DESIGN AND ANALYSIS OF ALGORITHMS

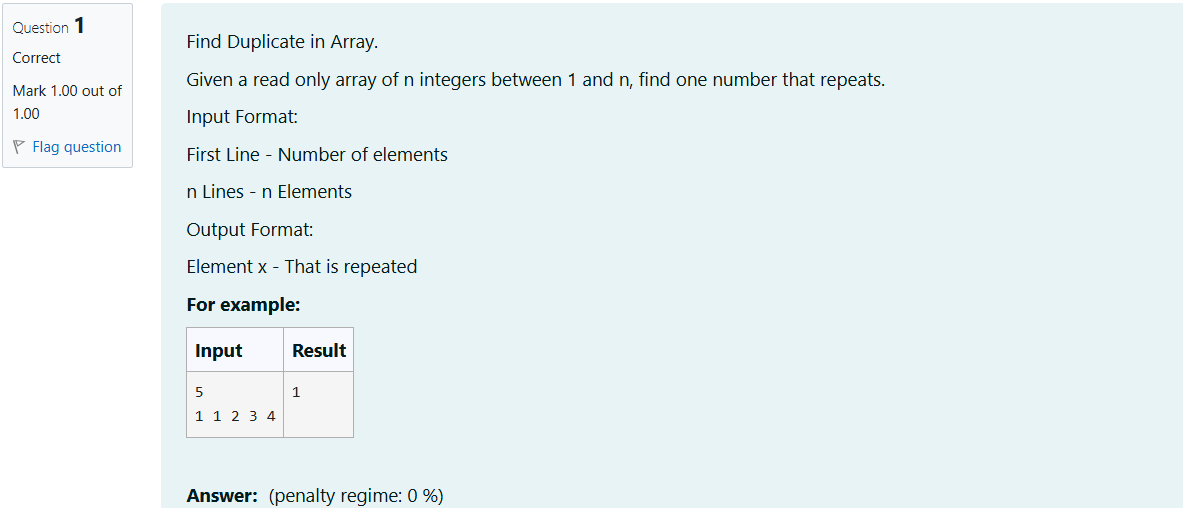
**NAME : P.R.DHANVIN**

**DEPT & SEC : CSE & ‘B’**

**ROLL NO : 230701071**

**WEEK : COMPETITIVE PROGRAMMING**

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



#include<stdio.h>

int main()

{

int n,c=0;

scanf("%d",&n);

int a[n];

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

{

scanf("%d",&a[i]);

}

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

{

for(int j=0;j<n;j++)

{

if (i!=j&&a[i]==a[j])

{

c=c+a[i];

break;

}

}

if(c>0)

{

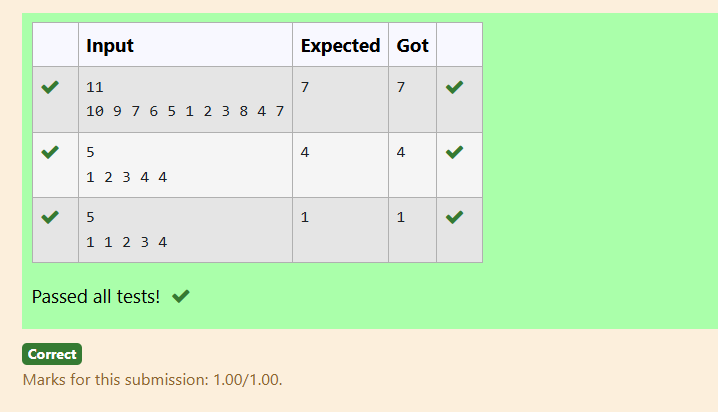
break;

}

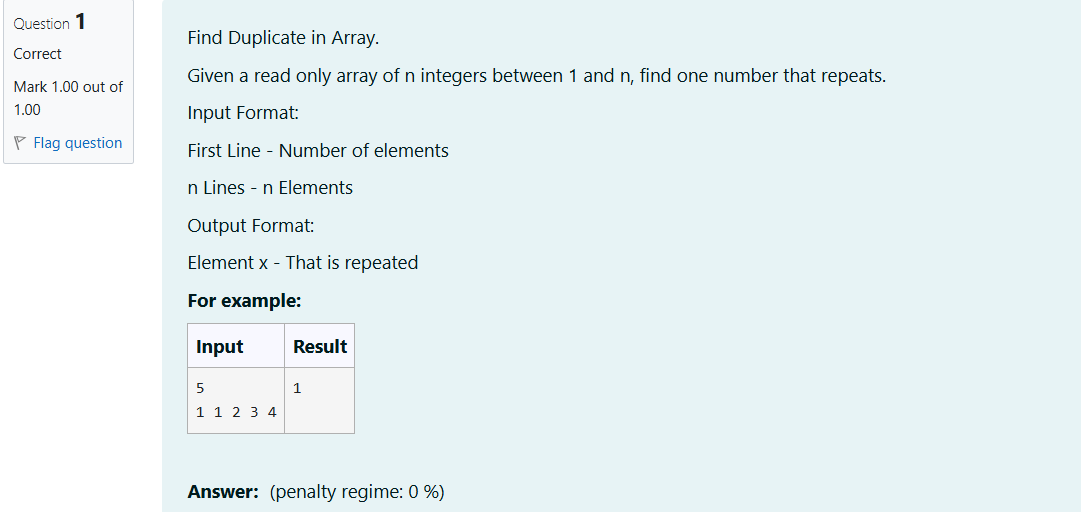
}

printf("%d",c);

}



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



#include<stdio.h>

int main()

{

int n;

scanf("%d",&n);

int a[n];

int ind,b;

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

{

scanf("%d",&b);

ind=b%n;

if(a[ind]!=0 && a[ind]==b) {

printf("%d",b);

break;

}

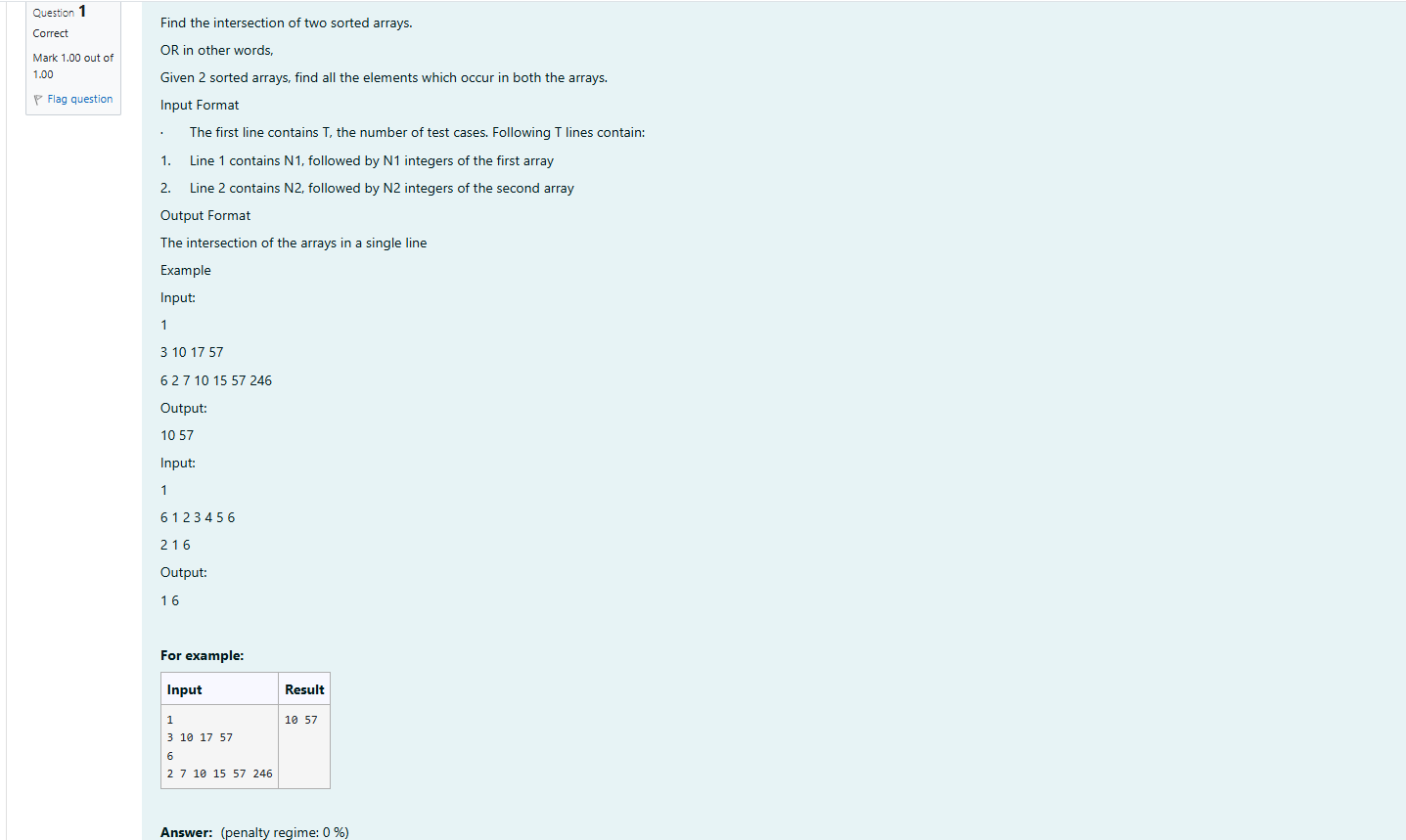
a[ind]=b;

}

}



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



#include <stdio.h>

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

int i = 0, j = 0;

int first = 1;

while (i < n1 && j < n2) {

if (arr1[i] < arr2[j]) {

i++;

} else if (arr1[i] > arr2[j]) {

j++;

} else {

if (first) {

printf("%d", arr1[i]);

first = 0;

} else {

printf(" %d", arr1[i]);

}

i++;

j++;

}

}

if (first) {

printf(" ");

}

}

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]);

}

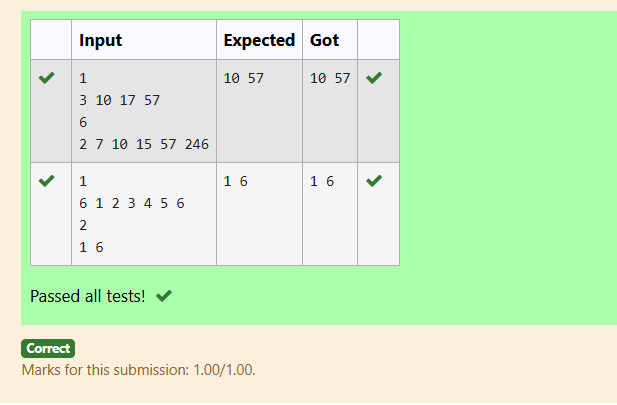
findIntersection(arr1, n1, arr2, n2);

printf("\n");

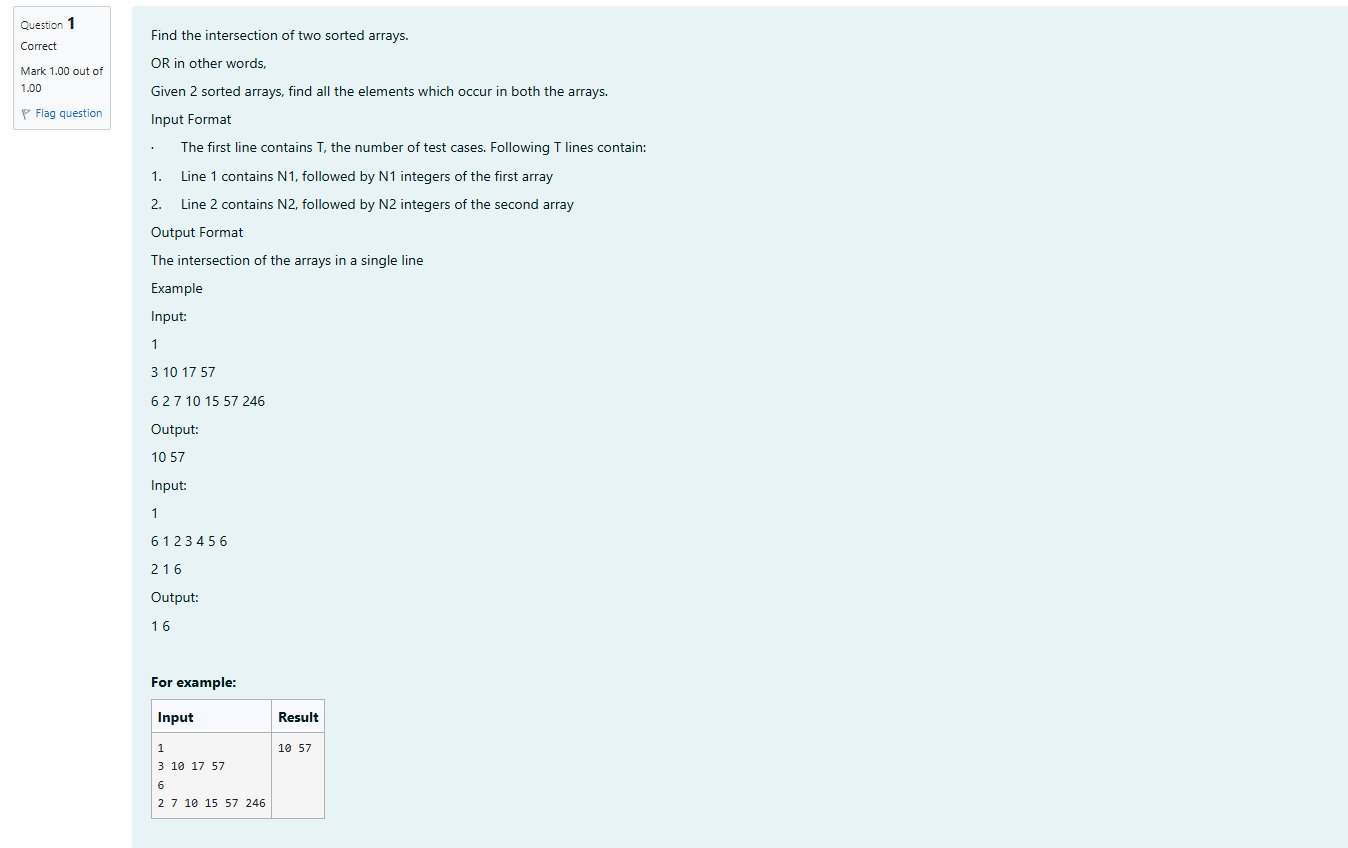
}

return 0;

}



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



#include <stdio.h>

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

int i = 0, j = 0;

int first = 1;

while (i < n1 && j < n2) {

if (arr1[i] < arr2[j]) {

i++;

} else if (arr1[i] > arr2[j]) {

j++;

} else {

if (first) {

printf("%d", arr1[i]);

first = 0;

} else {

printf(" %d", arr1[i]);}

i++;

j++;

}

}

}

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]);

}

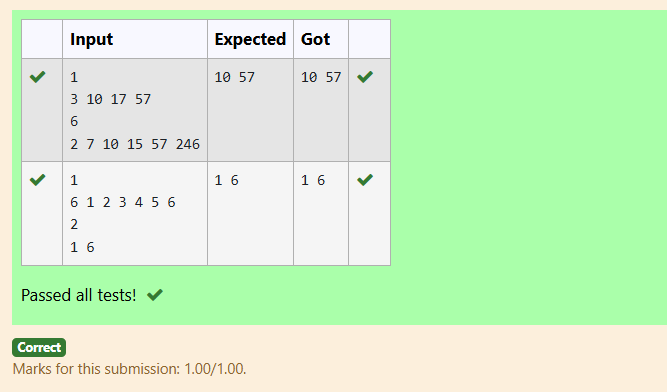
findIntersection(arr1, n1, arr2, n2);

printf("\n");

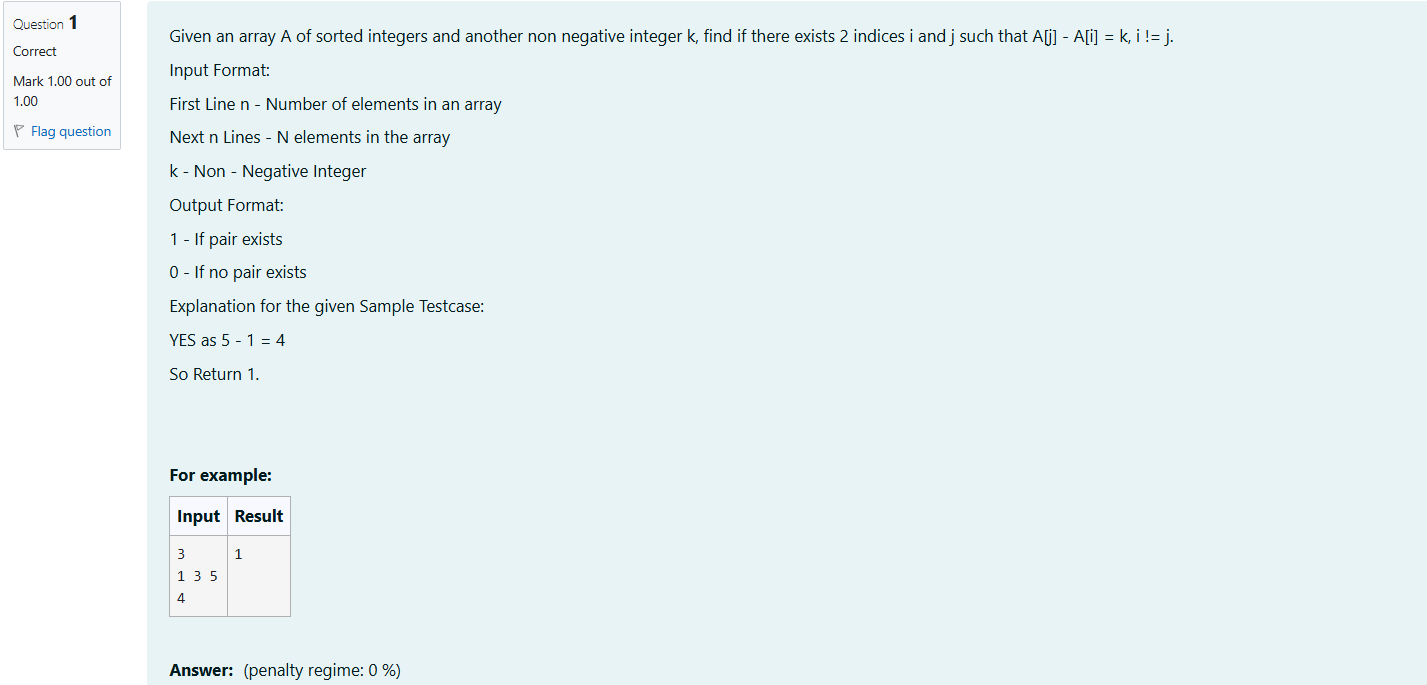
}

return 0;

}



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



#include <stdio.h>

int main() {

int n, k;

scanf("%d", &n);

int a[n];

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

scanf("%d", &a[i]);

}

scanf("%d", &k);

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

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

if (a[j] - a[i] == k) {

printf("1\n");

return 0;}

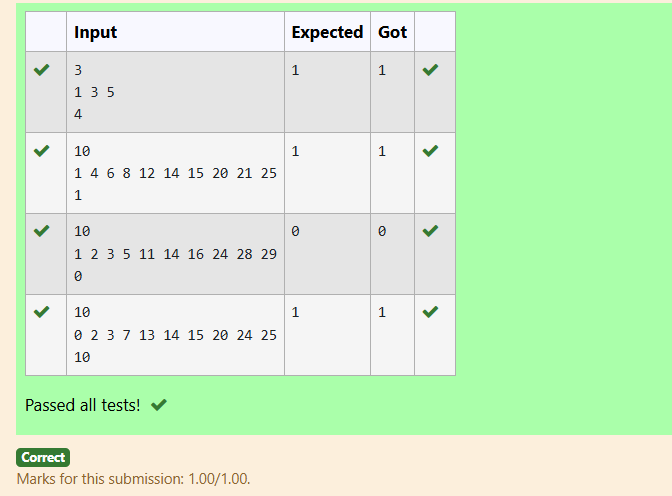
}

}

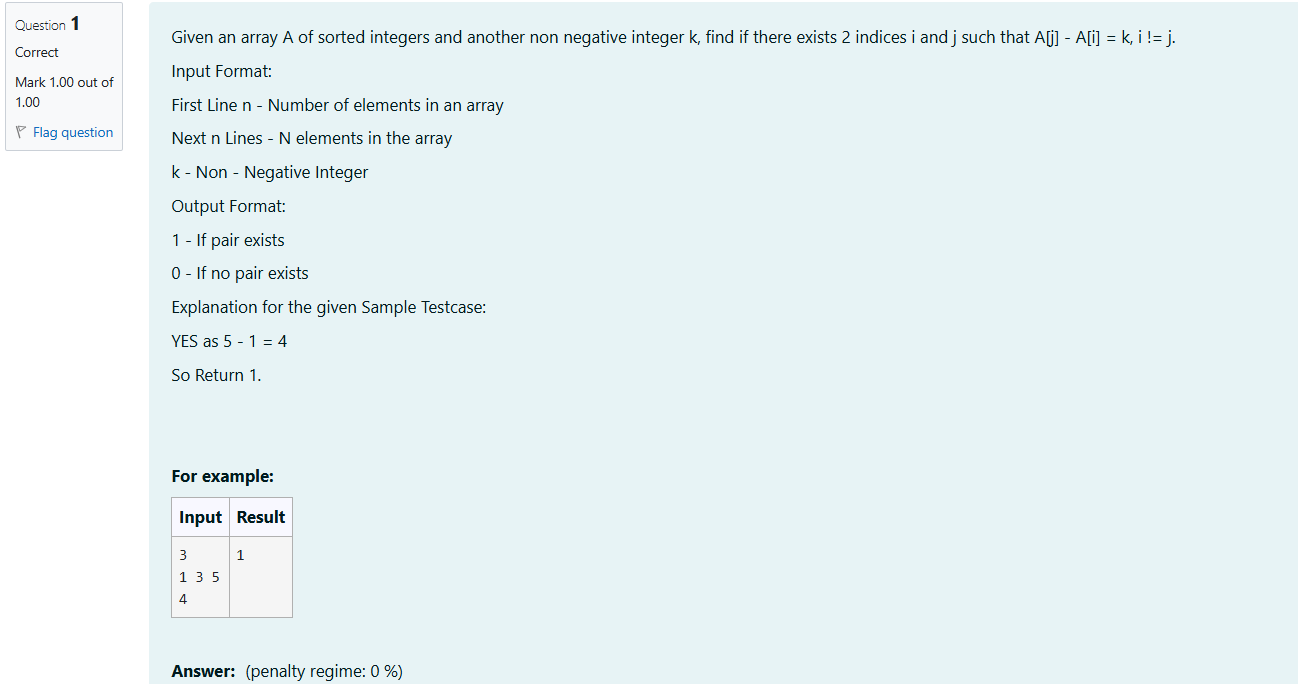
printf("0\n");

return 0;

}



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



#include <stdio.h>

int main() {

int n, k;

scanf("%d", &n);

int a[n];

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

scanf("%d", &a[i]);

}

scanf("%d", &k);

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

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

if (a[j] - a[i] == k) {

printf("1\n");

return 0;}

}

}

printf("0\n");

return 0;

}

