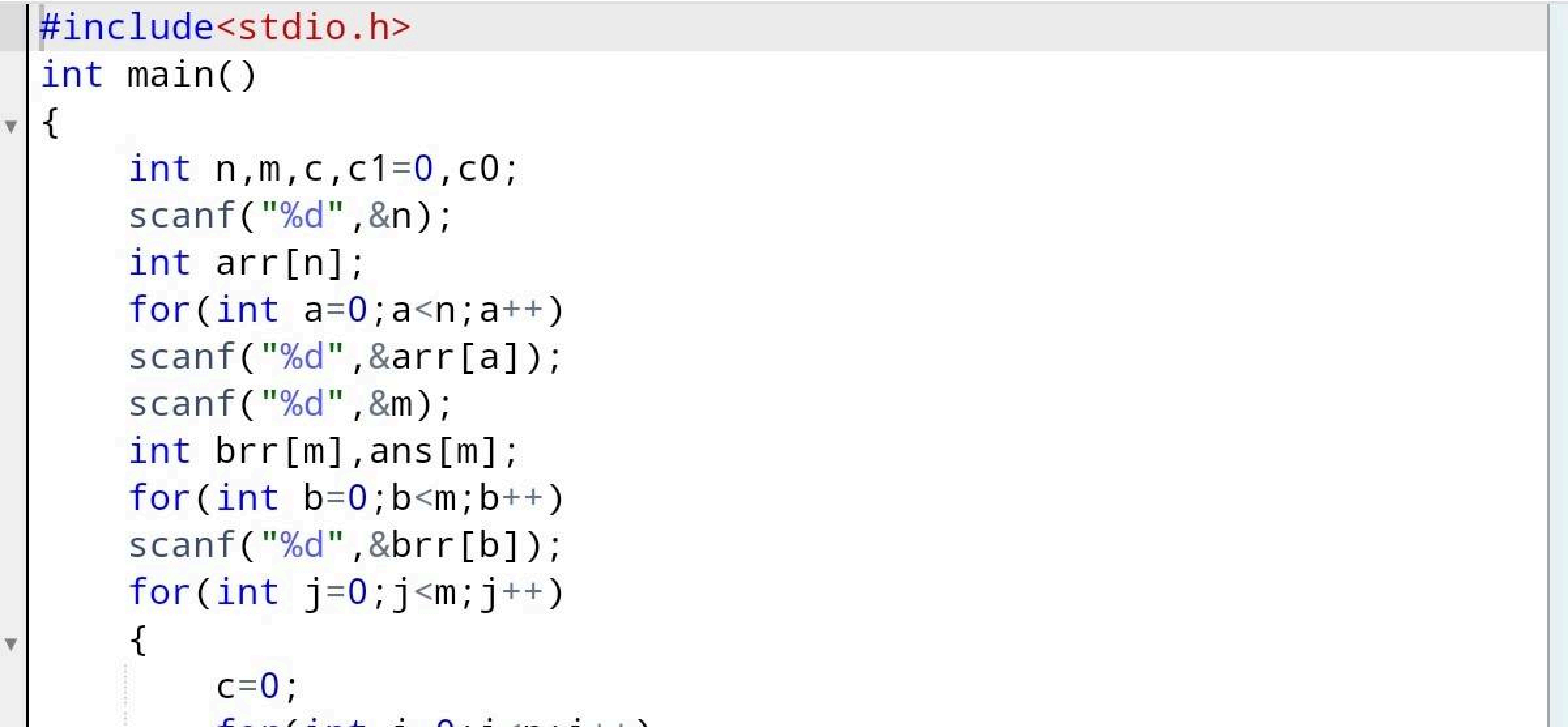
Passed all tests! ✓

Numeros the Artist had two lists that were permutations of one another. He was very proud. Unfortunately, while transporting them from one exhibition to another, some numbers were lost out of the first list. Can you find the missing numbers?

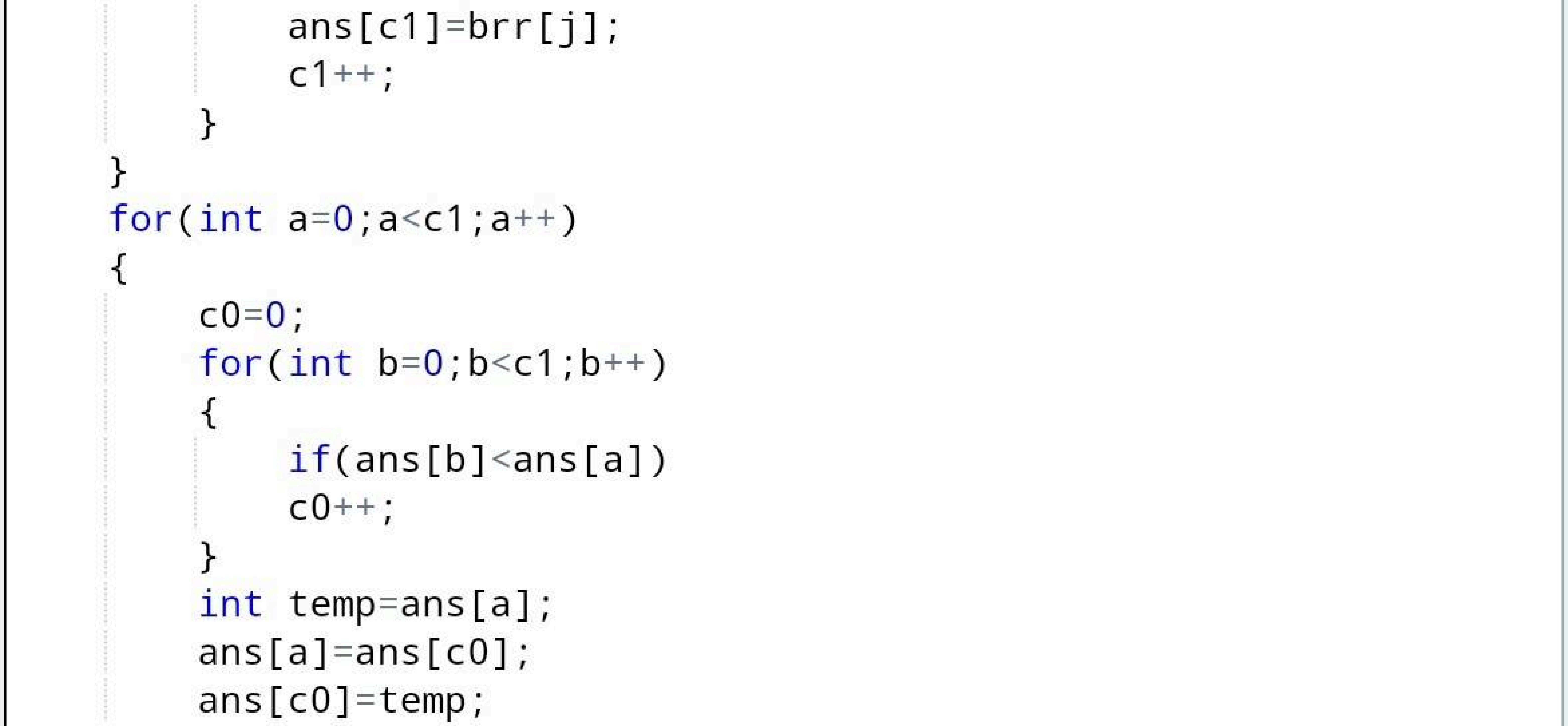
As an example, the array with some numbers missing, arr = [7, 2, 5, 3, 5, 3]. The original array of numbers brr = [7, 2, 5, 4, 6, 3, 5, 3]. The numbers missing are [4, 6].

Notes

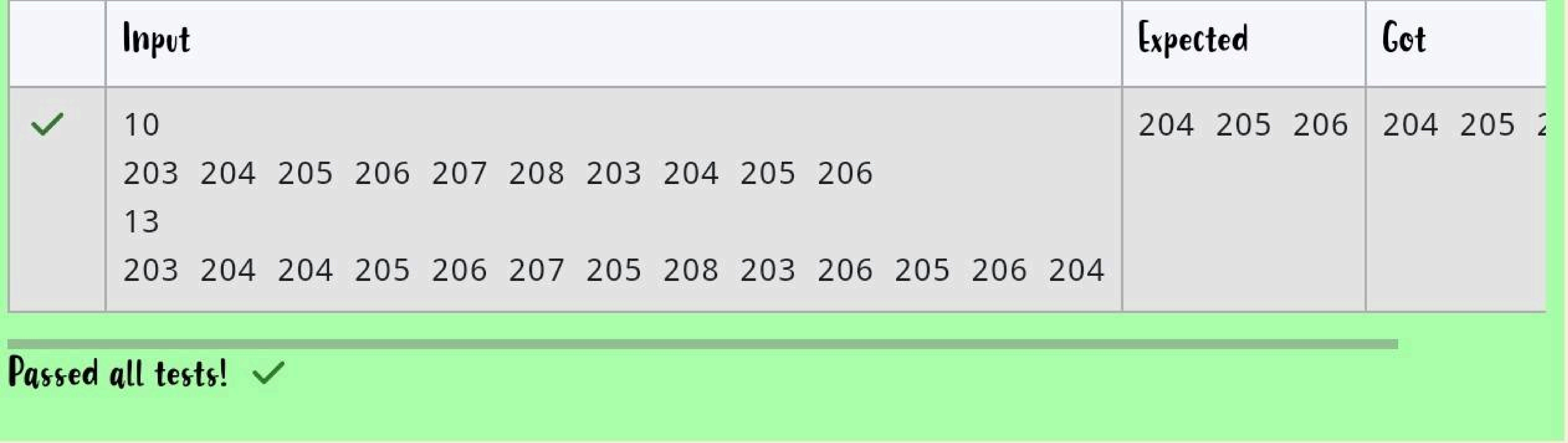
- If a number occurs multiple times in the lists, you must ensure that the frequency of that number in both



```c
#include<stdio.h>
int main()
{
    int n,m,c,c1=0,c0;
    scanf("%d",&n);
    int arr[n];
    for(int a=0;a<n;a++)
    scanf("%d",&arr[a]);
    scanf("%d",&m);
    int brr[m],ans[m];
    for(int b=0;b<m;b++)
    scanf("%d",&brr[b]);
    for(int j=0;j<m;j++)
    {
        c=0;
```



```
            ans[c1]=brr[j];
            c1++;
        }
    }
    for(int a=0;a<c1;a++)
    {
        c0=0;
        for(int b=0;b<c1;b++)
        {
            if(ans[b]<ans[a])
            c0++;
        }
        int temp=ans[a];
        ans[a]=ans[c0];
        ans[c0]=temp;
```



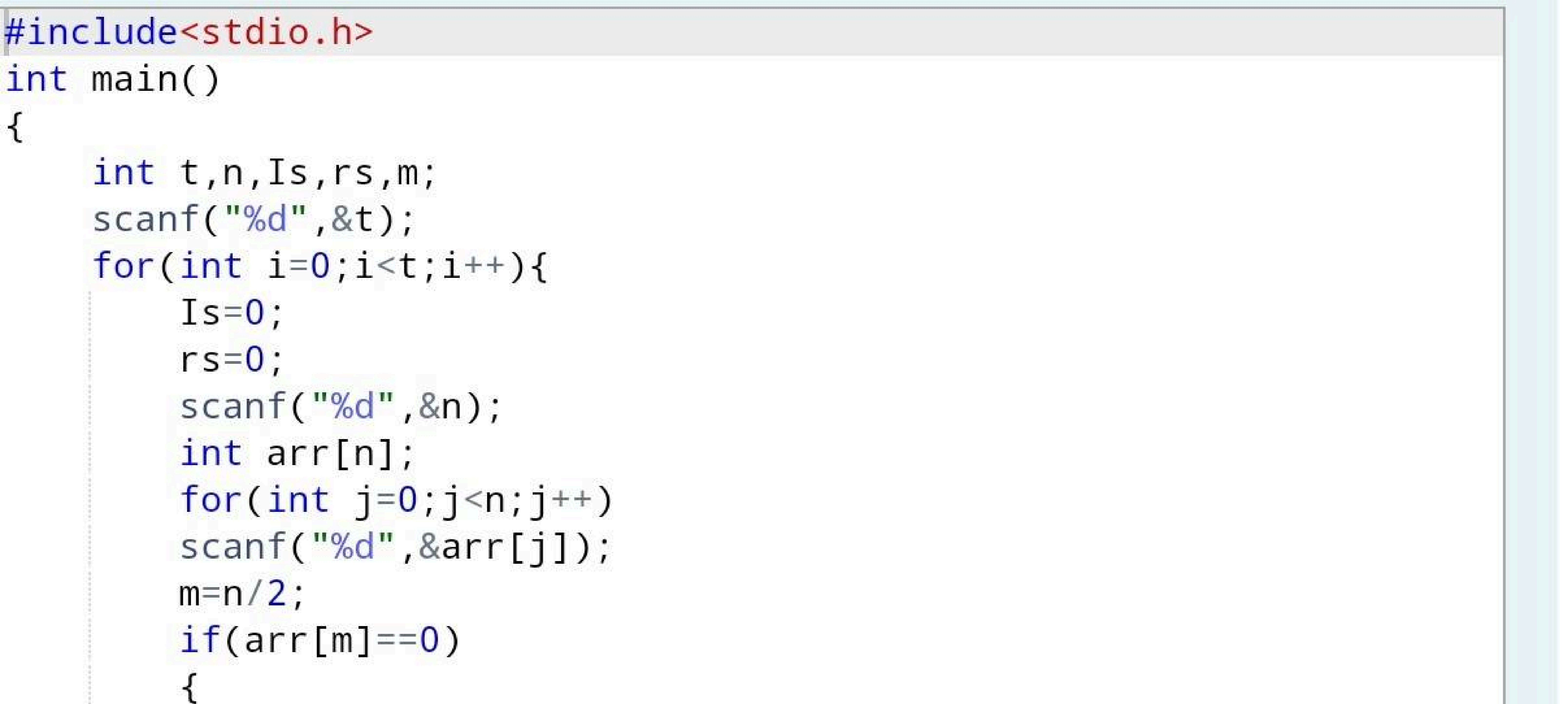
	Input	Expected	Got
✓	10 203 204 205 206 207 208 203 204 205 206 13 203 204 204 205 206 207 205 208 203 206 205 206 204	204 205 206	204 205 2

Passed all tests! ✓

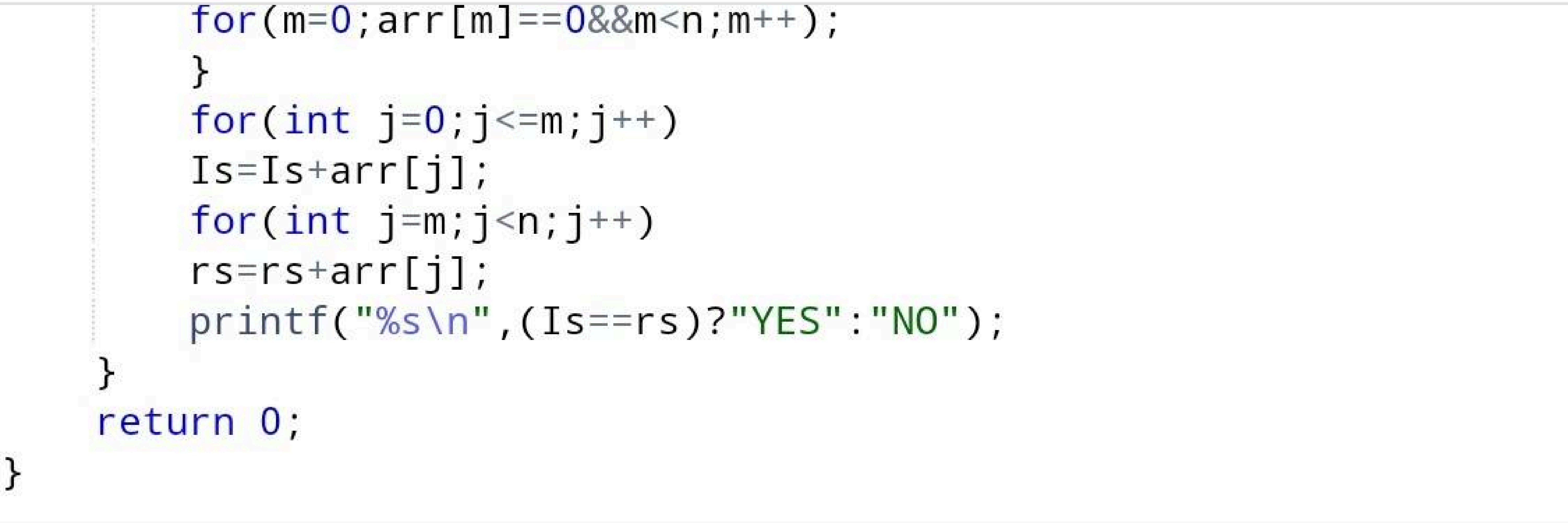
Watson gives Sherlock an array of integers. His challenge is to find an element of the array such that the sum of all elements to the left is equal to the sum of all elements to the right. For instance, given the array arr = [5, 6, 8, 11], 8 is between two subarrays that sum to 11. If your starting array is [1], that element satisfies the rule as left and right sum to 0.

You will be given arrays of integers and must determine whether there is an element that meets the criterion.

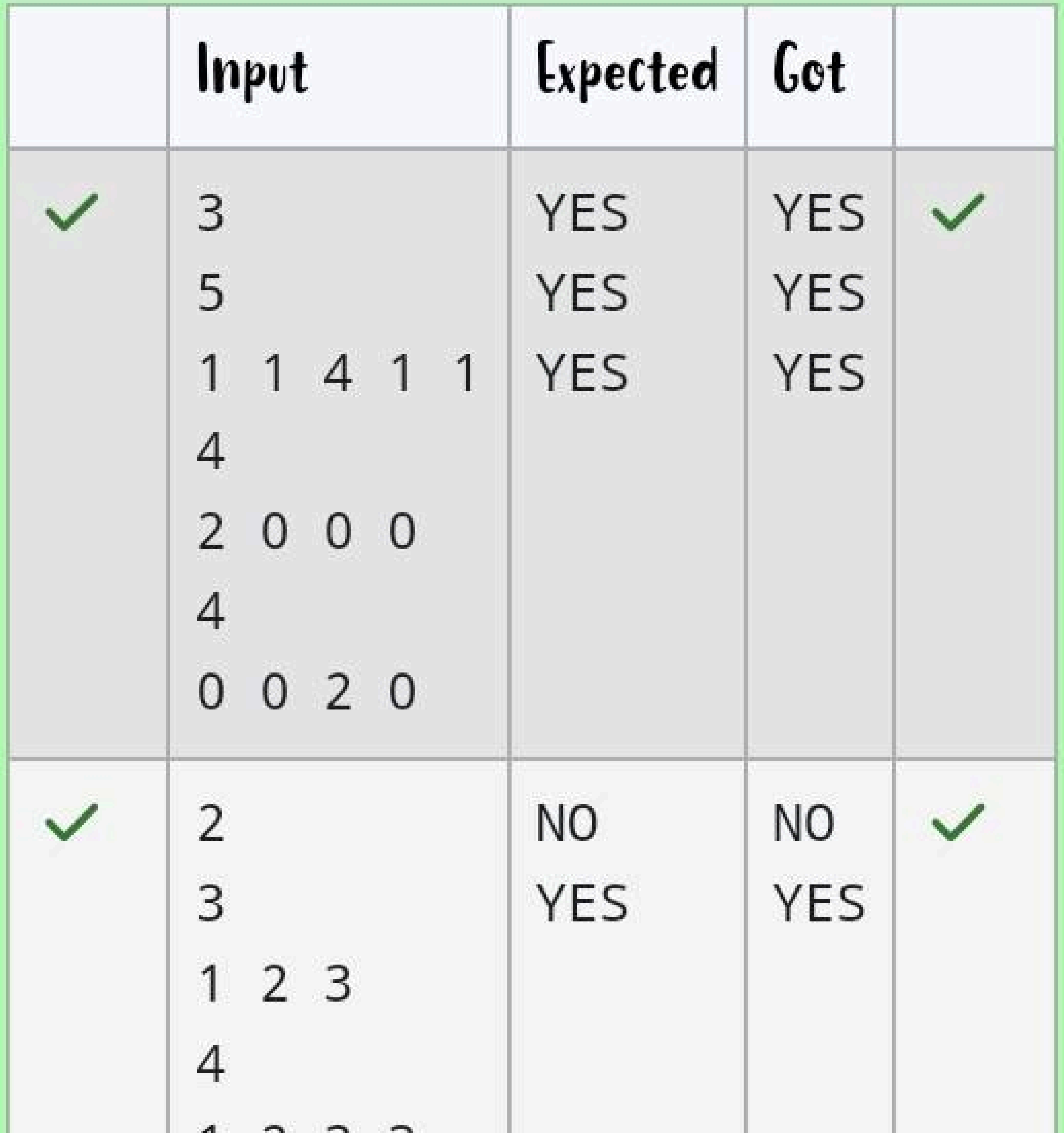
Complete the code in the editor below. It should return a string, either YES if there is an element meeting the



```
#include<stdio.h>
int main()
{
    int t,n,Is,rs,m;
    scanf("%d",&t);
    for(int i=0;i<t;i++){
        Is=0;
        rs=0;
        scanf("%d",&n);
        int arr[n];
        for(int j=0;j<n;j++)
        scanf("%d",&arr[j]);
        m=n/2;
        if(arr[m]==0)
        {
```



```c
        for(m=0;arr[m]==0&&m<n;m++);
        }
        for(int j=0;j<=m;j++)
        Is=Is+arr[j];
        for(int j=m;j<n;j++)
        rs=rs+arr[j];
        printf("%s\n",(Is==rs)?"YES":"NO");
    }
    return 0;
}
```



	Input	Expected	Got	
✓	3	YES	YES	✓
	5	YES	YES	
	1 1 4 1 1	YES	YES	
	4			
	2 0 0 0			
	4			
	0 0 2 0			
✓	2	NO	NO	✓
	3	YES	YES	
	1 2 3			
	4			
	1 [illegible]			