

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 1 2 3 4	

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6
7     int arr[n];
8     for(int i=0; i<n; i++) {
9         scanf("%d", &arr[i]);
10    }
11
12    int slow = arr[0];
13    int fast = arr[arr[0]];
14
15    // Find intersection point in cycle
16    while (slow != fast) {
17        slow = arr[slow];
18        fast = arr[arr[fast]];
19    }
20
21    // Find entrance to cycle (duplicate)
22    slow = 0;
23    while (slow != fast) {
24        slow = arr[slow];
25        fast = arr[fast];
26    }
27
28    printf("%d\n", slow);
29
30    return 0;
31}
32
```

	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 1 2 3 4	

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6
7     int arr[n];
8     for(int i = 0; i < n; i++) {
9         scanf("%d", &arr[i]);
10    }
11
12    int slow = arr[0];
13    int fast = arr[arr[0]];
14
15    // Detect cycle intersection
16    while(slow != fast) {
17        slow = arr[slow];
18        fast = arr[arr[fast]];
19    }
20
21    // Find start of cycle (duplicate)
22    slow = 0;
23    while(slow != fast) {
24        slow = arr[slow];
25        fast = arr[arr[fast]];
26    }
27
28    printf("%d\n", slow);
29
30    return 0;
31}
32|
```

	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:
 - Line 1 contains N1, followed by N1 integers of the first array
 - 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	10 57
3 10 17 57	
6	
2 7 10 15 57 246	

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int T;
5     scanf("%d", &T);
6
7     while (T--) {
8         int N1;
9         scanf("%d", &N1);
10        int arr1[N1];
11        for (int i = 0; i < N1; i++) {
12            scanf("%d", &arr1[i]);
13        }
14
15        int N2;
16        scanf("%d", &N2);
17        int arr2[N2];
18        for (int i = 0; i < N2; i++) {
19            scanf("%d", &arr2[i]);
20        }
21
22        int i = 0, j = 0;
23        int first = 1; // flag to avoid extra spaces before first element
24
25        while (i < N1 && j < N2) {
26            if (arr1[i] == arr2[j]) {
27                if (!first) printf(" ");
28                printf("%d", arr1[i]);
29                first = 0;
30                i++;
31                j++;
32            } else if (arr1[i] < arr2[j]) {
33                i++;
34            } else {
35                j++;
36            }
37        }
38        printf("\n");
39    }
40
41    return 0;
42 }
43 }
```

	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:
 - Line 1 contains N1, followed by N1 integers of the first array
 - 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	10 57
3 10 17 57	
6	
2 7 10 15 57 246	

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int T;
5     scanf("%d", &T);
6
7     while (T--) {
8         int N1;
9         scanf("%d", &N1);
10        int arr1[N1];
11        for (int i = 0; i < N1; i++) {
12            scanf("%d", &arr1[i]);
13        }
14
15        int N2;
16        scanf("%d", &N2);
17        int arr2[N2];
18        for (int i = 0; i < N2; i++) {
19            scanf("%d", &arr2[i]);
20        }
21
22        int i = 0, j = 0;
23        int first = 1; // to avoid leading space
24        while (i < N1 && j < N2) {
25            if (arr1[i] == arr2[j]) {
26                if (!first) printf(" ");
27                printf("%d", arr1[i]);
28                first = 0;
29                i++;
30                j++;
31            } else if (arr1[i] < arr2[j]) {
32                i++;
33            } else {
34                j++;
35            }
36        }
37        printf("\n");
38    }
39
40    return 0;
41 }
```

	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 $A[j] - A[i] = k$, $i \neq j$.

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
1 3 5	
4	

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6     int A[n];
7     for(int i = 0; i < n; i++) {
8         scanf("%d", &A[i]);
9     }
10
11    int k;
12    scanf("%d", &k);
13
14    int i = 0, j = 1;
15    int found = 0;
16
17    while(j < n && i < n) {
18        if (i != j) {
19            int diff = A[j] - A[i];
20            if(diff == k) {
21                found = 1;
22                break;
23            }
24            else if(diff < k) {
25                j++;
26            }
27            else {
28                i++;
29                // Ensure i != j
30                if(i == j) j++;
31            }
32        } else {
33            j++;
34        }
35    }
36
37    printf("%d\n", found);
38
39    return 0;
40}
41
```

	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
1 3 5	
4	

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6     int A[n];
7     for (int i = 0; i < n; i++) {
8         scanf("%d", &A[i]);
9     }
10    int k;
11    scanf("%d", &k);
12
13    int i = 0, j = 1;
14    while (j < n && i < n) {
15        int diff = A[j] - A[i];
16        if (diff == k && i != j) {
17            printf("1\n");
18            return 0;
19        } else if (diff < k) {
20            j++;
21        } else {
22            i++;
23            if (i == j) j++;
24        }
25    }
26
27    printf("0\n");
28    return 0;
29 }
```

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