Name	Aravinthaa S
Register No	230701032
Dept	CSE A

COMPETITIVE PROGRAMMING

Ex No – 5.1	FINDING DUPLICATES-O(n^2) TIME COMPLEXITY,O(1) SPACE COMPLEXITY

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

Aim:

To find a duplicate element in an array of n integers where the elements range between 1 and n. The solution should use O(n^2) time complexity and O(1) space complexity.

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 and Output:

	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! ✔

Ex No – 5.2	FINDING DUPLICATES-O(n) TIME COMPLEXITY,O(1) SPACE COMPLEXITY

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

Aim:

To find one repeated element in an array of n integers, where the integers are between 1 and n.

Program:

```
#include <stdio.h>

int main() {
   int n;
   scanf("%d", &n);
   int seen[n + 1];

   for (int i = 0; i <= n; i++) {
      seen[i] = 0;
   }
```

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

    return 0;
}
```

Input and Output:

	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! ✔

Ex No – 5.3	PRINT INTERSECTION OF 2 SORTED ARRAYS - O(m*n) TIME COMPLEXITY, O(1) SPACE COMPLEXITY.

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

Aim:

To find the intersection of two sorted arrays.

Program:

#include <stdio.h>

```
void findIntersection(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 (arr1[i] > arr2[j]) {
            j++;
        } else {
            printf("%d ", arr1[i]);
            i++;
            j++;
        }
    }
}

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]);
        }

        findIntersection(arr1, n1, arr2, n2);
        printf("\n");
    }

    return 0;
}
```

Input and Output:

	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! ✔

Ex No – 5.4	PRINT INTERSECTION OF 2 SORTED ARRAYS - O(m+n) TIME COMPLEXITY, O(1) SPACE COMPLEXITY.

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

Aim:

To find the intersection of two sorted arrays, meaning identifying all the elements that appear in both arrays, and print them.

Program:

```
#include <stdio.h>

void findIntersection(int arr1[], int n1, int arr2[], int n2) {
```

```
    int i = 0, j = 0;
    while (i < n1 && j < n2) {
        if (i > 0 && arr1[i] == arr1[i - 1]) {
            i++;
            continue;
        }

        if (arr1[i] < arr2[j]) {
            i++;
        } else if (arr1[i] > arr2[j]) {
            j++;
        } else {
            printf("%d ", arr1[i]);
            i++;
            j++;
        }
    }
}

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]);
        }

        findIntersection(arr1, n1, arr2, n2);
        printf("\n");
    }

    return 0;
}
```

Input and Output:

	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! ✔

Ex No – 5.5	PAIR WITH DIFFERENCE - O(n^2) TIME COMPLEXITY, O(1) SPACE COMPLEXITY.

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

Aim:

To determine if there exist two indices iii and jjj in a sorted array such that the difference A[j]−A[i]=kA[j] - A[i] = kA[j]−A[i]=k, where kkk is a given non-negative integer, and i≠ji \neq ji=j. Return 1 if such a pair exists, otherwise return 0.

Program:

```
#include <stdio.h>


int pairWithDifference(int A[], int n, int k) {
   for (int i = 0; i < n; i++) {
      for (int j = i + 1; j < n; j++) {
         if (A[j] - A[i] == k) {
            return 1;
         }
      }
   }
```

```
    return 0;
}

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 result = pairWithDifference(A, n, k);
    printf("%d\n", result);

    return 0;
}
```

Input and Output:

	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! ✔

Ex No – 5.6	PAIR WITH DIFFERENCE - O(n) TIME COMPLEXITY, O(1) SPACE COMPLEXITY.

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

Aim:

To check if there exists a pair of indices iii and jjj in a sorted array such that the difference A[j]−A[i]=kA[j] - A[i] = kA[j]−A[i]=k, where kkk is a non-negative integer. If such a pair exists, return 1; otherwise, return 0.

Program:

```
#include <stdio.h>


int pairWithDifference(int A[], int n, int k) {
   int i = 0, j = 0;


   while (j < n) {
      int diff = A[j] - A[i];


      if (diff == k && i != j) {
```

```
            return 1;
        }

        if (diff < k) {
            j++;
        } else {
            i++;
        }

        if (i == j) {
            j++;
        }
    }

    return 0;
}

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 result = pairWithDifference(A, n, k);
    printf("%d\n", result);
```

```
    return 0;
}
```

Input and Output:

	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! ✔