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

Started on	Wednesday, 8 October 2025, 3:22 PM
State	Finished
Completed on	Wednesday, 8 October 2025, 3:22 PM
Time taken	32 secs
Marks	1.00/1.00
Grade	**4.00** out of 4.00 (**100**%)

Question 1 | Correct Mark 1.00 out of 1.00 Flag question

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

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);

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

    int slow = arr[0];
    int fast = arr[arr[0]];

    // Find intersection point in cycle
    while (slow != fast) {
        slow = arr[slow];
        fast = arr[arr[fast]];
    }

    // Find entrance to cycle (duplicate)
    slow = 0;
    while (slow != fast) {
        slow = arr[slow];
        fast = arr[fast];
    }

    printf("%d\n", slow);

    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! ✔

Correct

Marks for this submission: 1.00/1.00.

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

Started on	Wednesday, 8 October 2025, 3:22 PM
State	Finished
Completed on	Wednesday, 8 October 2025, 3:23 PM
Time taken	33 secs
Marks	1.00/1.00
Grade	**4.00** out of 4.00 (**100**%)

Question 1 Correct Mark 1.00 out of 1.00 Flag question

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

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);

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

    int slow = arr[0];
    int fast = arr[arr[0]];

    // Detect cycle intersection
    while(slow != fast) {
        slow = arr[slow];
        fast = arr[arr[fast]];
    }

    // Find start of cycle (duplicate)
    slow = 0;
    while(slow != fast) {
        slow = arr[slow];
        fast = arr[fast];
    }

    printf("%d\n", slow);

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

Started on	Wednesday, 8 October 2025, 3:23 PM
State	Finished
Completed on	Wednesday, 8 October 2025, 3:24 PM
Time taken	44 secs
Marks	1.00/1.00
Grade	**30.00** out of 30.00 (**100**%)

Question 1 | Correct Mark 1.00 out of 1.00 Flag question

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 %)

```c
#include <stdio.h>

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]);
        }

        int i = 0, j = 0;
        int first = 1;  // flag to avoid extra spaces before first element

        while (i < N1 && j < N2) {
            if (arr1[i] == arr2[j]) {
                if (!first) printf(" ");
                printf("%d", arr1[i]);
                first = 0;
                i++;
                j++;
            } else if (arr1[i] < arr2[j]) {
                i++;
            } else {
                j++;
            }
        }
        printf("\n");
    }

    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.

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

Started on	Wednesday, 8 October 2025, 3:24 PM
State	Finished
Completed on	Wednesday, 8 October 2025, 3:25 PM
Time taken	53 secs
Marks	1.00/1.00
Grade	**30.00** out of 30.00 (**100**%)

Question 1 | Correct Mark 1.00 out of 1.00 Flag question

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 %)

```c
#include <stdio.h>

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]);
        }

        int i = 0, j = 0;
        int first = 1;  // to avoid leading space
        while (i < N1 && j < N2) {
            if (arr1[i] == arr2[j]) {
                if (!first) printf(" ");
                printf("%d", arr1[i]);
                first = 0;
                i++;
                j++;
            } else if (arr1[i] < arr2[j]) {
                i++;
            } else {
                j++;
            }
        }
        printf("\n");
    }

    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.

5-Pair with Difference-O(n^2)Time Complexity,O(1) Space Complexity

Started on	Wednesday, 8 October 2025, 3:25 PM
State	Finished
Completed on	Wednesday, 8 October 2025, 3:26 PM
Time taken	39 secs
Marks	1.00/1.00
Grade	**4.00** out of 4.00 (**100**%)

Question 1 | Correct Mark 1.00 out of 1.00 Flag question

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

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

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

    int k;
    scanf("%d", &k);

    int i = 0, j = 1;
    int found = 0;

    while(j < n && i < n) {
        if (i != j) {
            int diff = A[j] - A[i];
            if(diff == k) {
                found = 1;
                break;
            }
            else if(diff < k) {
                j++;
            }
            else {
                i++;
                // Ensure i != j
                if(i == j) j++;
            }
        } else {
            j++;
        }
    }

    printf("%d\n", found);

    return 0;
}
```

	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! ✔

Correct

Marks for this submission: 1.00/1.00

6-Pair with Difference -O(n) Time Complexity,O(1) Space Complexity

Started on	Wednesday, 8 October 2025, 3:26 PM
State	Finished
Completed on	Wednesday, 8 October 2025, 3:26 PM
Time taken	42 secs
Marks	1.00/1.00
Grade	**4.00** out of 4.00 (**100**%)

Question 1 | Correct Mark 1.00 out of 1.00 Flag question

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

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);
    int A[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &A[i]);
    }
    int k;
    scanf("%d", &k);

    int i = 0, j = 1;
    while (j < n && i < n) {
        int diff = A[j] - A[i];
        if (diff == k && i != j) {
            printf("1\n");
            return 0;
        } else if (diff < k) {
            j++;
        } else {
            i++;
            if (i == j) j++;
        }
    }

    printf("0\n");
    return 0;
}
```

	Input	Expected	Got	
✔	3 1 3 5 4	1	1	✔
✔	10 1 4 6 8 12 14 15 20 21 25 1	1	1	✔