

NAME: KAVINRAJ.S

REG NO: 241801120

COMPETITIVE PROGRAMMING

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

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 int main()
3 {
4     int n;
5     scanf("%d", &n);
6     int arr[n];
7     for (int i = 0; i < n; i++)
8     {
9         scanf("%d", &arr[i]);
10    }
11    int duplicate = -1;
12    for (int i = 0; i < n; i++)
13    {
14        for (int j = i + 1; j < n; j++)
15        {
16            if (arr[i] == arr[j]) {
17                duplicate = arr[i];
18                break;
19            }
20        }
21        if (duplicate != -1)
22            break;
23    }
24    if (duplicate != -1)
25        printf("%d\n", duplicate);
26    else
27        printf("No duplicate found\n");
28    return 0;
29 }
30
31 }
```

	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.

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

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 int main()
3 {
4     int n;
5     scanf("%d", &n);
6     int arr[n];
7     for (int i = 0; i < n; i++)
8         scanf("%d", &arr[i]);
9     int slow = arr[0];
10    int fast = arr[0];
11    do
12    {
13        slow = arr[slow];
14        fast = arr[arr[fast]];
15    } while (slow != fast);
16    slow = arr[0];
17    while (slow != fast)
18    {
19        slow = arr[slow];
20        fast = arr[fast];
21    }
22    printf("%d\n", slow);
23    return 0;
24 }
```

	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! ✓

PROGRAM 3: Print Intersection of 2 sorted arrays-O(m*n)Time Complexity,O(1) Space Complexity

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
```

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2 int main()
3 {
4     int t;
5     scanf("%d", &t);
6     while (t--)
7     {
8         int n1, n2;
9         scanf("%d", &n1);
10        int arr1[n1];
11        for (int i = 0; i < n1; i++)
12            scanf("%d", &arr1[i]);
13        scanf("%d", &n2);
14        int arr2[n2];
15        for (int i = 0; i < n2; i++)
16            scanf("%d", &arr2[i]);
17        for (int i = 0; i < n1; i++)
18        {
19            for (int j = 0; j < n2; j++)
20            {
21                if (arr1[i] == arr2[j])
22                {
23                    printf("%d ", arr1[i]);
24                    break;
25                }
26            }
27        }
28        printf("\n");
29    }
30    return 0;
31 }
32 }
```

	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.

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

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

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2 int main()
3 {
4     int t;
5     scanf("%d", &t);
6     while (t--)
7     {
8         int n1, n2;
9         scanf("%d", &n1);
10        int arr1[n1];
11        for (int i = 0; i < n1; i++)
12            scanf("%d", &arr1[i]);
13        scanf("%d", &n2);
14        int arr2[n2];
15        for (int i = 0; i < n2; i++)
16            scanf("%d", &arr2[i]);
17        int i = 0, j = 0;
18        while (i < n1 && j < n2)
19        {
20            if (arr1[i] < arr2[j])
21                i++;
22            else if (arr2[j] < arr1[i])
23                j++;
24            else {
25                printf("%d ", arr1[i]);
26                i++;
27                j++;
28            }
29        }
30        printf("\n");
31    }
32    return 0;
33 }
```

	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.

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

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.

Answer: (penalty regime: 0 %)

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

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

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

	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.