Finding duplicates

1.

Find Duplicate in Array.

Given a read only array of n integers between 1 and n, find one number that repeats.

Input Format:

First Line - Number of elements

n Lines - n Elements

Output Format:

Element x - That is repeated

For example:

Input	Result
5 1 1 2 3 4	1

Algorithm:

1. Start
2. Input integer n
3. Input array arr[n]
4. For i = 0 to n-1;
5. For j = i+1 to n-1:
6. If arr[i] == arr[j], print arr[i] and return
7. End

Program:

```
#include <stdio.h>

int main()

{

   int n;

   scanf("%d", &n);
```

```
    int arr[n];
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }
    for (int i = 0; i < n; i++)
    {
        for (int j = i + 1; j < n; j++)
        {
            if (arr[i] == arr[j]) {

                printf("%d\n", arr[i]);
                return 0;
            }
        }
    }
}
```

	Input	Expected	Got	
✔	11 10 9 7 6 5 1 2 3 8 4 7	7	7	✔
✔	5 1 2 3 4 4	4	4	✔
✔	5 1 1 2 3 4	1	1	✔

Passed all tests! ✔

2.

Find Duplicate in Array.

Given a read only array of n integers between 1 and n, find one number that repeats.

Input Format:

First Line - Number of elements

n Lines - n Elements

Output Format:

Element x - That is repeated

For example:

Input	Result
5 1 1 2 3 4	1

Algorithm:

1. Start
2. Input integer n
3. Input array arr[n]
4. For i = 0 to n-1:
5. For j = i+1 to n-1:
6. If arr[i] == arr[j], print arr[i] and return
7. End

Program:

```
#include <stdio.h>

int main()

{

   int n;

   scanf("%d", &n);
```

```
    int arr[n];
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }
    for (int i = 0; i < n; i++)
    {
        for (int j = i + 1; j < n; j++)
        {
            if (arr[i] == arr[j])
            {
                printf("%d", arr[i]);
                return 0;
            }
        }
    }
}
```

	Input	Expected	Got	
✔	11 10 9 7 6 5 1 2 3 8 4 7	7	7	✔
✔	5 1 2 3 4 4	4	4	✔
✔	5 1 1 2 3 4	1	1	✔

Passed all tests! ✔

3.

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

Algorithm:

1. Start
2. Input integer t (number of test cases)
3. For each test case (t--):
4. Input integers n1, n2 (sizes of arrays)
5. Input arrays arr1[n1] and arr2[n2]
6. For each element element in arr1:
7. For each element x in arr2:
8. If element == x, print element and break

9. End

Program:

```
#include <stdio.h>
void intersection(int arr1[],int n1,int arr2[],int n2){
   for (int i=0;i<n1;i++){
      int element=arr1[i];
      for (int j=0;j<n2;j++){
         if (arr2[j]==element) {
            printf("%d ",element);
            break;
         }
      }
   }
   printf("\n");
}

int main(){
   int t;
   scanf("%d",&t);
   while(t--){
      int n1,n2;
      scanf("%d",&n1);
      int arr1[n1];
      for(int i=0;i<n1;i++){
         scanf("%d",&arr1[i]);
      }
      scanf("%d",&n2);
```

```
        int arr2[n2];
        for(int i=0;i<n2;i++){
            scanf("%d",&arr2[i]);
        }
        intersection(arr1,n1,arr2,n2);
    }
}
```

	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! ✔

4.

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

Algorithm:

1. Input number of test cases t
2. For each test case:
3. Input arrays arr1[] and arr2[]
4. Initialize i = 0, j = 0
5. While i < n1 and j < n2:
6. If arr1[i] < arr2[j], increment i
7. Else if arr1[i] > arr2[j], increment j
8. Else, print arr1[i], and increment both i and j

9. End

Program:

```
#include <stdio.h>
void intersection(int arr1[], int n1, int arr2[], int n2) {
   int i=0,j=0;
   while (i<n1 && j<n2){
      if (arr1[i]<arr2[j]){
         i++;
      }
      else if (arr2[j]<arr1[i]){
         j++;
      }
      else{
         printf("%d ",arr1[i]);
         i++;
         j++;
      }
   }
   printf("\n");
}

int main(){
   int t;
   scanf("%d",&t);
   while (t--){
      int n1,n2;
      scanf("%d", &n1);
      int arr1[n1];
```

```
        for (int i=0;i<n1;i++){
            scanf("%d",&arr1[i]);
        }
        scanf("%d",&n2);
        int arr2[n2];
        for (int i=0;i<n2;i++){
            scanf("%d", &arr2[i]);
        }
        intersection(arr1,n1,arr2,n2);
    }
}
```

	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! ✔

5.

Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that A[j] - A[i] = k, i != j.

Input Format:

First Line n - Number of elements in an array

Next n Lines - N elements in the array

k - Non - Negative Integer

Output Format:

1 - If pair exists

0 - If no pair exists

Explanation for the given Sample Testcase:

YES as 5 - 1 = 4

So Return 1.

For example:

Input	Result
3 1 3 5 4	1

Algorithm:

1. Start
2. Input: Read integer n, array arr[] of size n, and integer k.
3. For each element arr[i] from index 0 to n-1:
4. For each subsequent element arr[j] from index i+1 to n-1:
5. If arr[j] - arr[i] == k, return 1 (pair found).
6. If arr[j] - arr[i] > k, break the inner loop.
7. If no valid pair is found, return 0.
8. Output the result (1 or 0).
9. End

Program:

```
#include <stdio.h>

int checkpair(int arr[],int n,int k){

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

      for (int j=i+1;j<n;j++){

         if(arr[j]-arr[i]==k){
```

```
            return 1;
        }
        else if(arr[j]-arr[i]>k){
            break;
        }
      }
   }
   return 0;
}

int main(){
   int n, k;
   scanf("%d", &n);
   int arr[n];
   for (int i=0;i<n;i++) {
      scanf("%d",&arr[i]);
   }
   scanf("%d",&k);
   int result=checkpair(arr,n,k);
   printf("%d\n",result);
}
```

	Input	Expected	Got	
✔	3 1 3 5 4	1	1	✔
✔	10 1 4 6 8 12 14 15 20 21 25 1	1	1	✔
✔	10 1 2 3 5 11 14 16 24 28 29 0	0	0	✔
✔	10 0 2 3 7 13 14 15 20 24 25 10	1	1	✔

Passed all tests! ✔

6.

Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that A[j] - A[i] = k, i != j.

Input Format:

First Line n - Number of elements in an array

Next n Lines - N elements in the array

k - Non - Negative Integer

Output Format:

1 - If pair exists

0 - If no pair exists

Explanation for the given Sample Testcase:

YES as 5 - 1 = 4

So Return 1.

For example:

Input	Result
3 1 3 5 4	1

Algorithm:

1. Start
2. Read integer n and integer k Input: Read array arr[] of size n.
3. Initialize two pointers: i = 0, j = 1.
4. While j < n:
5. Compute the difference diff = arr[j] - arr[i].
6. If diff == k and i != j, return 1 (pair found).
7. If diff < k, increment j.
8. Else, increment i.
9. If i == j, increment j
10. If no valid pair is found, return 0.
11. Print the result
12. End

Program:

```
#include <stdio.h>

int checkpair(int arr[],int n,int k){
```

```
    int i=0,j=1;
    while(j<n){
        int diff=arr[j]-arr[i];
        if (diff==k && i!=j){
            return 1;
        }
        else if(diff<k){
            j++;
        }
        else{
            i++;
        }
        if(i==j){
            j++;
        }
    }
    return 0;
}
   int main(){
    int n,k;
    scanf("%d",&n);
    int arr[n];
    for (int i=0;i<n;i++){
        scanf("%d",&arr[i]);
    }
    scanf("%d",&k);
    int result=checkpair(arr,n,k);
    printf("%d\n",result);
}
```

	Input	Expected	Got	
✔	3 1 3 5 4	1	1	✔
✔	10 1 4 6 8 12 14 15 20 21 25 1	1	1	✔
✔	10 1 2 3 5 11 14 16 24 28 29 0	0	0	✔
✔	10 0 2 3 7 13 14 15 20 24 25 10	1	1	✔

Passed all tests! ✔