Question **1**

Correct

Mark 1.00 out of 1.00

Flag question

Given two numbers, write a C program to swap the given numbers.

For example:

Input	Result
10 20	20 10

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int a,b;
    scanf("%d",&a);
    scanf("%d",&b);
    printf("%d %d",b,a);
}
```

	Input	Expected	Got	
✔	10 20	20 10	20 10	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

```c
#include<stdio.h>
int main(){
    int m,p,c,t;
    scanf("%d %d %d",&m,&p,&c);
    t=m+p+c;
    if((m>=65 && p>=55 && c==50)||t>=180)
    {
        printf("The candidate is eligible");
    }
    else
    {
        printf("The candidate is not eligible");
    }
}
```

	Input	Expected	Got	
✔	70 60 80	The candidate is eligible	The candidate is eligible	✔
✔	50 80 80	The candidate is eligible	The candidate is eligible	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

```c
#include<stdio.h>
int main(){
    int a;
    scanf("%d",&a);
    if(a>2000)
    {
        int t;
        t=a-(a/10);
        printf("%d",t);
    }
    else
    {
        printf("%d",a);
    }
}
```

	Input	Expected	Got	
✔	1900	1900	1900	✔
✔	3000	2700	2700	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

```c
#include<stdio.h>
int main()
{
    int a,b,cal,sum;
    scanf("%d %d",&a,&b);
    cal=200*b;
    sum=(a*b)+cal;
    printf("%d",sum);
}
```

	Input	Expected	Got	
✔	500 3	2100	2100	✔
✔	100 3	900	900	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

```c
#include<stdio.h>
int main(){
    int m,n,x;
    scanf("%d %d %d",&m,&n,&x);
    for(int i=n;i>=m;i--)
    {
        if(i%x==0)
        {
            printf("%d ",i);
        }
    }
}
```

	Input	Expected	Got	
✔	2 40 7	35 28 21 14 7	35 28 21 14 7	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find the quotient and reminder of given integers.

For example:

Input	Result
12 3	4 0

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int a,b;
    scanf("%d %d",&a,&b);
    printf("%d\n",a/b);
    printf("%d",a%b);
}
```

	Input	Expected	Got	
✔	12 3	4 0	4 0	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find the biggest among the given 3 integers?

For example:

Input	Result
10 20 30	30

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int a,b,c;
    scanf("%d %d %d",&a,&b,&c);
    if(a>b && a>c)
    {
        printf("%d",a);
    }
    else if (b>a && b>c)
    {
        printf("%d",b);
    }
    else
    {
        printf("%d",c);
    }
}
```

	Input	Expected	Got	
✔	10 20 30	30	30	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find whether the given integer is odd or even?

For example:

Input	Result
12	Even
11	Odd

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int a;
    scanf("%d",&a);
    if(a%2==0)
    {
        printf("Even");
    }
    else{
        printf("Odd");
    }
}
```

	Input	Expected	Got	
✔	12	Even	Even	✔
✔	11	Odd	Odd	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find the factorial of given n.

For example:

Input	Result
5	120

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int a;
    unsigned long long pro=1;
    scanf("%d",&a);
    for(int i=1;i<=a;i++)
    {
        pro*=i;
    }
    printf("%llu\n",pro);
}
```

	Input	Expected	Got	
✔	5	120	120	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find the sum first N natural numbers.

For example:

Input	Result
3	6

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int a,n;
    scanf("%d",&a);
    n=a+1;
    n=a*n;
    n=n/2;
    printf("%d",n);
}
```

	Input	Expected	Got	
✔	3	6	6	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find the Nth term in the fibonacci series.

For example:

Input	Result
0	0
1	1
4	3

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int a=0,b=1,n;
    scanf("%d",&n);
    if(n==0)
    {
        b=a;
    }
    else
    {
        for(int i=2;i<=n;i++)
        {
            int c=a+b;
            a=b;
            b=c;
        }
    }
    printf("%d",b);
}
```

	Input	Expected	Got	
✔	0	0	0	✔
✔	1	1	1	✔
✔	4	3	3	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find the power of integers.

input:

a b

output:

a^b value

For example:

Input	Result
2 5	32

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
#include<math.h>
int main(){
    int a,b,s;
    scanf("%d %d",&a,&b);
    s=(int)pow(a,b);
    printf("%d",s);
}
```

	Input	Expected	Got	
✔	2 5	32	32	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find Whether the given integer is prime or not.

For example:

Input	Result
7	Prime
9	No Prime

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int n,c=0,i;
    scanf("%d",&n);
    for(i=2;i<n;i++)
    {
        if(n%i==0)
        {
            c+=1;
        }
    }
    if (c==0)
    {
        printf("Prime");
    }
    else
    {
        printf("No Prime");
    }
}
```

	Input	Expected	Got	
✔	7	Prime	Prime	✔
✔	9	No Prime	No Prime	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Started on	Monday, 19 August 2024, 8:22 AM
State	Finished
Completed on	Monday, 19 August 2024, 8:35 AM
Time taken	13 mins 4 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

```
Convert the following algorithm into a program and find its time complexity using the counter method.
void function (int n)
{
    int i= 1;
```

```
    int s =1;
```

```
    while(s <= n)
    {
         i++;
         s += i;
     }
}
```

Note: No need of counter increment for declarations and scanf() and count variable printf() statements.

Input:
A positive Integer n
Output:
Print the value of the counter variable

For example:

Input	Result
9	12

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    int n;
    int c=0;
    scanf("%d",&n);
    int i=1;
    c++;
    int s=1;
    c++;
    while(s<=n){
         c++;
         i++;
         c++;
         s+=i;
         c++;
    }
    c++;
    printf("%d",c);
}
```

	Input	Expected	Got	
✔	9	12	12	✔
✔	4	9	9	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ BASIC C PROGRAMMING-PRACTICE

Jump to...

Problem 2: Finding Complexity using Counter method ►

Started on	Thursday, 8 August 2024, 9:07 AM
State	Finished
Completed on	Thursday, 8 August 2024, 9:36 AM
Time taken	29 mins 19 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Convert the following algorithm into a program and find its time complexity using the counter method.

```
void func(int n)
{
    if(n==1)
    {
      printf("*");
    }
    else
    {
     for(int i=1; i<=n; i++)
     {
       for(int j=1; j<=n; j++)
       {
          printf("*");
          printf("*");
          break;
       }
     }
   }
 }
```

Note: No need of counter increment for declarations and scanf() and count variable printf() statements.

Input:

A positive Integer n

Output:

Print the value of the counter variable

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main()
{
    int n;
    int count=0;
    scanf("%d",&n);
    if(n==1)
    {
     count++;
     count++;
    }
    else{
        count++;
        for(int i=1;i<=n;i++)
        {
            count++;
            for(int j=1;j<=n;j++)
            {
                count++;
                count++;
                count++;
                break;
            }
            count++;
        }
        count++;
    }
    printf("%d",count);
}
```

	Input	Expected	Got	
✔	2	12	12	✔
✔	1000	5002	5002	✔
✔	143	717	717	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Problem 1: Finding Complexity using Counter Method

Jump to...

Problem 3: Finding Complexity using Counter Method ►

Started on	Thursday, 8 August 2024, 9:25 PM
State	Finished
Completed on	Thursday, 8 August 2024, 9:54 PM
Time taken	28 mins 38 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Convert the following algorithm into a program and find its time complexity using counter method.

```
Factor(num) {
{
   for (i = 1; i <= num;++i)
   {
    if (num % i== 0)
       {
         printf("%d ", i);
       }
    }
 }
```

Note: No need of counter increment for declarations and scanf() and counter variable printf() statement.

Input:
A positive Integer n
Output:
Print the value of the counter variable

Answer:

```c
#include<stdio.h>
int main()
{
    int a;
    int c=0;
    scanf("%d",&a);
    for(int i=1;i<=a;++i)
    {
     c++;
        if(a%i==0)
        {
            c++;
        }
        c++;
    }
    c++;
   printf("%d",c);
}
```

	Input	Expected	Got	
✔	12	31	31	✔
✔	25	54	54	✔
✔	4	12	12	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Problem 2: Finding Complexity using Counter method

Jump to...

Problem 4: Finding Complexity using Counter Method ►

Started on	Thursday, 8 August 2024, 9:56 PM
State	Finished
Completed on	Thursday, 8 August 2024, 10:09 PM
Time taken	13 mins 10 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Convert the following algorithm into a program and find its time

complexity using counter method.

```
void function(int n)
{
    int c= 0;
    for(int i=n/2; i<n; i++)
        for(int j=1; j<n; j = 2 * j)
            for(int k=1; k<n; k = k * 2)
                c++;
}
```

Note: No need of counter increment for declarations and scanf() and count variable printf() statements.

Input:
A positive Integer n
Output:
Print the value of the counter variable

Answer:

```
#include<stdio.h>
int main()
{
    int n;
    int c=0;
    scanf("%d",&n);
    for(int i=n/2;i<n;i++)
    {
        c++;
        for(int j=1;j<n;j=2*j)
        {
            c++;
            for(int k=1;k<n;k=k*2)
            {
                c++;
                c++;
            }
            c++;
        }
        c++;
    }
    c++;
    c++;
    printf("%d",c);
}
```

	Input	Expected	Got	
✔	4	30	30	✔
✔	10	212	212	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Problem 3: Finding Complexity using Counter Method

Jump to...

Problem 5: Finding Complexity using counter method ►

Started on	Thursday, 8 August 2024, 10:09 PM
State	Finished
Completed on	Thursday, 8 August 2024, 10:14 PM
Time taken	4 mins 23 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Convert the following algorithm into a program and find its time complexity using counter method.

```
void reverse(int n)
{
   int rev = 0, remainder;
   while (n != 0)
    {
        remainder = n % 10;
        rev = rev * 10 + remainder;
        n/= 10;

    }
print(rev);
}
```

Note: No need of counter increment for declarations and scanf() and count variable printf() statements.

Input:
A positive Integer n
Output:
Print the value of the counter variable

Answer:

```c
#include<stdio.h>
int main()
{
    int n;
    int c=0;
    scanf("%d",&n);
    int rev=0,remainder;
    c++;
    while(n!=0)
    {
        c++;
        remainder=n%10;
        c++;
        rev=rev*10+remainder;
        c++;
        n/=10;
        c++;
    }
    c++;
    c++;
    printf("%d",c);

}
```

	Input	Expected	Got	
✔	12	11	11	✔
✔	1234	19	19	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Problem 4: Finding Complexity using Counter Method

Jump to...

1-Number of Zeros in a Given Array ►

Started on	Thursday, 19 September 2024, 8:26 AM
State	Finished
Completed on	Thursday, 19 September 2024, 8:35 AM
Time taken	8 mins 27 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Problem Statement

Given an array of 1s and 0s this has all 1s first followed by all 0s. Aim is to find the number of 0s. Write a program using Divide and Conquer to Count the number of zeroes in the given array.

Input Format

First Line Contains Integer m – Size of array

Next m lines Contains m numbers – Elements of an array

Output Format

First Line Contains Integer – Number of zeroes present in the given array.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int a,b[20];
    int c=0;
    scanf("%d",&a);
    for(int i=0;i<a;i++){
        scanf("%d",&b[i]);
    }
    for(int i=0;i<a;i++){
        if(b[i]==0){
           c++;
        }
    }
    printf("%d",c);
}
```

	Input	Expected	Got	
✔	5 1 1 1 0 0	2	2	✔
✔	10 1 1 1 1 1 1 1 1 1 1	0	0	✔

	Input	Expected	Got	
✔	8 0 0 0 0 0 0 0 0	8	8	✔
✔	17 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0	2	2	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Problem 5: Finding Complexity using counter method

Jump to...

2-Majority Element ►

Started on	Thursday, 19 September 2024, 8:35 AM
State	Finished
Completed on	Thursday, 3 October 2024, 8:29 AM
Time taken	13 days 23 hours
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Given an array `nums` of size `n`, return *the majority element*.

The majority element is the element that appears more than `⌊n / 2⌋` times. You may assume that the majority element always exists in the array.

Example 1:

```
Input: nums = [3,2,3]
Output: 3
```

Example 2:

```
Input: nums = [2,2,1,1,1,2,2]
Output: 2
```

Constraints:

- `n == nums.length`
- `1 <= n <= 5 * 10^4`
- `-2^31 <= nums[i] <= 2^31 - 1`

For example:

Input	Result
3 3 2 3	3
7 2 2 1 1 1 2 2	2

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
int majorityElement(int* nums, int numsSize) {
    int candidate = 0;
    int count = 0;
    for (int i = 0; i < numsSize; i++) {
        if (count == 0) {
            candidate = nums[i];
            count = 1;
        } else if (nums[i] == candidate) {
            count++;
        } else {
            count--;
        }
    }
    return candidate;
}
int main() {
    int n;
    scanf("%d", &n);
    int nums[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &nums[i]);
    }
    int majority = majorityElement(nums, n);
    printf("%d\n", majority);
    return 0;
}
```

```
28
29
```

	Input	Expected	Got	
✔	3 3 2 3	3	3	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 1-Number of Zeros in a Given Array

Jump to...

3-Finding Floor Value ►

Started on	Thursday, 3 October 2024, 8:31 AM
State	Finished
Completed on	Thursday, 3 October 2024, 8:34 AM
Time taken	3 mins 42 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Problem Statement:

Given a sorted array and a value x, the floor of x is the largest element in array smaller than or equal to x. Write divide and conquer algorithm to find floor of x.

Input Format

First Line Contains Integer n – Size of array

Next n lines Contains n numbers – Elements of an array

Last Line Contains Integer x – Value for x

Output Format

First Line Contains Integer – Floor value for x

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
int findFloor(int* arr, int n, int x) {
    int left = 0;
    int right = n - 1;
    int floorValue = -1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (arr[mid] == x) {
            return arr[mid];
        }
        else if (arr[mid] < x) {
            floorValue = arr[mid];
            left = mid + 1;
        }
        else {
            right = mid - 1;
        }
    }
    return floorValue;
}
int main() {
    int n;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    int x;
    scanf("%d", &x);
    int floorValue = findFloor(arr, n, x);
    printf("%d\n",floorValue);

    return 0;
}

```

	Input	Expected	Got	
✔	6 1 2 8 10 12 19 5	2	2	✔
✔	5 10 22 85 108 129 100	85	85	✔
✔	7 3 5 7 9 11 13 15 10	9	9	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 2-Majority Element

Jump to...

4-Two Elements sum to x ►

Started on	Thursday, 3 October 2024, 8:34 AM
State	Finished
Completed on	Thursday, 3 October 2024, 8:37 AM
Time taken	2 mins 5 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Problem Statement:

Given a sorted array of integers say arr[] and a number x. Write a recursive program using divide and conquer strategy to check if there exist two elements in the array whose sum = x. If there exist such two elements then return the numbers, otherwise print as "No".

Note: Write a Divide and Conquer Solution

Input Format

First Line Contains Integer n – Size of array

Next n lines Contains n numbers – Elements of an array

Last Line Contains Integer x – Sum Value

Output Format

First Line Contains Integer – Element1

Second Line Contains Integer – Element2 (Element 1 and Elements 2 together sums to value "x")

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
void findPairWithSum(int* arr, int left, int right, int x) {
    if (left >= right) {
        printf("No\n");
        return;
    }
    int currentSum = arr[left] + arr[right];
    if (currentSum == x) {
        printf("%d\n", arr[left]);
        printf("%d\n", arr[right]);
        return;
    } else if (currentSum < x) {
        findPairWithSum(arr, left + 1, right, x);
    } else {
        findPairWithSum(arr, left, right - 1, x);
    }
}
int main() {
    int n;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    int x;
    scanf("%d", &x);
    findPairWithSum(arr, 0, n - 1, x);
    return 0;
}
```

	Input	Expected	Got	
✔	4 2 4 8 10 14	4 10	4 10	✔

	Input	Expected	Got	
✔	5 2 4 6 8 10 100	No	No	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 3-Finding Floor Value

Jump to...

6-Implementation of Quick Sort ►

Started on	Thursday, 3 October 2024, 8:37 AM
State	Finished
Completed on	Thursday, 3 October 2024, 8:44 AM
Time taken	6 mins 55 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Write a Program to Implement the Quick Sort Algorithm

Input Format:
The first line contains the no of elements in the list-n
The next n lines contain the elements.

Output:
Sorted list of elements

For example:

Input	Result
5 67 34 12 98 78	12 34 67 78 98

Answer:

```c
#include <stdio.h>
void swap(int* a, int* b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}
int partition(int arr[], int low, int high) {
    int pivot = arr[high];
    int i = (low - 1);
    for (int j = low; j < high; j++) {
        if (arr[j] <= pivot) {
            i++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[i + 1], &arr[high]);
    return (i + 1);
}
void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}
int main() {
    int n;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    quickSort(arr, 0, n - 1);
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}

```

	Input	Expected	Got	
✔	5 67 34 12 98 78	12 34 67 78 98	12 34 67 78 98	✔
✔	10 1 56 78 90 32 56 11 10 90 114	1 10 11 32 56 56 78 90 90 114	1 10 11 32 56 56 78 90 90 114	✔
✔	12 9 8 7 6 5 4 3 2 1 10 11 90	1 2 3 4 5 6 7 8 9 10 11 90	1 2 3 4 5 6 7 8 9 10 11 90	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 4-Two Elements sum to x

Jump to...

1-G-Coin Problem ►

Started on	Thursday, 5 September 2024, 8:40 AM
State	Finished
Completed on	Thursday, 5 September 2024, 9:05 AM
Time taken	24 mins 59 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Write a program to take value V and we want to make change for V Rs, and we have infinite supply of each of the denominations in Indian currency, i.e., we have infinite supply of { 1, 2, 5, 10, 20, 50, 100, 500, 1000} valued coins/notes, what is the minimum number of coins and/or notes needed to make the change.

Input Format:

Take an integer from stdin.

Output Format:

print the integer which is change of the number.

Example Input :

64

Output:

4

Explanaton:

We need a 50 Rs note and a 10 Rs note and two 2 rupee coins.

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main()
{
    int a;
    scanf("%d",&a);
    int d[] = {1000, 500, 100, 50, 20, 10, 5, 2, 1};
    int n = sizeof(d) / sizeof(d[0]);
    int c = 0;
    for (int i = 0; i < n; i++){
        c += a / d[i];
        a %= d[i];
    }
    printf("%d",c);

}
```

	Input	Expected	Got	
✔	49	5	5	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 6-Implementation of Quick Sort

Jump to...

2-G-Cookies Problem ►

Started on	Thursday, 12 September 2024, 8:58 AM
State	Finished
Completed on	Thursday, 12 September 2024, 9:08 AM
Time taken	10 mins 26 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie.

Each child i has a greed factor g[i], which is the minimum size of a cookie that the child will be content with; and each cookie j has a size s[j]. If s[j] >= g[i], we can assign the cookie j to the child i, and the child i will be content. Your goal is to maximize the number of your content children and output the maximum number.

Example 1:

Input:

3

1 2 3

2

1 1

Output:

1

Explanation: You have 3 children and 2 cookies. The greed factors of 3 children are 1, 2, 3.

And even though you have 2 cookies, since their size is both 1, you could only make the child whose greed factor is 1 content.

You need to output 1.

Constraints:

1 <= g.length <= 3 * 10^4

0 <= s.length <= 3 * 10^4

1 <= g[i], s[j] <= 2^31 - 1

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <stdlib.h>
int compare(const void *a, const void *b) {
    return (*(int*)a - *(int*)b);
}
int main() {
    int numChildren, numCookies;
    scanf("%d", &numChildren);
    int *greedFactors = (int*)malloc(numChildren * sizeof(int));
    for (int i = 0; i < numChildren; i++) {
        scanf("%d", &greedFactors[i]);
    }
    scanf("%d", &numCookies);
    int *cookieSizes = (int*)malloc(numCookies * sizeof(int));
    for (int j = 0; j < numCookies; j++) {
        scanf("%d", &cookieSizes[j]);
    }
    qsort(greedFactors, numChildren, sizeof(int), compare);
    qsort(cookieSizes, numCookies, sizeof(int), compare);
    int childIndex = 0;
    int cookieIndex = 0;
    int contentChildren = 0;
    while (childIndex < numChildren && cookieIndex < numCookies) {
        if (cookieSizes[cookieIndex] >= greedFactors[childIndex]) {
            contentChildren++;
            childIndex++;
        }
        cookieIndex++;
    }
    printf("%d\n", contentChildren);
    free(greedFactors);
```

```
    free(cookieSizes);
    return 0;
}
```

	Input	Expected	Got	
✔	2 1 2 3 1 2 3	2	2	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 1-G-Coin Problem

Jump to...

3-G-Burger Problem ►

Started on	Thursday, 10 October 2024, 8:14 AM
State	Finished
Completed on	Thursday, 10 October 2024, 8:20 AM
Time taken	5 mins 59 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

A person needs to eat burgers. Each burger contains a count of calorie. After eating the burger, the person needs to run a distance to burn out his calories.

If he has eaten i burgers with c calories each, then he has to run at least $3^i * c$ kilometers to burn out the calories. For example, if he ate 3

burgers with the count of calorie in the order: [1, 3, 2], the kilometers he needs to run are $(3^0 * 1) + (3^1 * 3) + (3^2 * 2) = 1 + 9 + 18 = 28$.

But this is not the minimum, so need to try out other orders of consumption and choose the minimum value. Determine the minimum distance

he needs to run. Note: He can eat burger in any order and use an efficient sorting algorithm.Apply greedy approach to solve the problem.

Input Format

First Line contains the number of burgers

Second line contains calories of each burger which is n space-separate integers

Output Format

Print: Minimum number of kilometers needed to run to burn out the calories

Sample Input

```
3
5 10 7
```

Sample Output

```
76
```

For example:

Test	Input	Result
Test Case 1	3 1 3 2	18

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int compareDescending(const void *a, const void *b) { return (*(int *)b - *(int *)a);
}
int main()
{
    int numBurgers;
    scanf("%d", &numBurgers);
    int *calories = (int *)malloc(numBurgers * sizeof(int));
    for (int i = 0; i < numBurgers; i++)
    {
        scanf("%d", &calories[i]);

    }
    qsort(calories, numBurgers, sizeof(int), compareDescending);
    int totalDistance = 0;
    for (int i = 0; i < numBurgers; i++)
    {
        int power = (int)pow( numBurgers, i);
        totalDistance += power * calories[i];
        }
        printf("%d\n", totalDistance);
        free(calories);
        return 0;

```

```
27 }
```

	Test	Input	Expected	Got	
✔	Test Case 1	3 1 3 2	18	18	✔
✔	Test Case 2	4 7 4 9 6	389	389	✔
✔	Test Case 3	3 5 10 7	76	76	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 2-G-Cookies Problem

Jump to...

4-G-Array Sum max problem ►

Started on	Thursday, 12 September 2024, 9:24 AM
State	Finished
Completed on	Thursday, 12 September 2024, 9:26 AM
Time taken	2 mins 55 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Given an array of N integer, we have to maximize the sum of arr[i] * i, where i is the index of the element (i = 0, 1, 2, ..., N).Write an algorithm based on Greedy technique with a Complexity O(nlogn).

Input Format:

First line specifies the number of elements-n

The next n lines contain the array elements.

Output Format:

Maximum Array Sum to be printed.

Sample Input:

5

2 5 3 4 0

Sample output:

40

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <stdlib.h>
int compare(const void *a, const void *b) {
    return (*(int*)a - *(int*)b);
}
int main() {
    int n;
    scanf("%d", &n);
    int *arr = (int*)malloc(n * sizeof(int));
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    qsort(arr, n, sizeof(int), compare);
    int max_sum = 0;
    for (int i = 0; i < n; i++) {
        max_sum += arr[i] * i;
    }
    printf("%d\n", max_sum);
    free(arr);

    return 0;
}

```

	Input	Expected	Got	
✔	5 2 5 3 4 0	40	40	✔

	Input	Expected	Got	
✔	10 2 2 2 4 4 3 3 5 5 5	191	191	✔
✔	2 45 3	45	45	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 3-G-Burger Problem

Jump to...

5-G-Product of Array elements-Minimum ►

Started on	Thursday, 12 September 2024, 9:12 AM
State	Finished
Completed on	Thursday, 12 September 2024, 9:13 AM
Time taken	1 min 25 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Given two arrays array_One[] and array_Two[] of same size N. We need to first rearrange the arrays such that the sum of the product of pairs(1 element from each) is minimum. That is SUM (A[i] * B[i]) for all i is minimum.

For example:

Input	Result
3 1 2 3 4 5 6	28

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <stdlib.h>
int compareAscending(const void *a, const void *b) {
    return (*(int *)a - *(int *)b);
}
int compareDescending(const void *a, const void *b) {
    return (*(int *)b - *(int *)a);
}

int main() {
    int n;
    scanf("%d", &n);
    int *array_One = (int *)malloc(n * sizeof(int));
    int *array_Two = (int *)malloc(n * sizeof(int));
    for (int i = 0; i < n; i++) {
        scanf("%d", &array_One[i]);
    }
    for (int i = 0; i < n; i++) {
        scanf("%d", &array_Two[i]);
    }
    qsort(array_One, n, sizeof(int), compareAscending);
    qsort(array_Two, n, sizeof(int), compareDescending);
    long long minSum = 0;
    for (int i = 0; i < n; i++) {
        minSum += (long long)array_One[i] * array_Two[i];
    }
    printf("%lld\n", minSum);
    free(array_One);
    free(array_Two);
    return 0;
}

```

	Input	Expected	Got	
✔	3 1 2 3 4 5 6	28	28	✔

	Input	Expected	Got	
✔	4 7 5 1 2 1 3 4 1	22	22	✔
✔	5 20 10 30 10 40 8 9 4 3 10	590	590	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 4-G-Array Sum max problem

Jump to...

1-DP-Playing with Numbers ►

Started on	Thursday, 10 October 2024, 8:28 AM
State	Finished
Completed on	Thursday, 10 October 2024, 8:29 AM
Time taken	17 secs
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 10.00 out of 10.00

Playing with Numbers:

Ram and Sita are playing with numbers by giving puzzles to each other. Now it was Ram term, so he gave Sita a positive integer 'n' and two numbers 1 and 3. He asked her to find the possible ways by which the number n can be represented using 1 and 3.Write any efficient algorithm to find the possible ways.

Example 1:

Input: *6*
Output:*6*
Explanation: *There are 6 ways to 6 represent number with 1 and 3*

1+1+1+1+1+1
3+3
1+1+1+3
1+1+3+1
1+3+1+1
3+1+1+1

Input Format

First Line contains the number n

Output Format

Print: The number of possible ways 'n' can be represented using 1 and 3

Sample Input

6

Sample Output

6

Answer: (penalty regime: 0 %)

```
#include <stdio.h>
unsigned long long countWays(int n) {
    unsigned long long dp[n + 1];
    dp[0] = 1;
    for (int i = 1; i <= n; i++) {
        dp[i] = 0;
        dp[i] += dp[i - 1];
        if (i >= 3) {
            dp[i] += dp[i - 3];
        }
    }
    return dp[n];
}
int main() {
    int n;
    scanf("%d", &n);
    printf("%llu\n", countWays(n));
    return 0;
}
```

	Input	Expected	Got	
✔	6	6	6	✔
✔	25	8641	8641	✔
✔	100	24382819596721629	24382819596721629	✔

Passed all tests! ✔

Correct

Marks for this submission: 10.00/10.00.

◄ 5-G-Product of Array elements-Minimum

Jump to...

2-DP-Playing with chessboard ►

Started on	Thursday, 10 October 2024, 8:29 AM
State	Finished
Completed on	Thursday, 10 October 2024, 8:39 AM
Time taken	9 mins 22 secs
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 10.00 out of 10.00

Playing with Chessboard:

Ram is given with an n*n chessboard with each cell with a monetary value. Ram stands at the (0,0), that the position of the top left white rook. He is been given a task to reach the bottom right black rook position (n-1, n-1) constrained that he needs to reach the position by traveling the maximum monetary path under the condition that he can only travel one step right or one step down the board. Help ram to achieve it by providing an efficient DP algorithm.

Example:

Input

3

1 2 4

2 3 4

8 7 1

Output:

19

Explanation:

Totally there will be 6 paths among that the optimal is

Optimal path value:1+2+8+7+1=19

Input Format

First Line contains the integer n

The next n lines contain the n*n chessboard values

Output Format

Print Maximum monetary value of the path

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#define MAX_SIZE 100
int maxMonetaryPath(int board[MAX_SIZE][MAX_SIZE], int n) {
    int dp[MAX_SIZE][MAX_SIZE];
    dp[0][0] = board[0][0];
    for (int j = 1; j < n; j++) {
        dp[0][j] = dp[0][j - 1] + board[0][j];
    }
    for (int i = 1; i < n; i++) {
        dp[i][0] = dp[i - 1][0] + board[i][0];
    }
    for (int i = 1; i < n; i++) {
        for (int j = 1; j < n; j++) {
            dp[i][j] = board[i][j] + (dp[i - 1][j] > dp[i][j - 1] ? dp[i - 1][j] : dp[i][j - 1]);
        }
    }
    return dp[n - 1][n - 1];
}

int main() {
    int n;
    int board[MAX_SIZE][MAX_SIZE];
    scanf("%d",&n);
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &board[i][j]);
        }
    }
    int maxPathValue = maxMonetaryPath(board, n);
```

```
30      printf("%d\n", maxPathValue);
31
32      return 0;
33  }
34
```

	Input	Expected	Got	
✔	3 1 2 4 2 3 4 8 7 1	19	19	✔
✔	3 1 3 1 1 5 1 4 2 1	12	12	✔
✔	4 1 1 3 4 1 5 7 8 2 3 4 6 1 6 9 0	28	28	✔

Passed all tests! ✔

Correct

Marks for this submission: 10.00/10.00.

◄ 1-DP-Playing with Numbers

Jump to...

3-DP-Longest Common Subsequence ►

Started on	Thursday, 10 October 2024, 8:39 AM
State	Finished
Completed on	Thursday, 10 October 2024, 8:51 AM
Time taken	11 mins 47 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Given two strings find the length of the common longest subsequence(need not be contiguous) between the two.

Example:

s1: ggtabe

s2: tgatasb

s1	a	g	g	t	a	b	
s2	g	x	t	x	a	y	b

The length is 4

Solveing it using Dynamic Programming

For example:

Input	Result
aab azb	2

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <string.h>
int longest_common_subsequence(char *s1, char *s2) {
    int m = strlen(s1);
    int n = strlen(s2);
        int dp[m + 1][n + 1];
    for (int i = 0; i <= m; i++) {
        for (int j = 0; j <= n; j++) {
            if (i == 0 || j == 0) {
                dp[i][j] = 0;
            } else if (s1[i - 1] == s2[j - 1]) {
                dp[i][j] = dp[i - 1][j - 1] + 1;
            } else {
                dp[i][j] = (dp[i - 1][j] > dp[i][j - 1]) ? dp[i - 1][j] : dp[i][j - 1];  // Take the maxim
            }
        }
    }
    return dp[m][n];
}

int main() {
    char s1[100], s2[100];
    fgets(s1, sizeof(s1), stdin);
    s1[strcspn(s1, "\n")] = 0;
    fgets(s2, sizeof(s2), stdin);
    s2[strcspn(s2, "\n")] = 0;
    int result = longest_common_subsequence(s1, s2);
    printf("%d\n", result);
    return 0;
}
```

	Input	Expected	Got	
✔	aab azb	2	2	✔
✔	ABCD ABCD	4	4	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 2-DP-Playing with chessboard

Jump to...

4-DP-Longest non-decreasing Subsequence ►

Started on	Thursday, 10 October 2024, 9:08 AM
State	Finished
Completed on	Thursday, 10 October 2024, 9:09 AM
Time taken	1 min 5 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Problem statement:

Find the length of the Longest Non-decreasing Subsequence in a given Sequence.

Eg:

Input:9

Sequence:[-1,3,4,5,2,2,2,2,3]

the subsequence is [-1,2,2,2,2,3]

Output:6

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
int longest_non_decreasing_subsequence(int arr[], int n) {
    int dp[n];
    int max_length = 1;
    for (int i = 0; i < n; i++) {
        dp[i] = 1;
    }
    for (int i = 1; i < n; i++) {
        for (int j = 0; j < i; j++) {
            if (arr[i] >= arr[j]) {
                dp[i] = (dp[i] > dp[j] + 1) ? dp[i] : (dp[j] + 1);
            }
        }
        if (dp[i] > max_length) {
            max_length = dp[i];
        }
    }
    return max_length;
}
int main() {
    int n;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    int result = longest_non_decreasing_subsequence(arr, n);
    printf("%d\n", result);
    return 0;
}
```

	Input	Expected	Got	
✔	9 -1 3 4 5 2 2 2 2 3	6	6	✔
✔	7 1 2 2 4 5 7 6	6	6	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 3-DP-Longest Common Subsequence

Jump to...

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

Started on	Saturday, 2 November 2024, 1:35 PM
State	Finished
Completed on	Saturday, 2 November 2024, 1:37 PM
Time taken	2 mins 53 secs
Marks	1.00/1.00
Grade	**4.00** out of 4.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

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 find_duplicate(int* nums, int n) {
    int slow = nums[0];
    int fast = nums[0];


    do {
        slow = nums[slow];
        fast = nums[nums[fast]];
    } while (slow != fast);


    slow = nums[0];
    while (slow != fast) {
        slow = nums[slow];
        fast = nums[fast];
    }

    return slow;
}

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


    int result = find_duplicate(nums, n);
    printf("%d\n", result);

    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.

◄ 4-DP-Longest non-decreasing Subsequence

Jump to...

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

Started on	Saturday, 2 November 2024, 2:15 PM
State	Finished
Completed on	Saturday, 2 November 2024, 2:17 PM
Time taken	1 min 24 secs
Marks	1.00/1.00
Grade	**4.00** out of 4.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

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 find_duplicate(int* nums, int n) {
    int slow = nums[0];
    int fast = nums[0];


    do {
        slow = nums[slow];
        fast = nums[nums[fast]];
    } while (slow != fast);


    slow = nums[0];
    while (slow != fast) {
        slow = nums[slow];
        fast = nums[fast];
    }

    return slow;
}

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


    int result = find_duplicate(nums, n);
    printf("%d\n", result);

    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.

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

Jump to...

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

Started on	Saturday, 2 November 2024, 1:51 PM
State	Finished
Completed on	Saturday, 2 November 2024, 2:01 PM
Time taken	9 mins 58 secs
Marks	1.00/1.00
Grade	**30.00** out of 30.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

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 %)

```
#include <stdio.h>

void find_intersection(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++;
        }
    }
    printf("\n");
}

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]);
        }


        find_intersection(arr1, n1, arr2, n2);
    }
```

	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.

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

Jump to...

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

Started on	Saturday, 2 November 2024, 2:09 PM
State	Finished
Completed on	Saturday, 2 November 2024, 2:12 PM
Time taken	2 mins 58 secs
Marks	1.00/1.00
Grade	**30.00** out of 30.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

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 %)

```
#include <stdio.h>

void find_intersection(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++;
        }
    }
    printf("\n");
}

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]);
        }
        find_intersection(arr1, n1, arr2, n2);
    }

    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.

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

Jump to...

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

Started on	Saturday, 2 November 2024, 2:22 PM
State	Finished
Completed on	Saturday, 2 November 2024, 2:29 PM
Time taken	6 mins 59 secs
Marks	1.00/1.00
Grade	**4.00** out of 4.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

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

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

        if (diff == k && i != j) {
            return 1;
        } else if (diff < k) {
            j++;
        } else {
            i++;
        }
    }
    return 0;
}

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

    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.

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

Jump to...

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

Started on	Saturday, 2 November 2024, 2:30 PM
State	Finished
Completed on	Saturday, 2 November 2024, 2:30 PM
Time taken	53 secs
Marks	1.00/1.00
Grade	**4.00** out of 4.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

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

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

        if (diff == k && i != j) {
            return 1;
        } else if (diff < k) {
            j++;
        } else {
            i++;
        }
    }
    return 0;
}

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

    return 0;
}
```

```
34
```

	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.

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

Jump to...