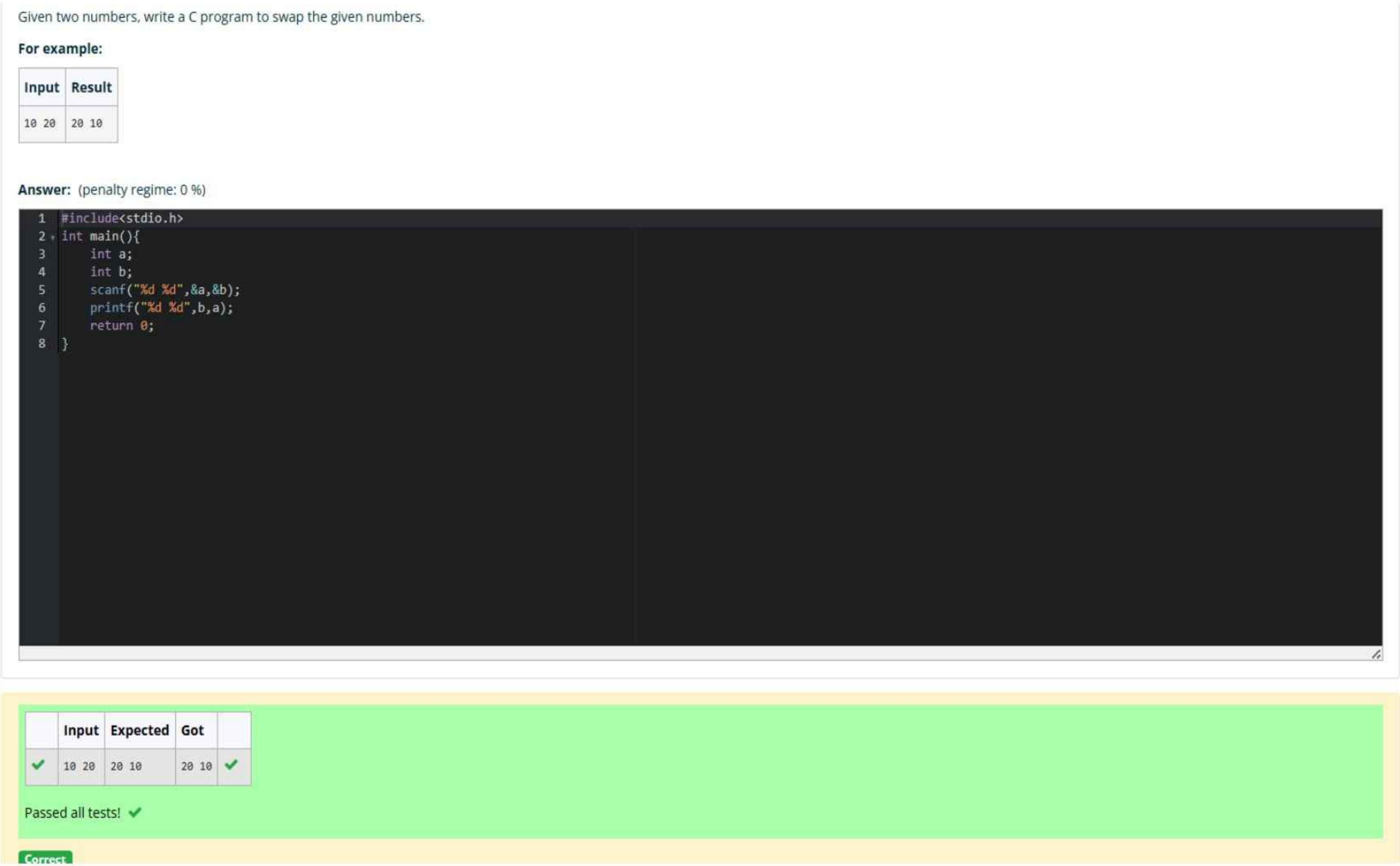


Given two numbers, write a C program to swap the given numbers.

For example:

Input	Result
10 20	20 10

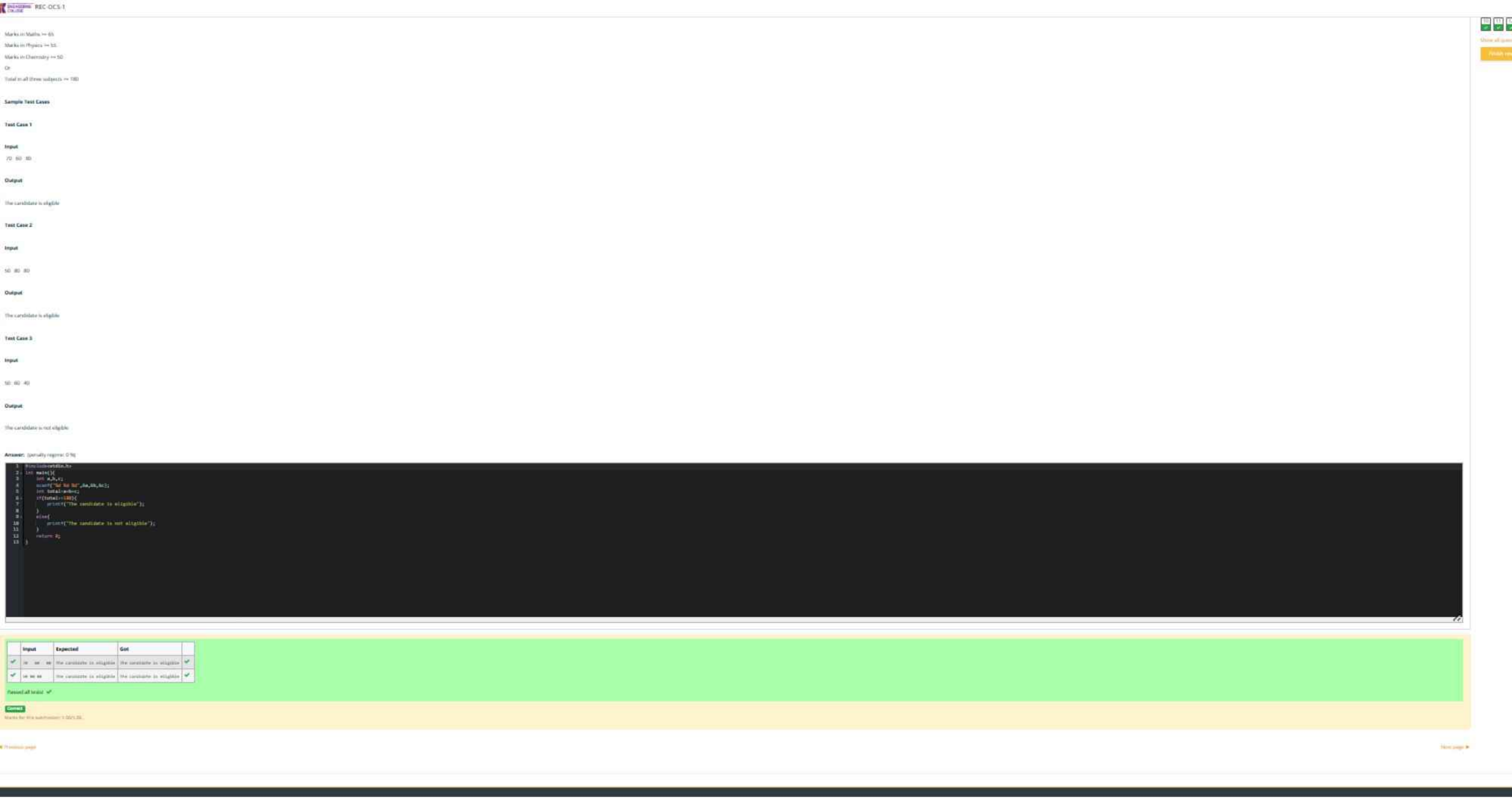
Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    int a;
    int b;
    scanf("%d %d",&a,&b);
    printf("%d %d",b,a);
    return 0;
}
```

	Input	Expected	Got	
✓	10 20	20 10	20 10	✓

Passed all tests! ✓

Correct



Marks in Maths >= 65

Marks in Physics >= 55

Marks in Chemistry >= 50

Or

Total in all three subjects >= 180

Sample Test Cases

Test Case 1

Input

70 60 80

Output

The candidate is eligible

Test Case 2

Input

50 80 80

Output

The candidate is eligible

Test Case 3

Input

50 60 40

Output

The candidate is not eligible

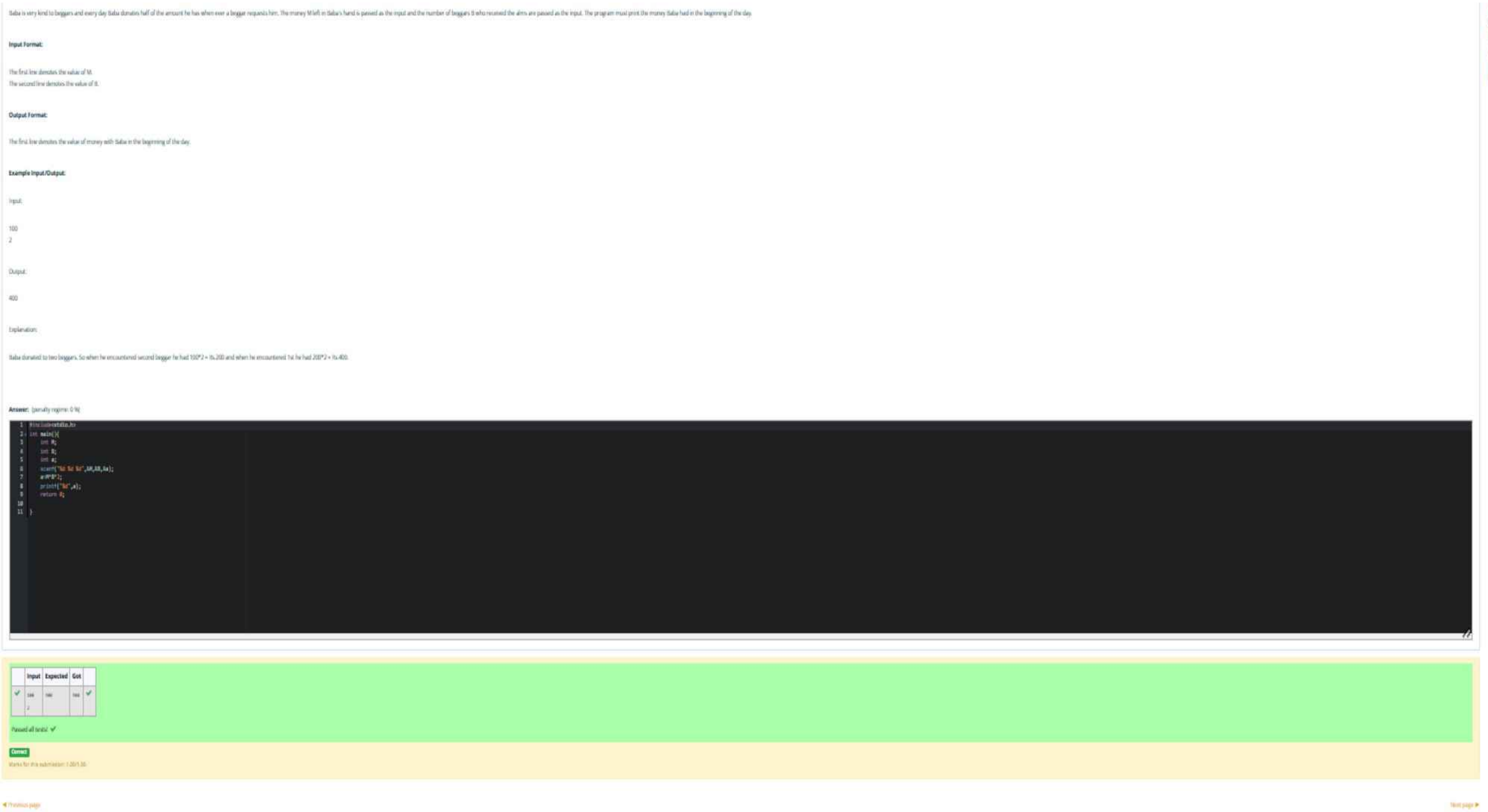
Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    int a,b,c;
    scanf("%d %d %d",&a,&b,&c);
    int total=a+b+c;
    if(total>=180){
        printf("The candidate is eligible");
    }
    else{
        printf("The candidate is not eligible");
    }
    return 0;
}
```

	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! ✓

Marks for this submission: 1.00/1.00.



Baba is very kind to beggars and every day Baba donates half of the amount he has when ever a beggar requests him. The money M left in Baba's hand is passed as the input and the number of beggars B who received the alms are passed as the input. The program must print the money Baba had in the beginning of the day.

Input Format:

The first line denotes the value of M.
The second line denotes the value of B.

Output Format:

The first line denotes the value of money with Baba in the beginning of the day.

Example Input/Output:

Input:

100
2

Output:

400

Explanation:

Baba donated to two beggars. So when he encountered second beggar he had 100*2 = Rs.200 and when he encountered 1st he had 200*2 = Rs.400.

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    int M;
    int B;
    int a;
    scanf("%d %d %d",&M,&B,&a);
    a=M*B*2;
    printf("%d",a);
    return 0;

}
```

	Input	Expected	Got	
✓	100 2	400	400	✓

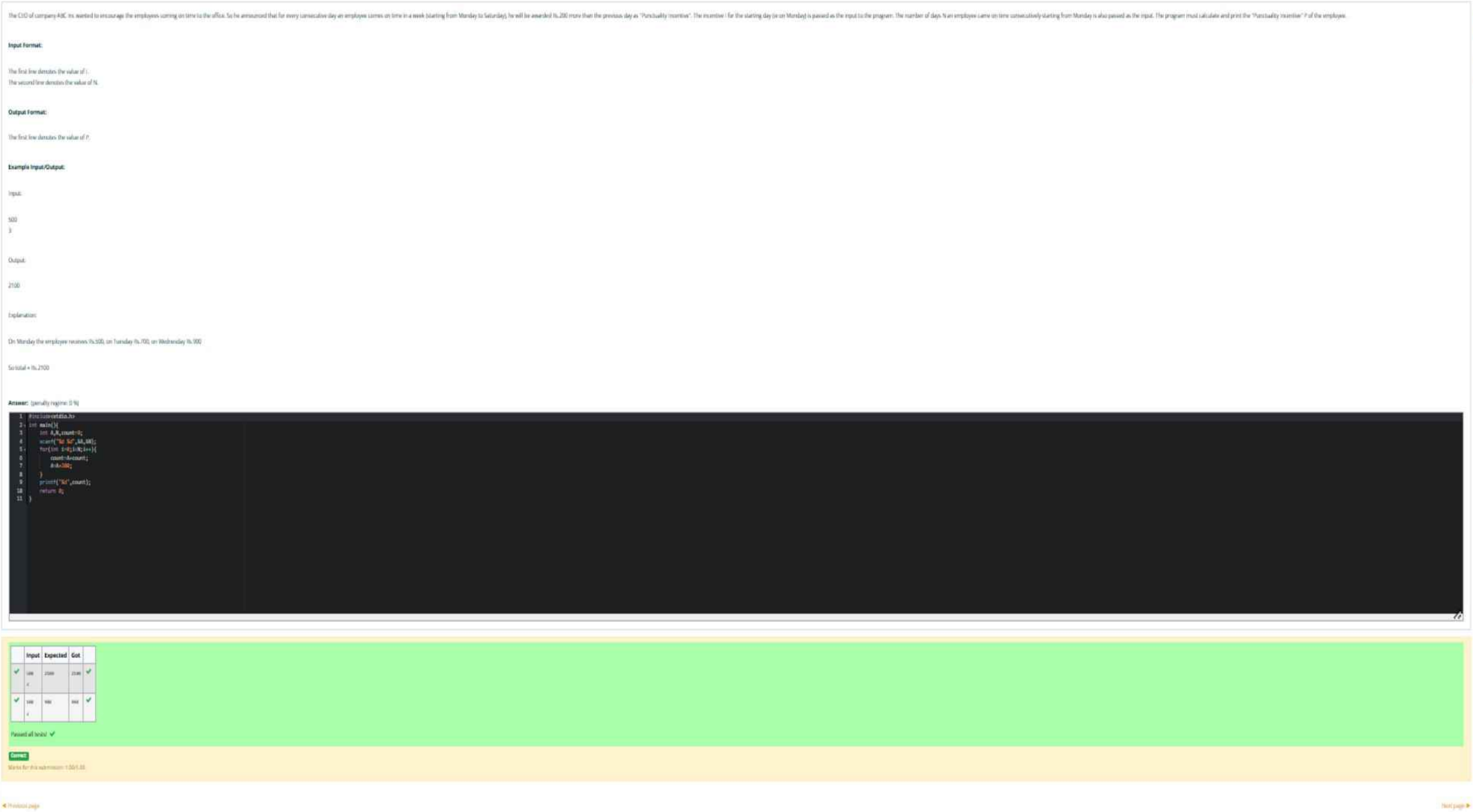
Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

◀ Previous page

Next page ▶



The CEO of company ABC Inc wanted to encourage the employees coming on time to the office. So he announced that for every consecutive day an employee comes on time in a week (starting from Monday to Saturday), he will be awarded Rs.200 more than the previous day as "Punctuality incentive". The incentive I for the starting day (ie on Monday) is passed as the input to the program. The number of days N an employee came on time consecutively starting from Monday is also passed as the input. The program must calculate and print the "Punctuality Incentive" P of the employee.

Input Format:

The first line denotes the value of I.
The second line denotes the value of N.

Output Format:

The first line denotes the value of P.

Example Input/Output:

Input:

500
3

Output:

2100

Explanation:

On Monday the employee receives Rs.500, on Tuesday Rs.700, on Wednesday Rs.900

So total = Rs.2100

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int A,N,count=0;
    scanf("%d %d",&A,&N);
    for(int i=0;i<N;i++){
        count=A+count;
        A=A+200;
    }
    printf("%d",count);
    return 0;
}
```

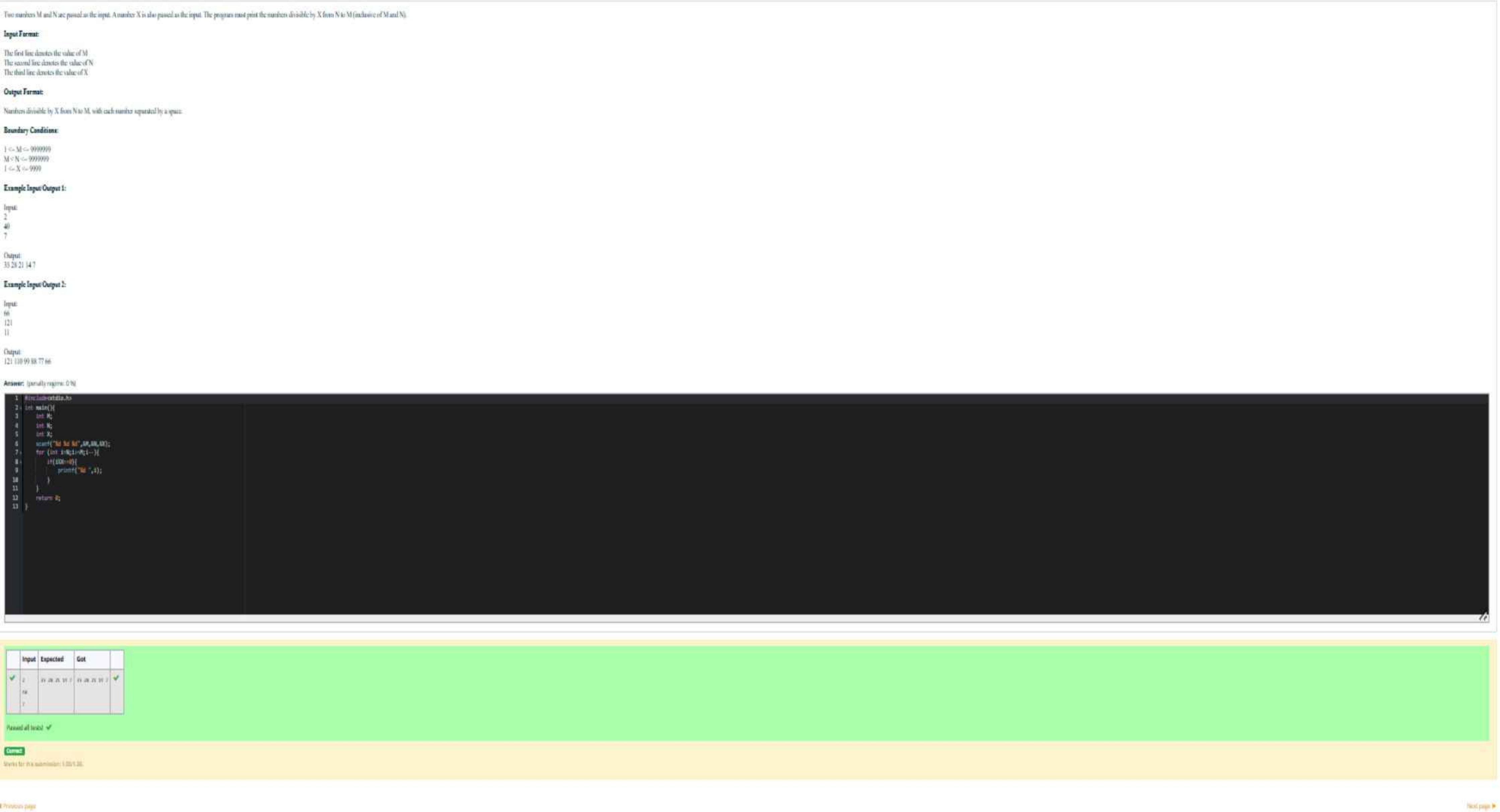
	Input	Expected	Got	
✓	500 3	2100	2100	✓
✓	100 3	900	900	✓

Passed all tests! ✓

Marks for this submission: 1.00/1.00.

◄ Previous page

Next page ►



Two numbers M and N are passed as the input. A number X is also passed as the input. The program must print the numbers divisible by X from N to M (inclusive of M and N).

Input Format:

The first line denotes the value of M
The second line denotes the value of N
The third line denotes the value of X

Output Format:

Numbers divisible by X from N to M, with each number separated by a space.

Boundary Conditions:

1 <= M <= 9999999
M < N <= 9999999
1 <= X <= 9999

Example Input/Output 1:

Input:
2
40
7

Output:
35 28 21 14 7

Example Input/Output 2:

Input:
66
121
11

Output:
121 110 99 88 77 66

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    int M;
    int N;
    int X;
    scanf("%d %d %d",&M,&N,&X);
    for (int i=N;i>=M;i--){
        if(i%X==0){
            printf("%d ",i);
        }
    }
    return 0;
}
```

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

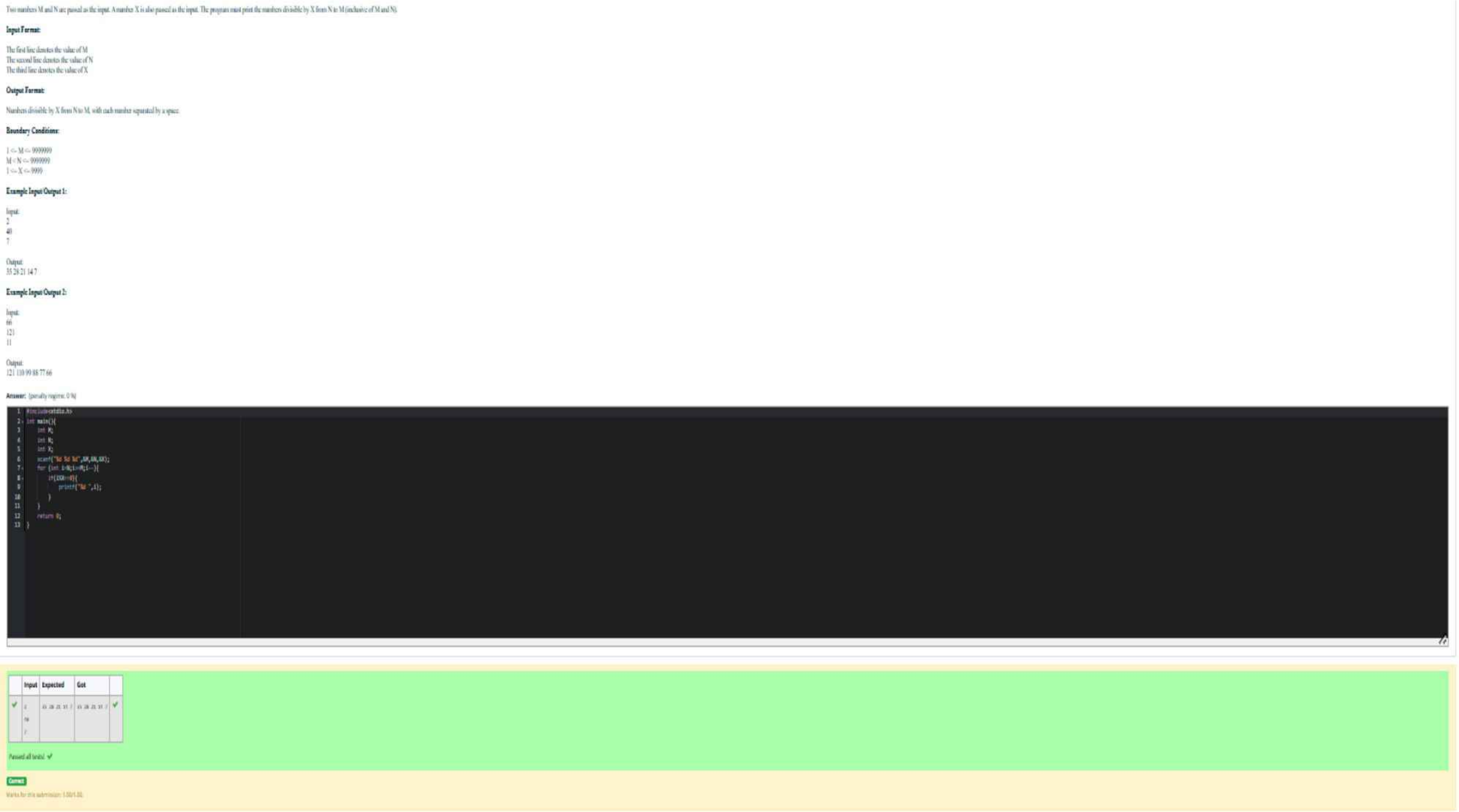
Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

◄ Previous page

Next page ►



Two numbers M and N are passed as the input. A number X is also passed as the input. The program must print the numbers divisible by X from N to M (inclusive of M and N).

Input Format:

The first line denotes the value of M
The second line denotes the value of N
The third line denotes the value of X

Output Format:

Numbers divisible by X from N to M, with each number separated by a space.

Boundary Conditions:

1 <= M <= 9999999
M < N <= 9999999
1 <= X <= 9999

Example Input/Output 1:

Input:
2
40
7

Output:
35 28 21 14 7

Example Input/Output 2:

Input:
66
121
11

Output:
121 110 99 88 77 66

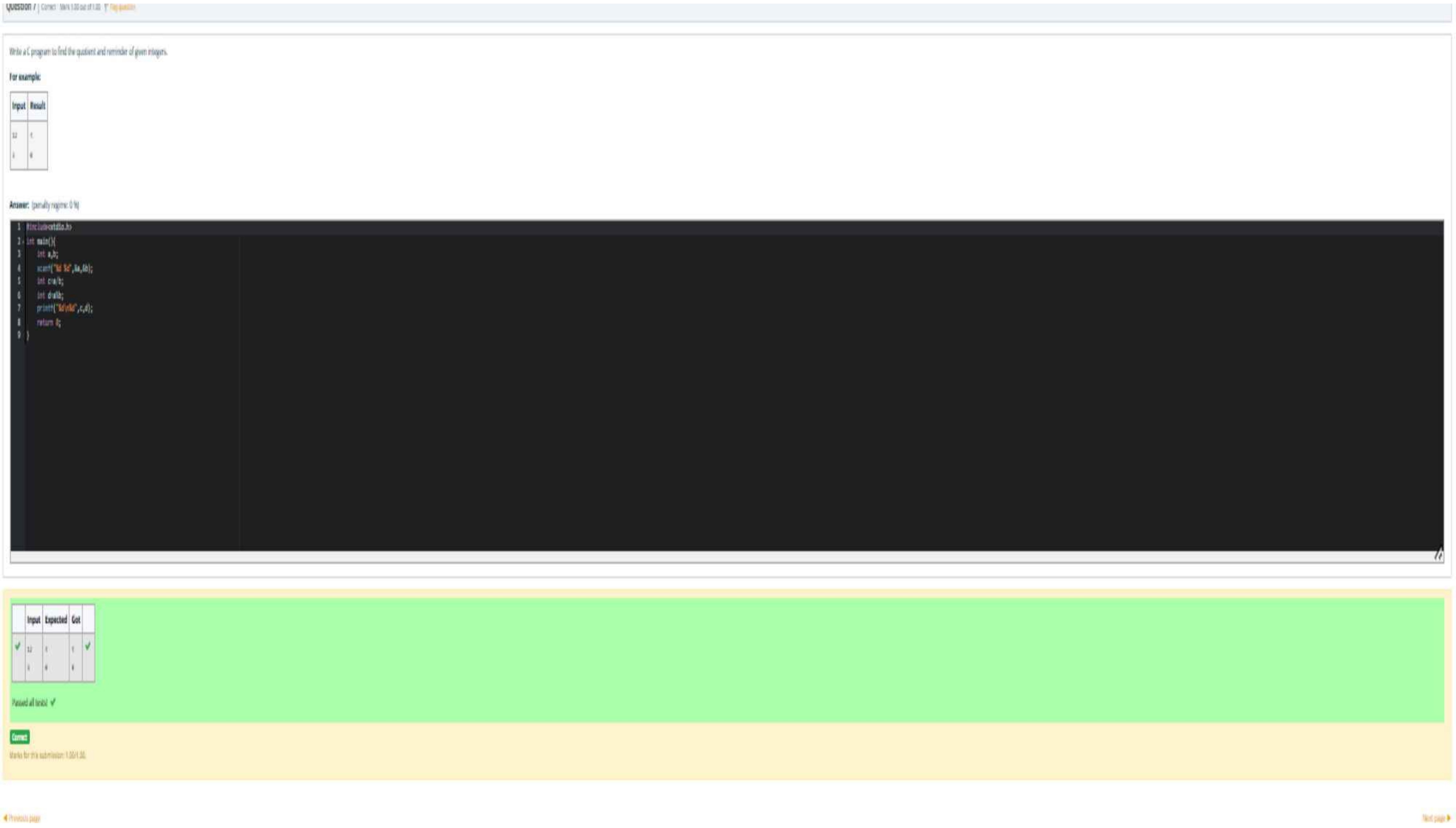
Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    int M;
    int N;
    int X;
    scanf("%d %d %d",&M,&N,&X);
    for (int i=N;i>=M;i--){
        if(i%X==0){
            printf("%d ",i);
        }
    }
    return 0;
}
```

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

Passed all tests! ✓

Marks for this submission: 1.00/1.00.



Question 7 | Correct Mark 1.00 out of 1.00 Flag question

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 %)

```
#include<stdio.h>
int main(){
    int a,b;
    scanf("%d %d",&a,&b);
    int c=a/b;
    int d=a%b;
    printf("%d\n%d",c,d);
    return 0;
}
```

	Input	Expected	Got	
✓	12 3	4 0	4 0	✓

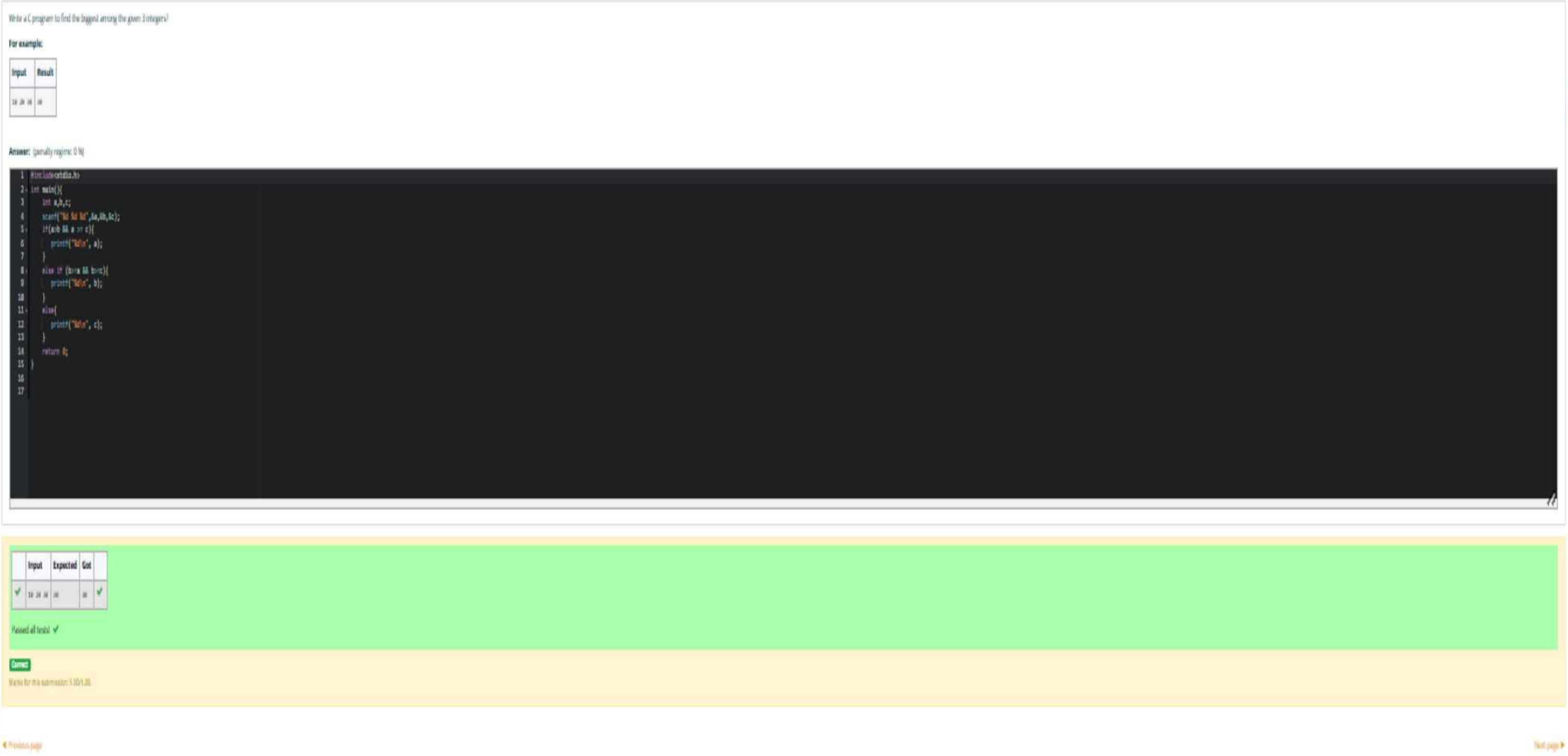
Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

◀ Previous page

Next page ▶



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 %)

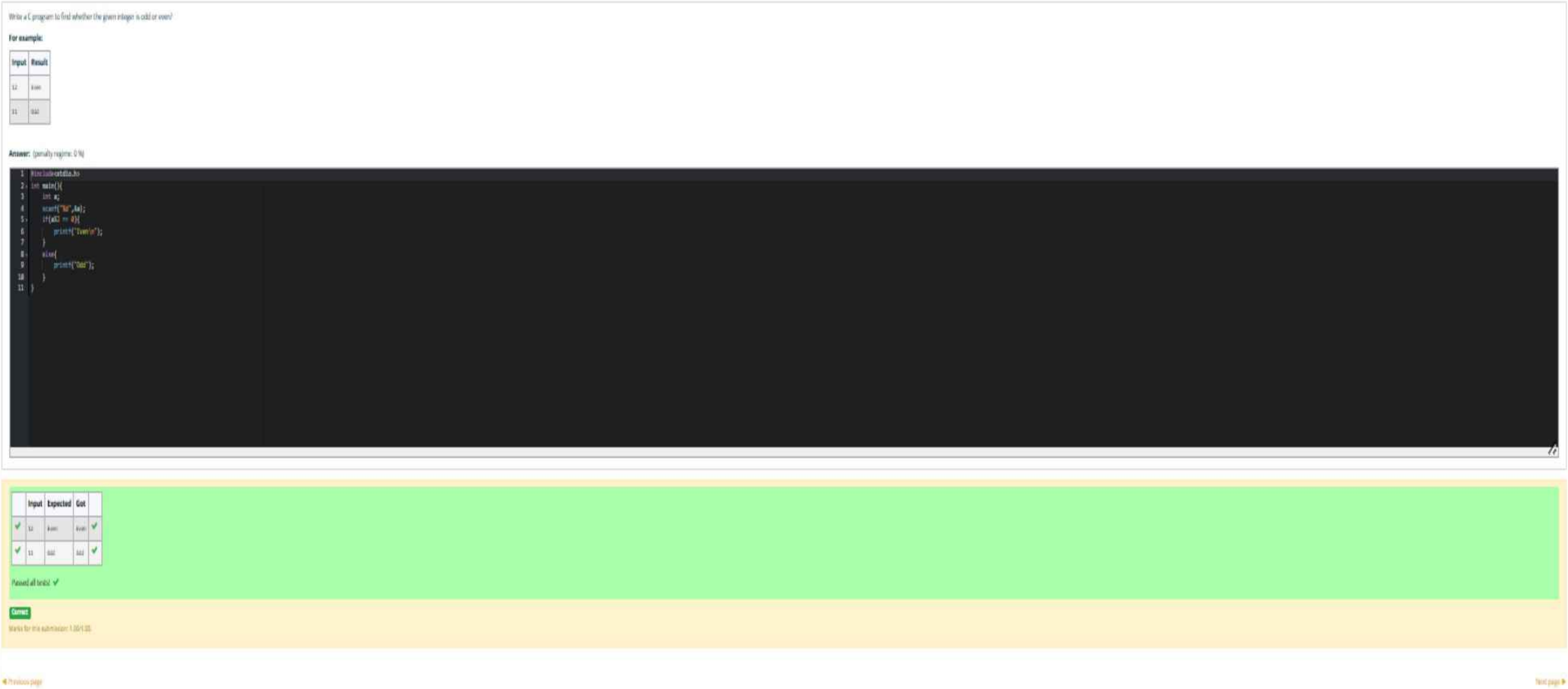
```c
#include<stdio.h>
int main(){
    int a,b,c;
    scanf("%d %d %d",&a,&b,&c);
    if(a>b && a >= c){
        printf("%d\n", a);
    }
    else if (b>=a && b>=c){
        printf("%d\n", b);
    }
    else{
        printf("%d\n", c);
    }
    return 0;
}
```

	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\n");
    }
    else{
        printf("Odd");
    }
}
```

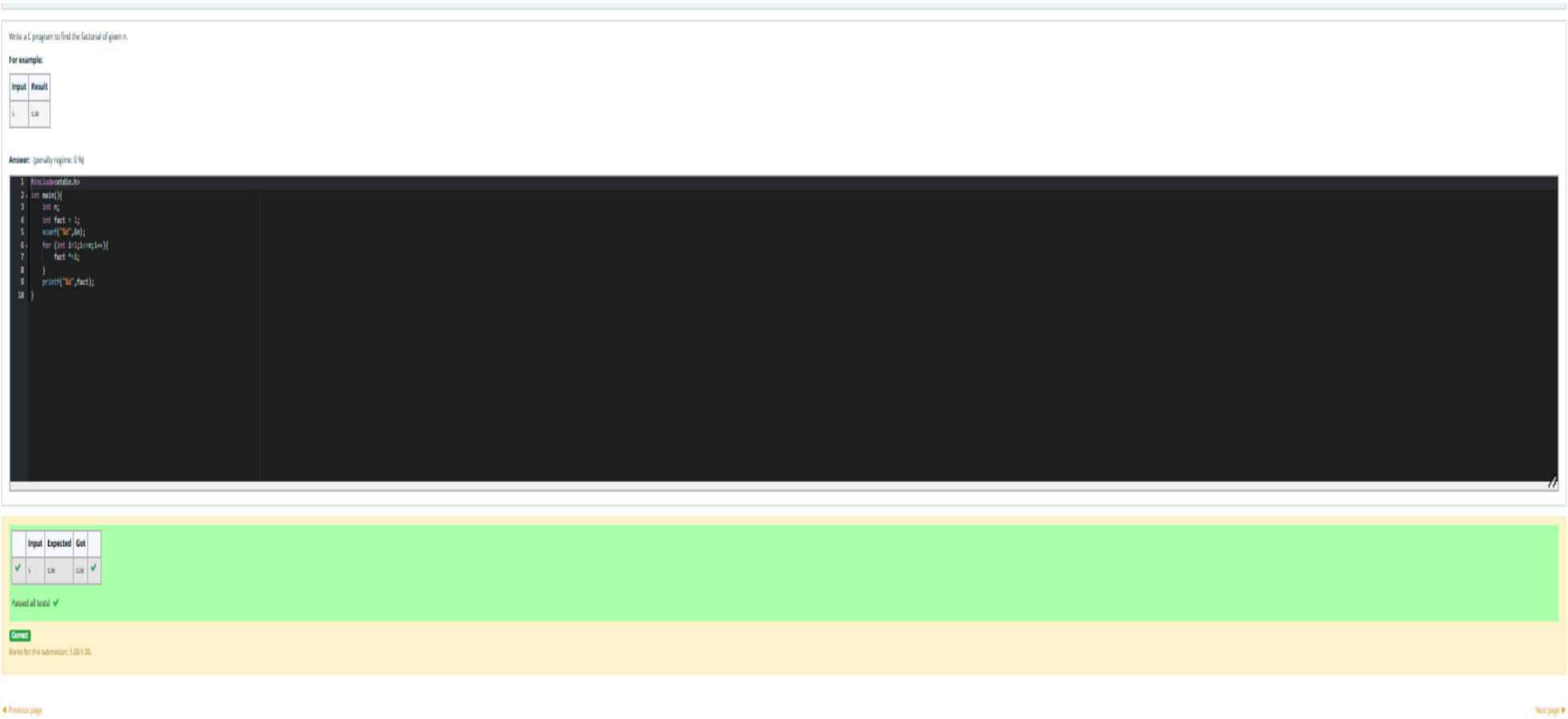
	Input	Expected	Got	
✓	12	Even	Even	✓
✓	11	Odd	Odd	✓

Passed all tests! ✓

Marks for this submission: 1.00/1.00.

◀ Previous page

Next page ▶



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

For example:

Input	Result
5	120

Answer: (penalty regime: 0 %)

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

	Input	Expected	Got	
✓	5	120	120	✓

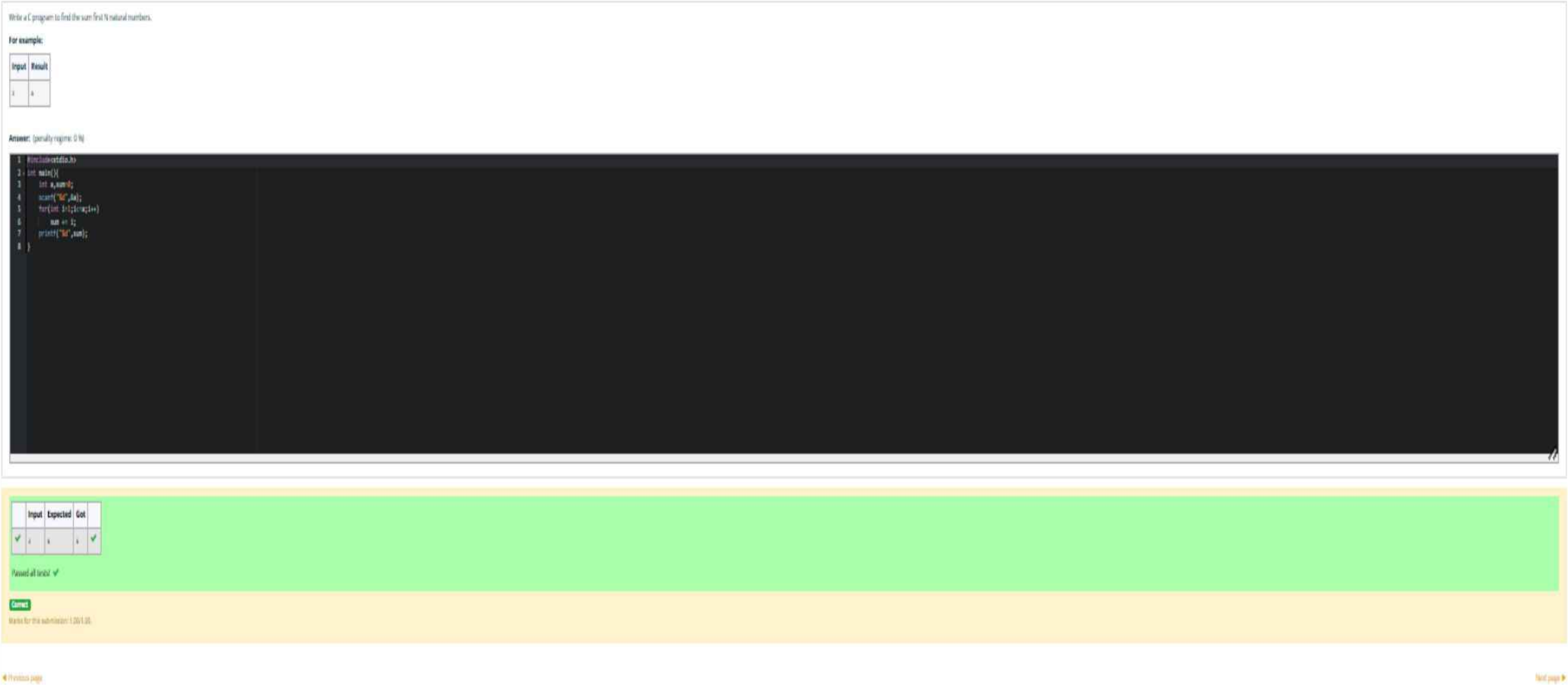
Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

◀ Previous page

Next page ▶



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 n,sum=0;
    scanf("%d",&n);
    for(int i=1;i<=n;i++)
        sum += i;
    printf("%d",sum);
}
```

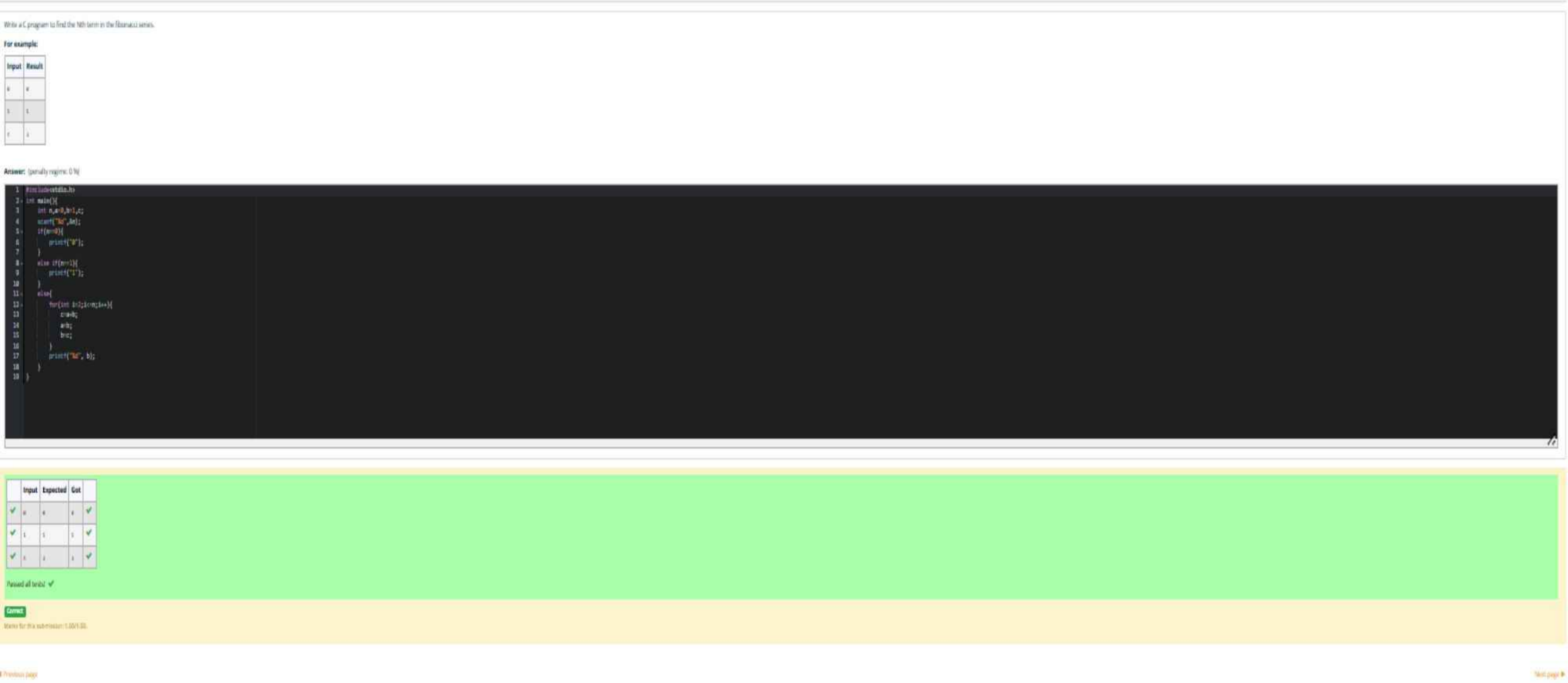
	Input	Expected	Got	
✓	3	6	6	✓

Passed all tests! ✓

Marks for this submission: 1.00/1.00.

◄ Previous page

Next page ►



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 %)

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

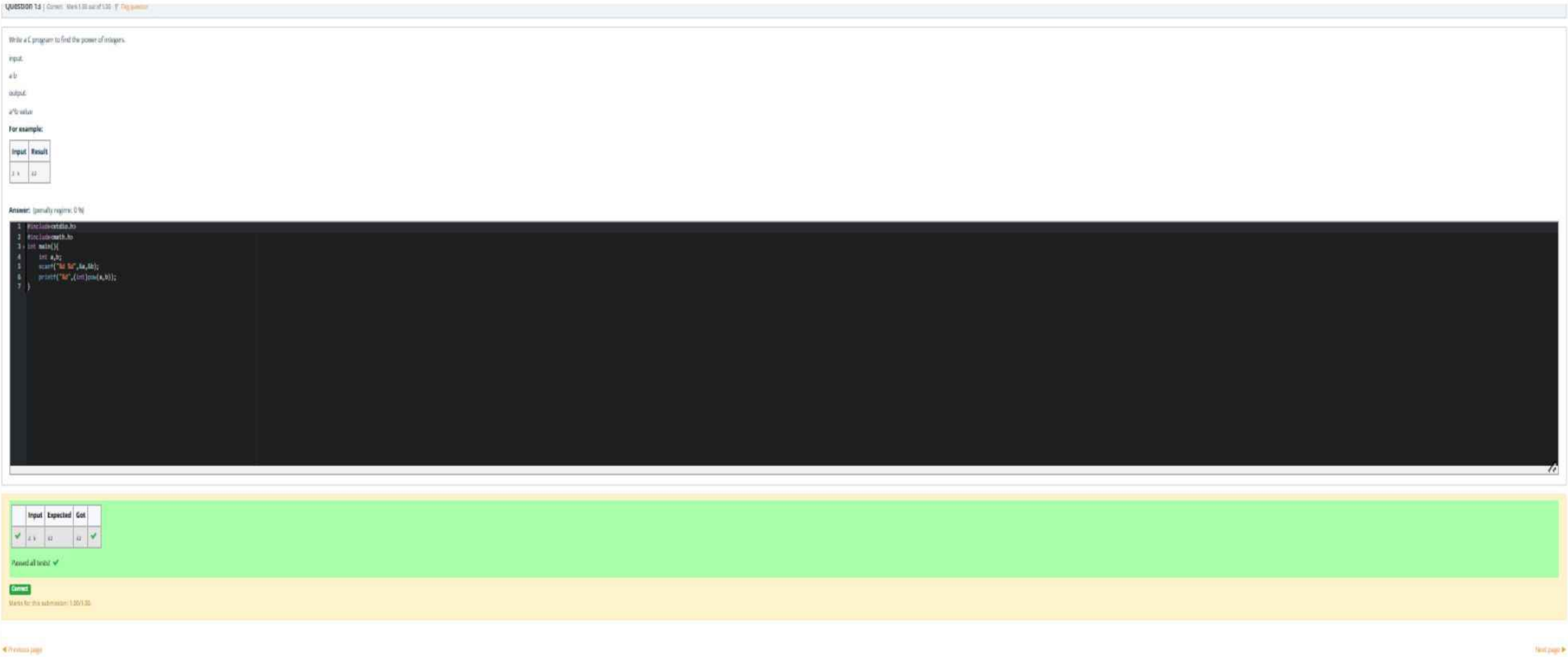
	Input	Expected	Got	
✓	0	0	0	✓
✓	1	1	1	✓
✓	4	3	3	✓

Passed all tests! ✓

Marks for this submission: 1.00/1.00.

◀ Previous page

Next page ▶



Question 13 | Correct Mark 1.00 out of 1.00 Flag question

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 %)

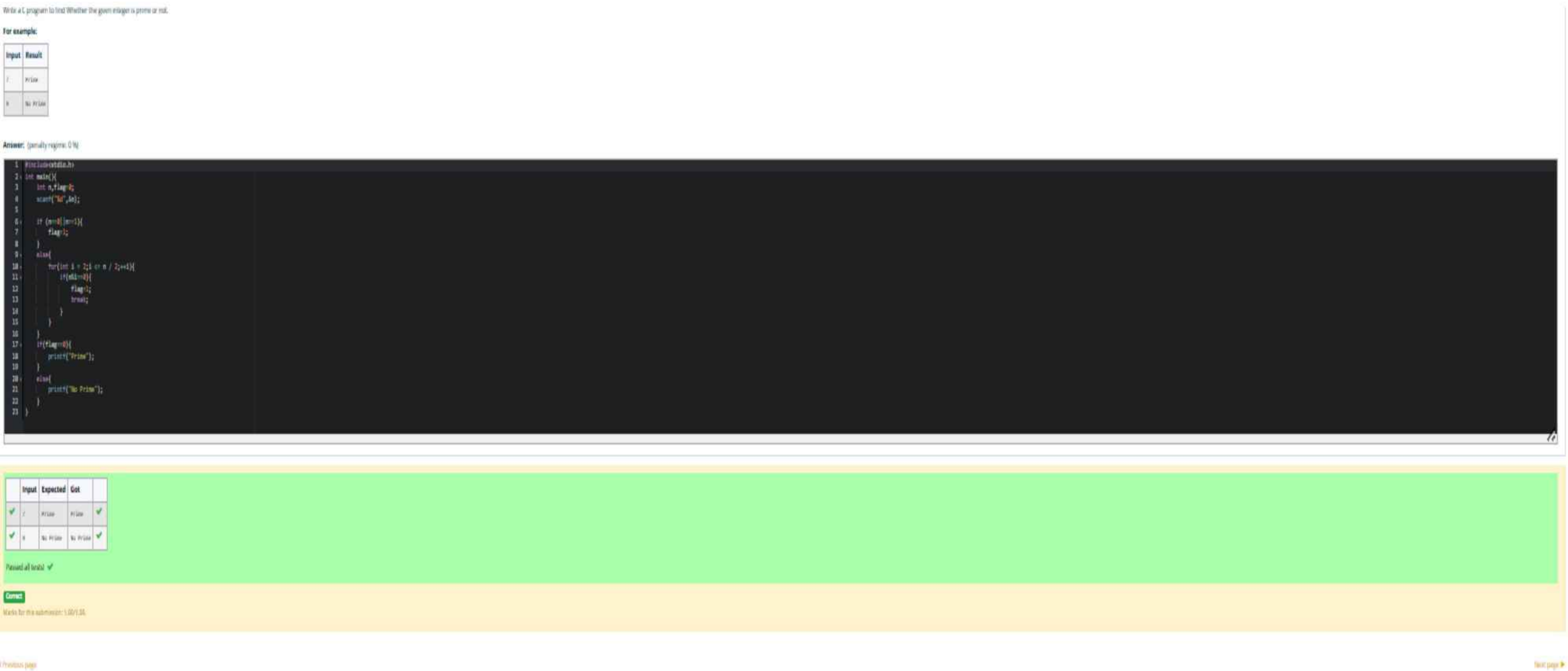
```
#include<stdio.h>
#include<math.h>
int main(){
    int a,b;
    scanf("%d %d",&a,&b);
    printf("%d",(int)pow(a,b));
}
```

	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
8	No Prime

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main(){
    int n,flag=0;
    scanf("%d",&n);

    if (n==0||n==1){
        flag=1;
    }
    else{
        for(int i = 2;i <= n / 2;++i){
            if(n%i==0){
                flag=1;
                break;
            }
        }
    }
    if(flag==0){
        printf("Prime");
    }
    else{
        printf("No Prime");
    }
}
```

	Input	Expected	Got	
✓	7	Prime	Prime	✓
✓	8	No Prime	No Prime	✓

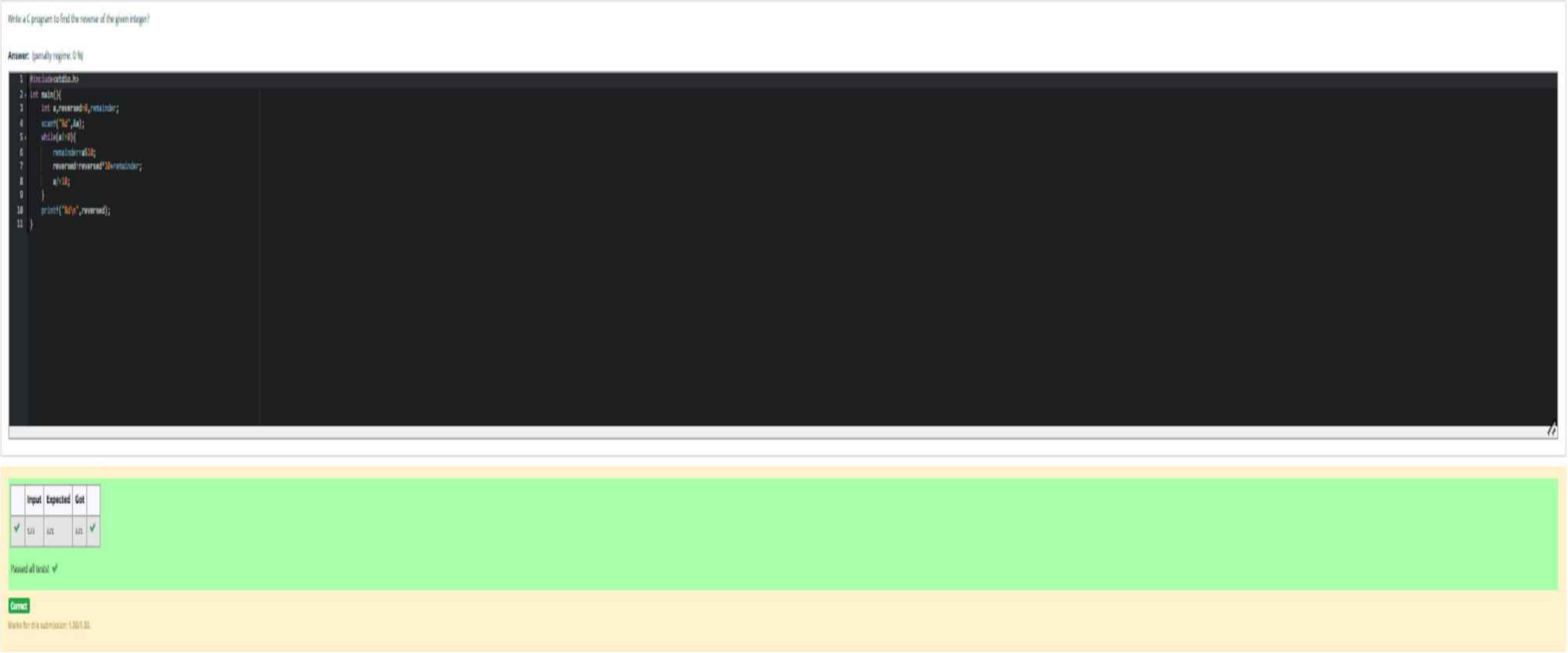
Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

◀ Previous page

Next page ▶



Write a C program to find the reverse of the given integer?

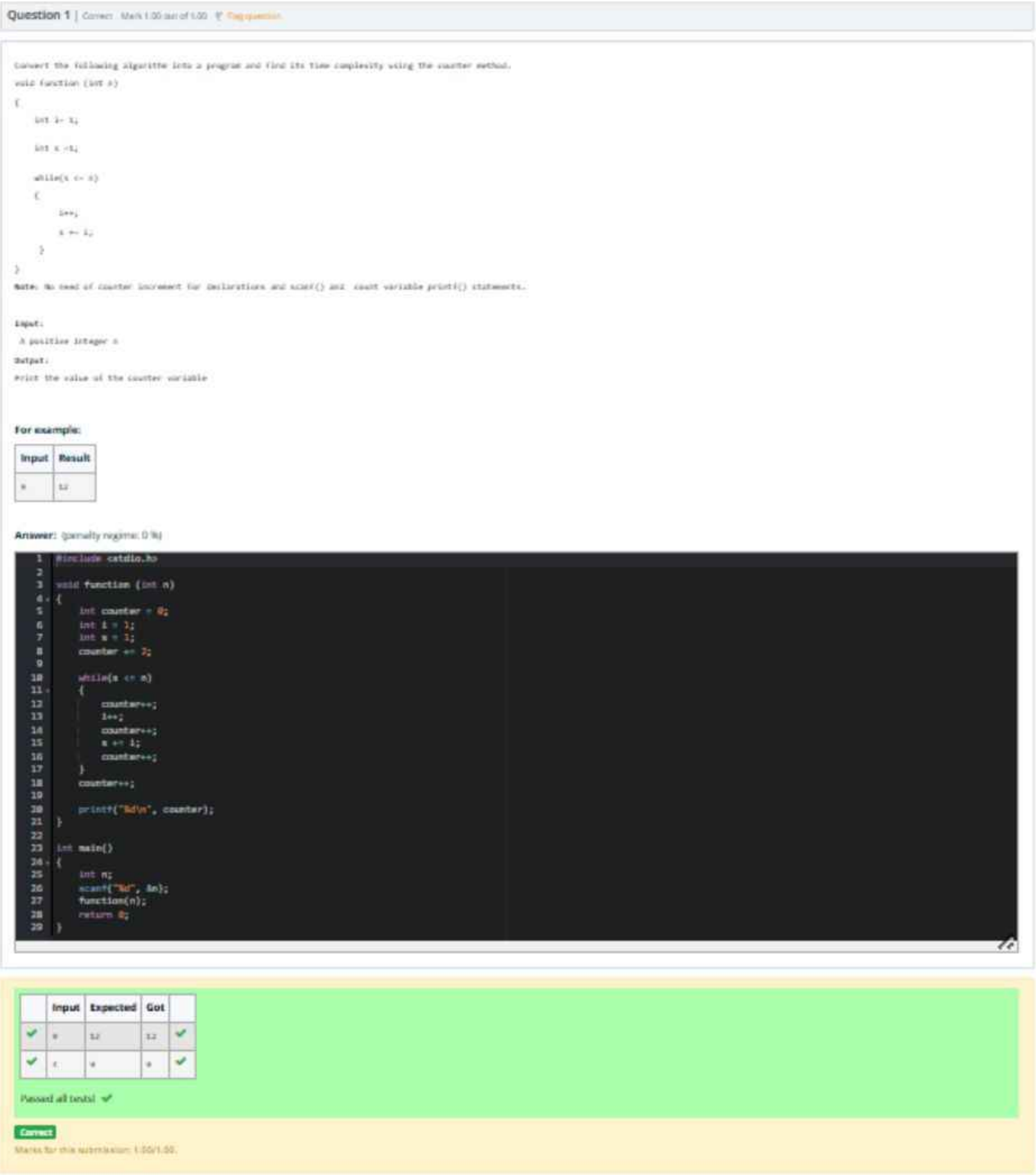
Answer: (penalty regime: 0 %)

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

	Input	Expected	Got	
✓	123	321	321	✓

Passed all tests! ✓

Marks for this submission: 1.00/1.00.



Question 1 | Correct Mark 1.00 out of 1.00 Flag question

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 %)

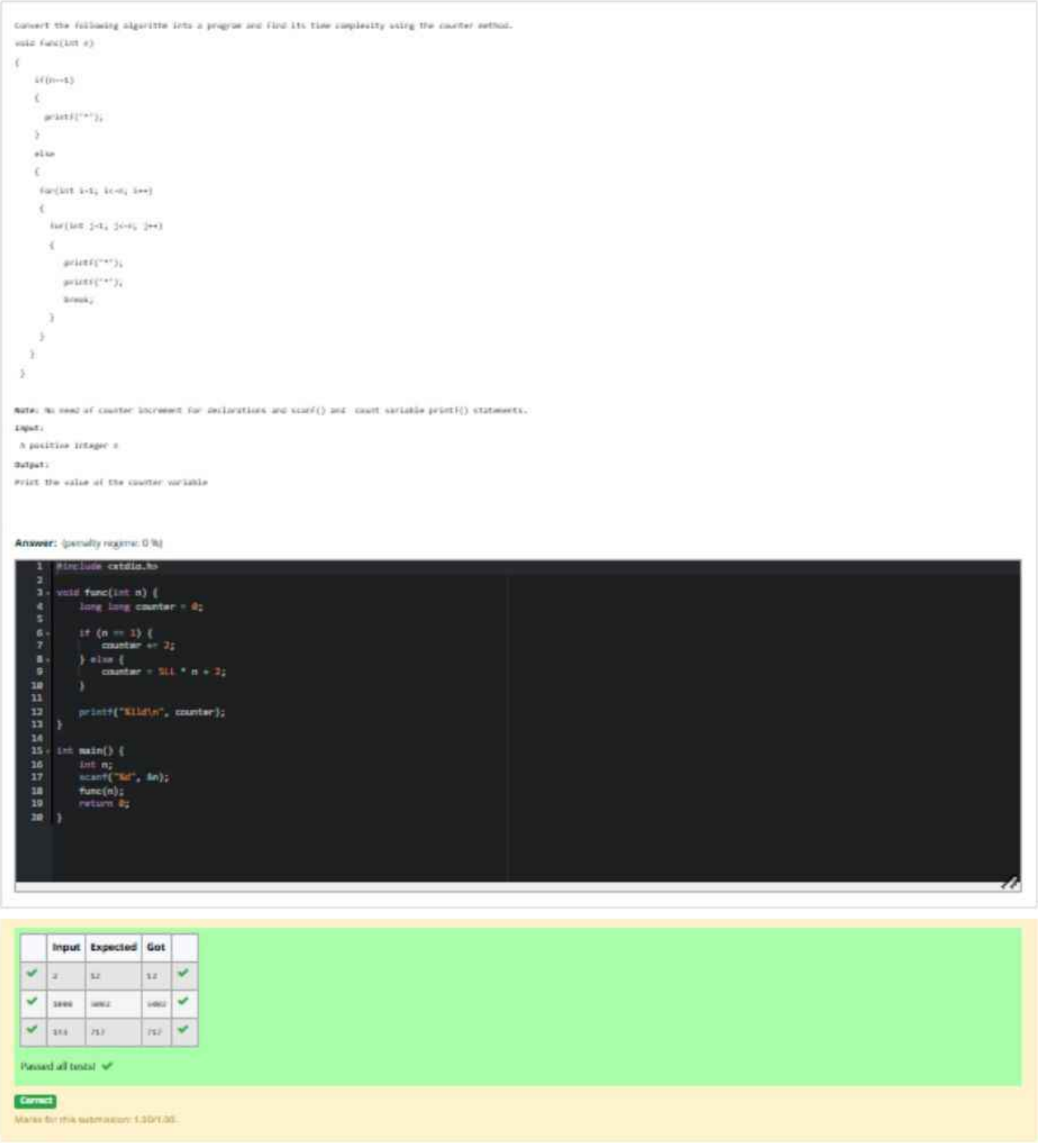
```
#include <stdio.h>

void function (int n)
{
    int counter = 0;
    int i = 1;
    int s = 1;
    counter += 2;

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

    printf("%d\n", counter);
}

int main()
{
    int n;
    scanf("%d", &n);
    function(n);
    return 0;
}
```

	Input	Expected	Got	
✓	9	12	12	✓
✓	4	9	9	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



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 %)

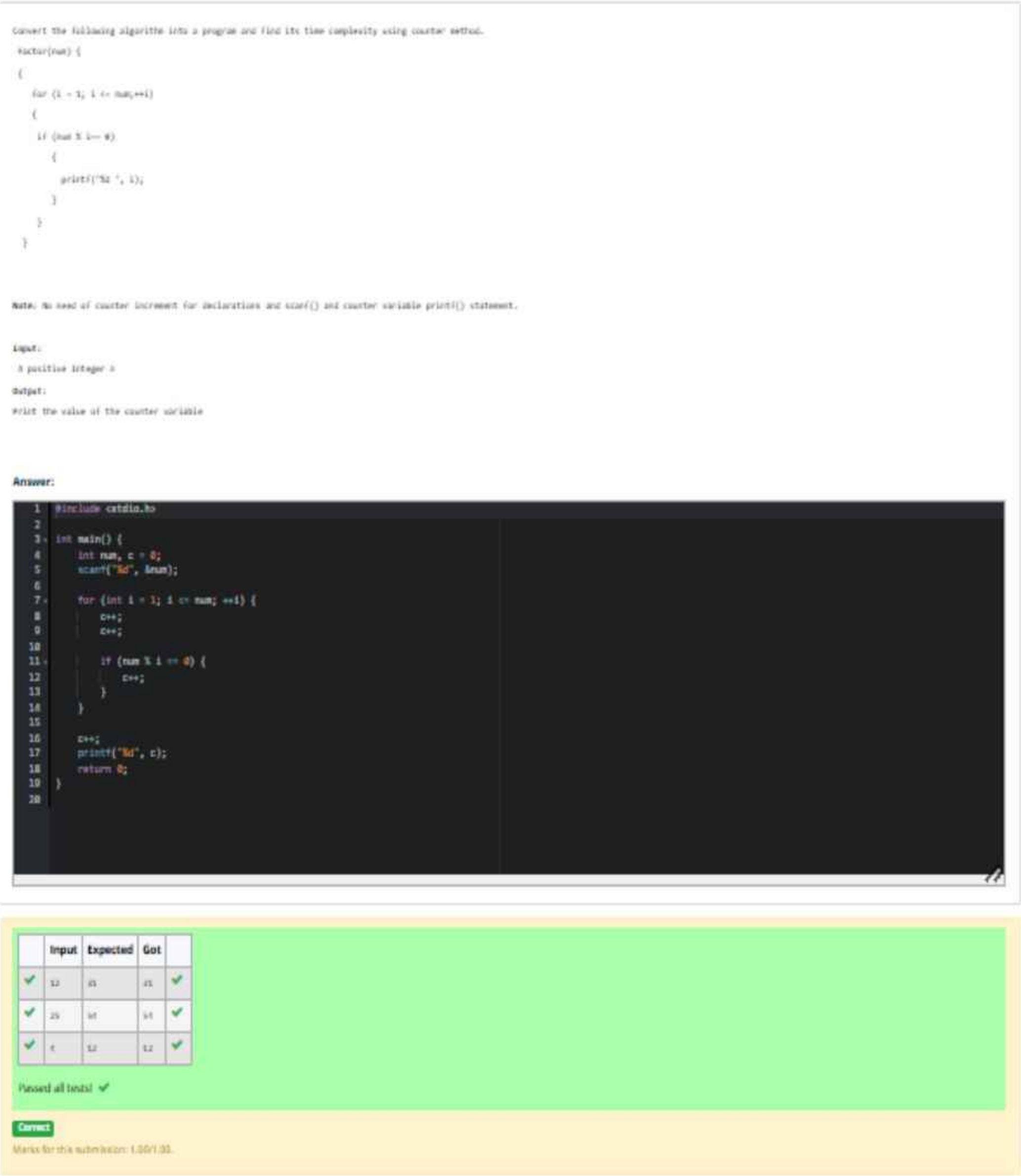
```
#include <stdio.h>

void func(int n) {
    long long counter = 0;

    if (n == 1) {
        counter += 2;
    } else {
        counter = 5LL * n + 2;
    }

    printf("%lld\n", counter);
}

int main() {
    int n;
    scanf("%d", &n);
    func(n);
    return 0;
}
```

	Input	Expected	Got	
✓	2	12	12	✓
✓	1000	5002	5002	✓
✓	151	757	757	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



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

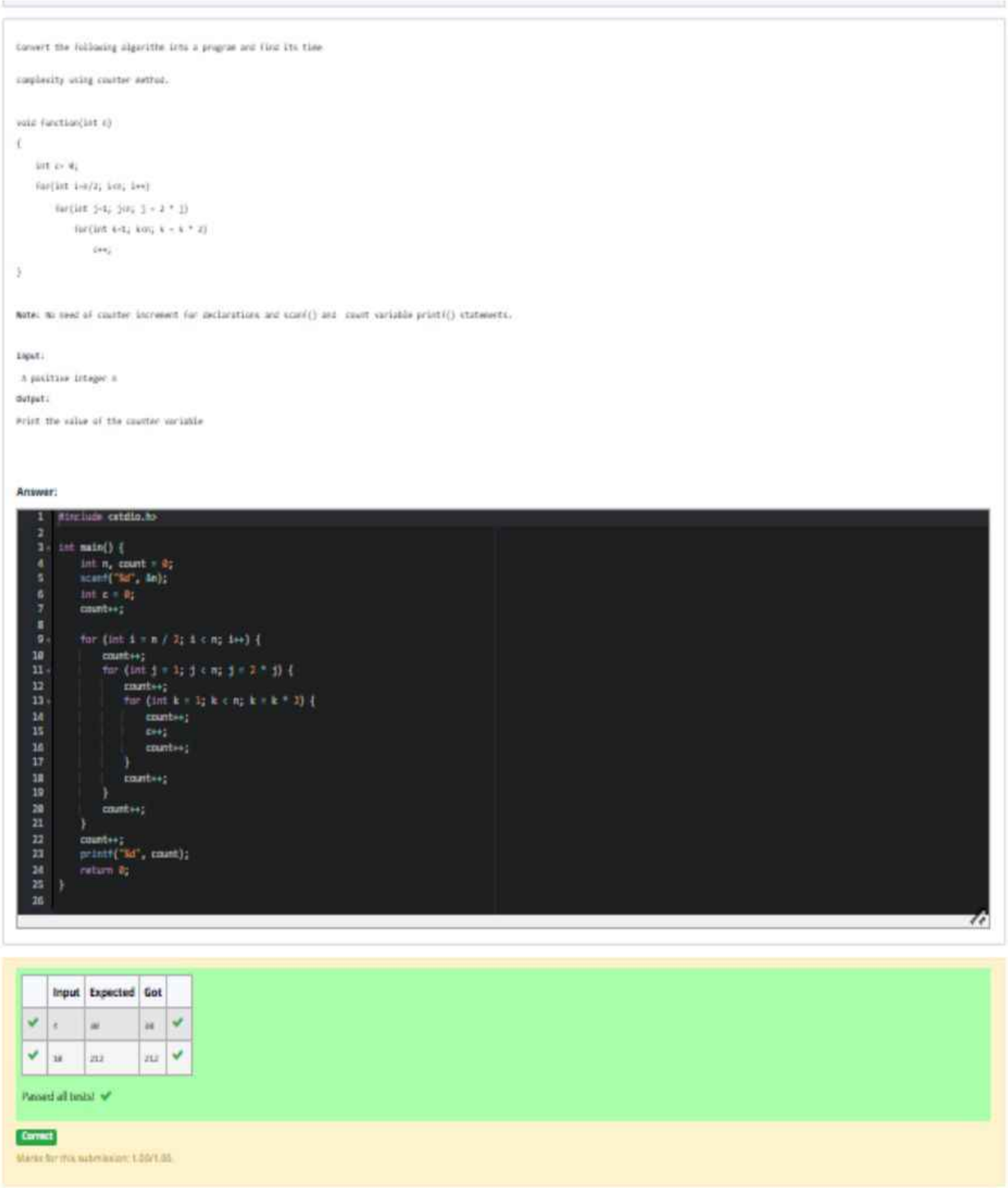
Answer:

```c
#include <stdio.h>

int main() {
    int num, c = 0;
    scanf("%d", &num);

    for (int i = 1; i <= num; ++i) {
        c++;
        c++;

        if (num % i == 0) {
            c++;
        }
    }

    c++;
    printf("%d", c);
    return 0;
}
```

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

Passed all tests! ✓

Marks for this submission: 1.00/1.00.



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:

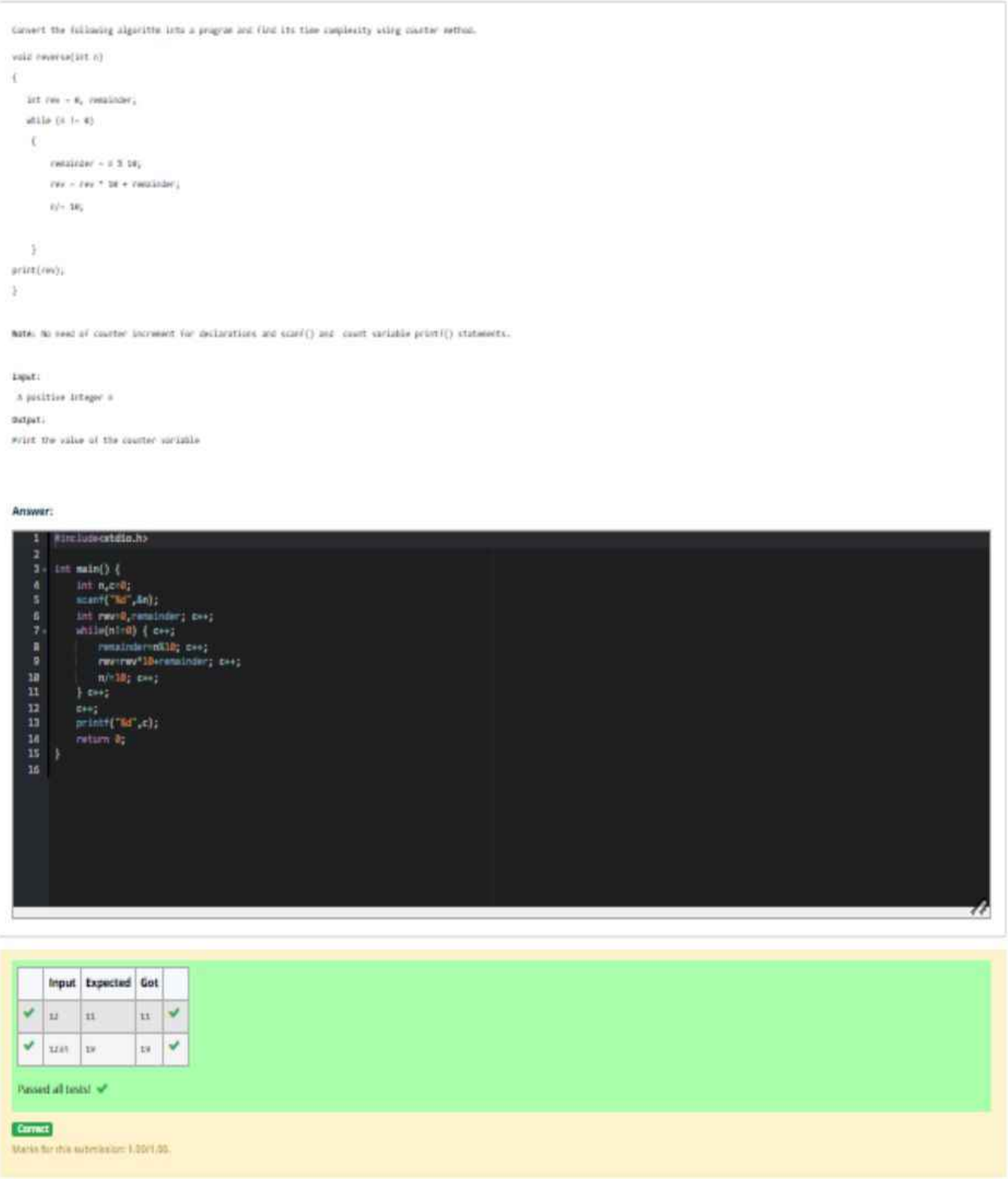
```
#include <stdio.h>

int main() {
    int n, count = 0;
    scanf("%d", &n);
    int c = 0;
    count++;

    for (int i = n / 2; i < n; i++) {
        count++;
        for (int j = 1; j < n; j = 2 * j) {
            count++;
            for (int k = 1; k < n; k = k * 2) {
                count++;
                c++;
                count++;
            }
            count++;
        }
        count++;
    }
    count++;
    printf("%d", count);
    return 0;
}
```

	Input	Expected	Got	
✓	4	30	30	✓
✓	10	212	212	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



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

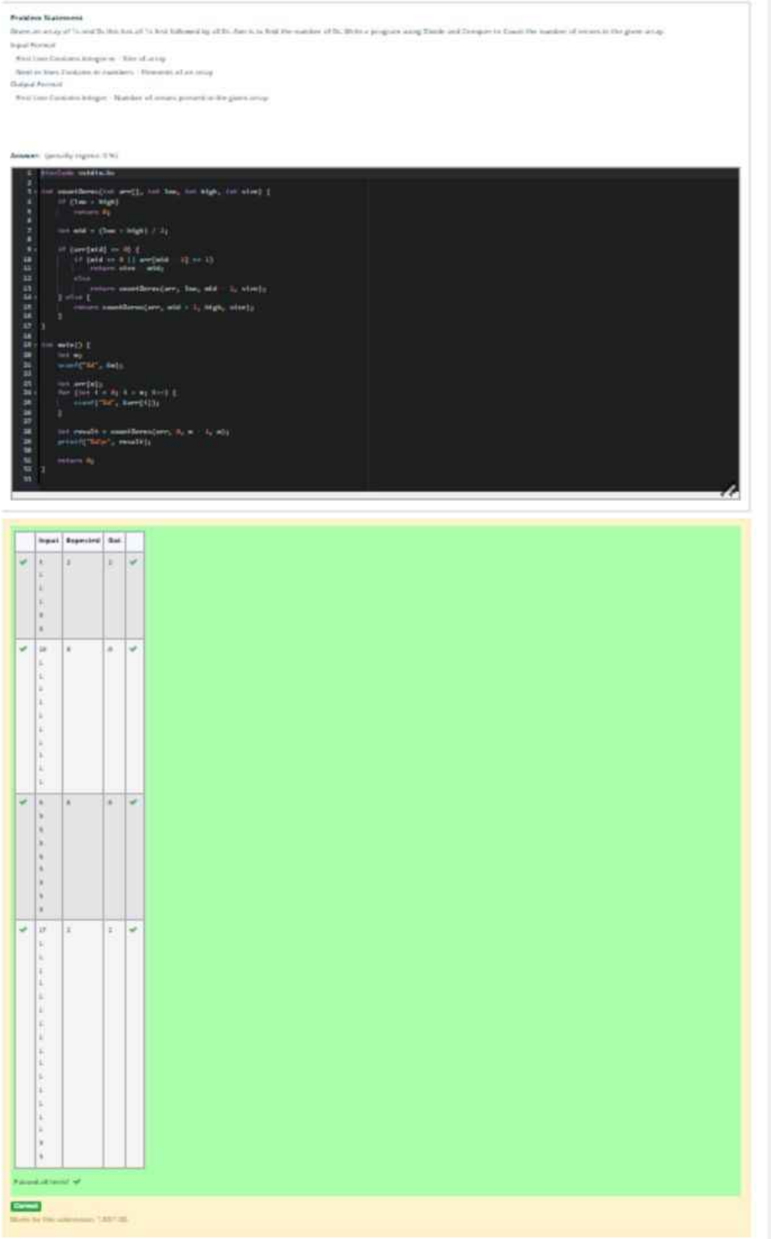
Answer:

```
#include<stdio.h>

int main() {
    int n,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);
    return 0;
}
```

	Input	Expected	Got	
✓	12	11	11	✓
✓	1234	19	19	✓

Passed all tests! ✓

Marks for this submission: 1.00/1.00.



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 %)

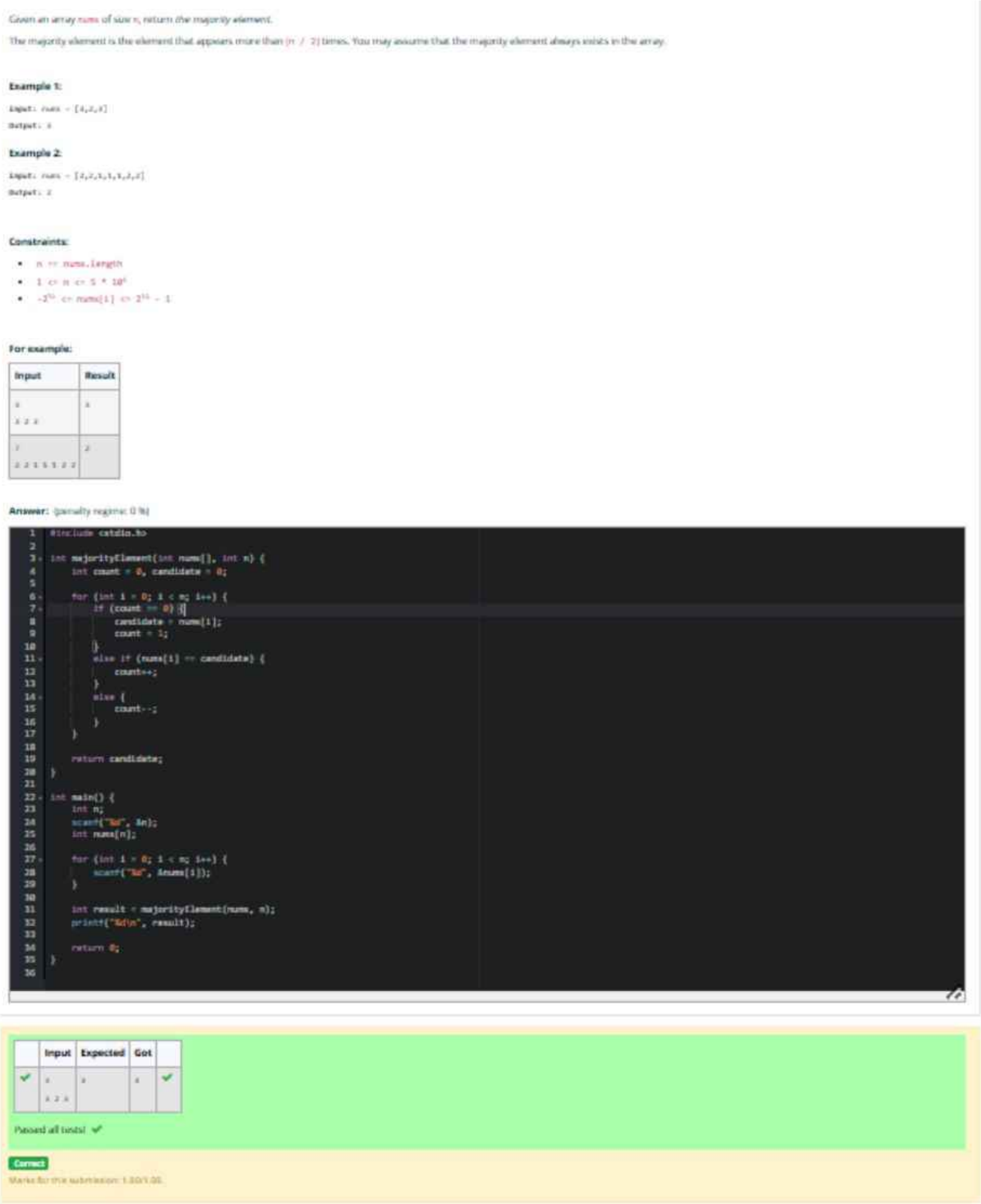
```c
#include <stdio.h>

int countZeroes(int arr[], int low, int high, int size) {
    if (low > high)
        return 0;

    int mid = (low + high) / 2;

    if (arr[mid] == 0) {
        if (mid == 0 || arr[mid - 1] == 1)
            return size - mid;
        else
            return countZeroes(arr, low, mid - 1, size);
    } else {
        return countZeroes(arr, mid + 1, high, size);
    }
}

int main() {
    int m;
    scanf("%d", &m);

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

    int result = countZeroes(arr, 0, m - 1, m);
    printf("%d\n", result);

    return 0;
}

```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



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 %)

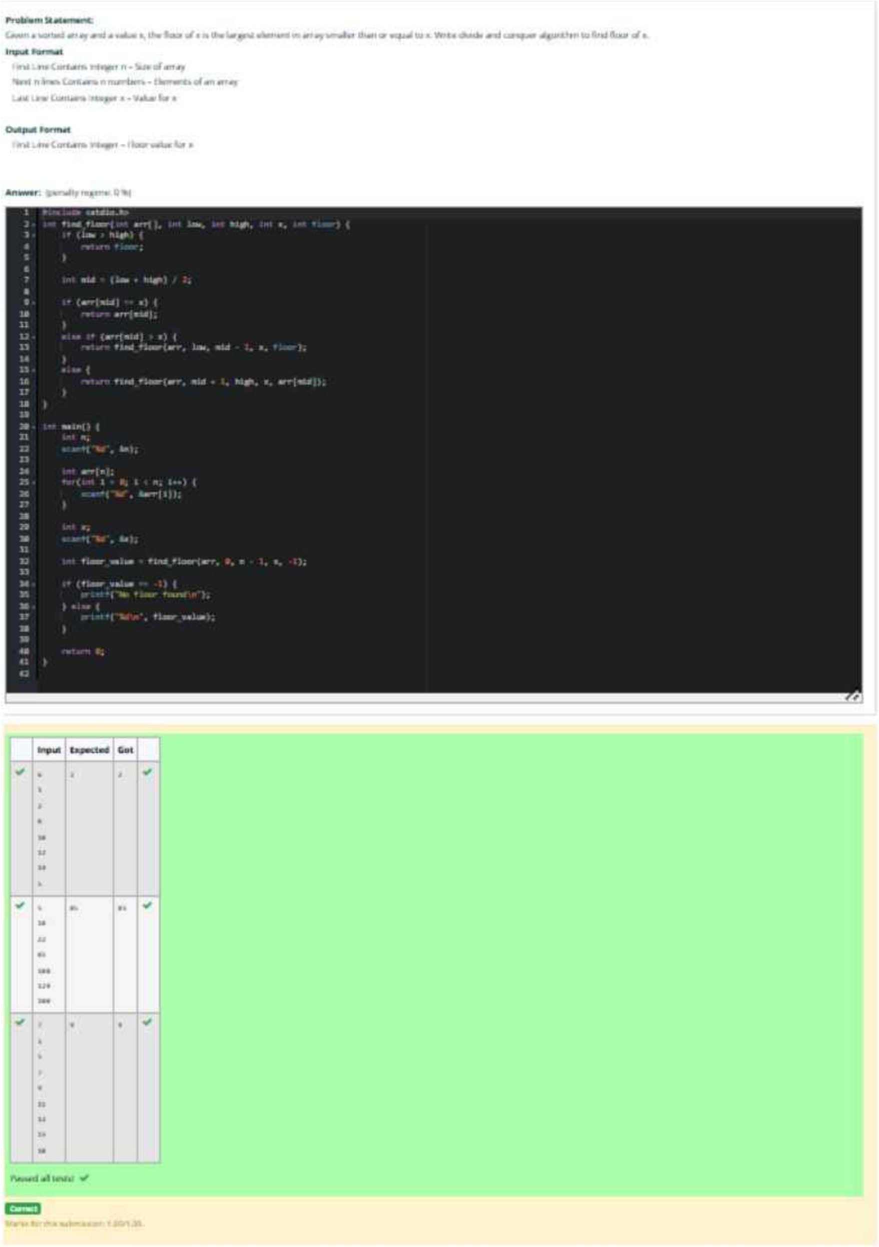
```c
#include <stdio.h>

int majorityElement(int nums[], int n) {
    int count = 0, candidate = 0;

    for (int i = 0; i < n; 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 result = majorityElement(nums, n);
    printf("%d\n", result);

    return 0;
}

```

	Input	Expected	Got	
✓	3 3 2 3	3	3	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



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 %)

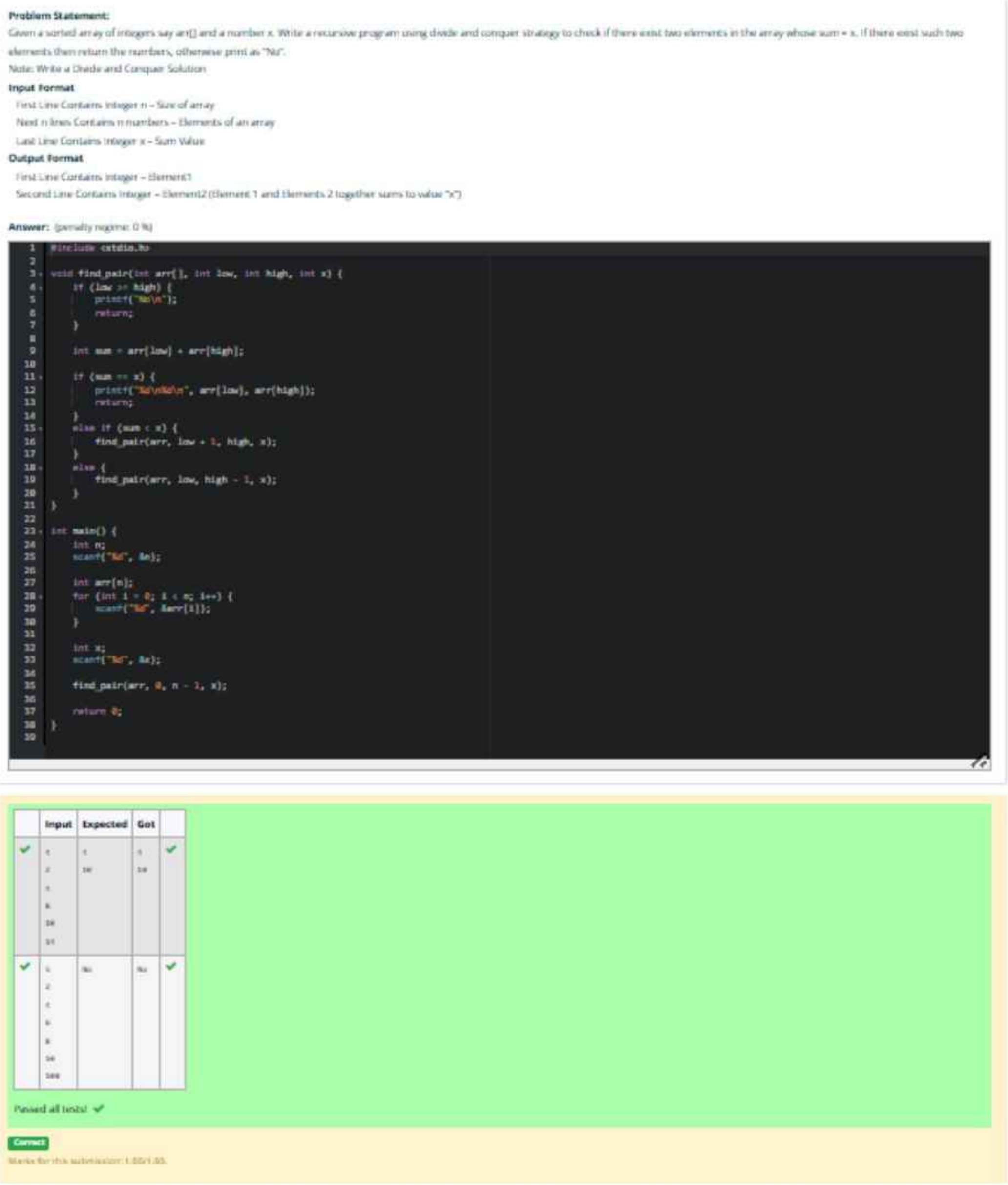
```c
#include <stdio.h>
int find_floor(int arr[], int low, int high, int x, int floor) {
    if (low > high) {
        return floor;
    }

    int mid = (low + high) / 2;

    if (arr[mid] == x) {
        return arr[mid];
    }
    else if (arr[mid] > x) {
        return find_floor(arr, low, mid - 1, x, floor);
    }
    else {
        return find_floor(arr, mid + 1, high, x, arr[mid]);
    }
}

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 floor_value = find_floor(arr, 0, n - 1, x, -1);

    if (floor_value == -1) {
        printf("No floor found\n");
    } else {
        printf("%d\n", floor_value);
    }

    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.



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 %)

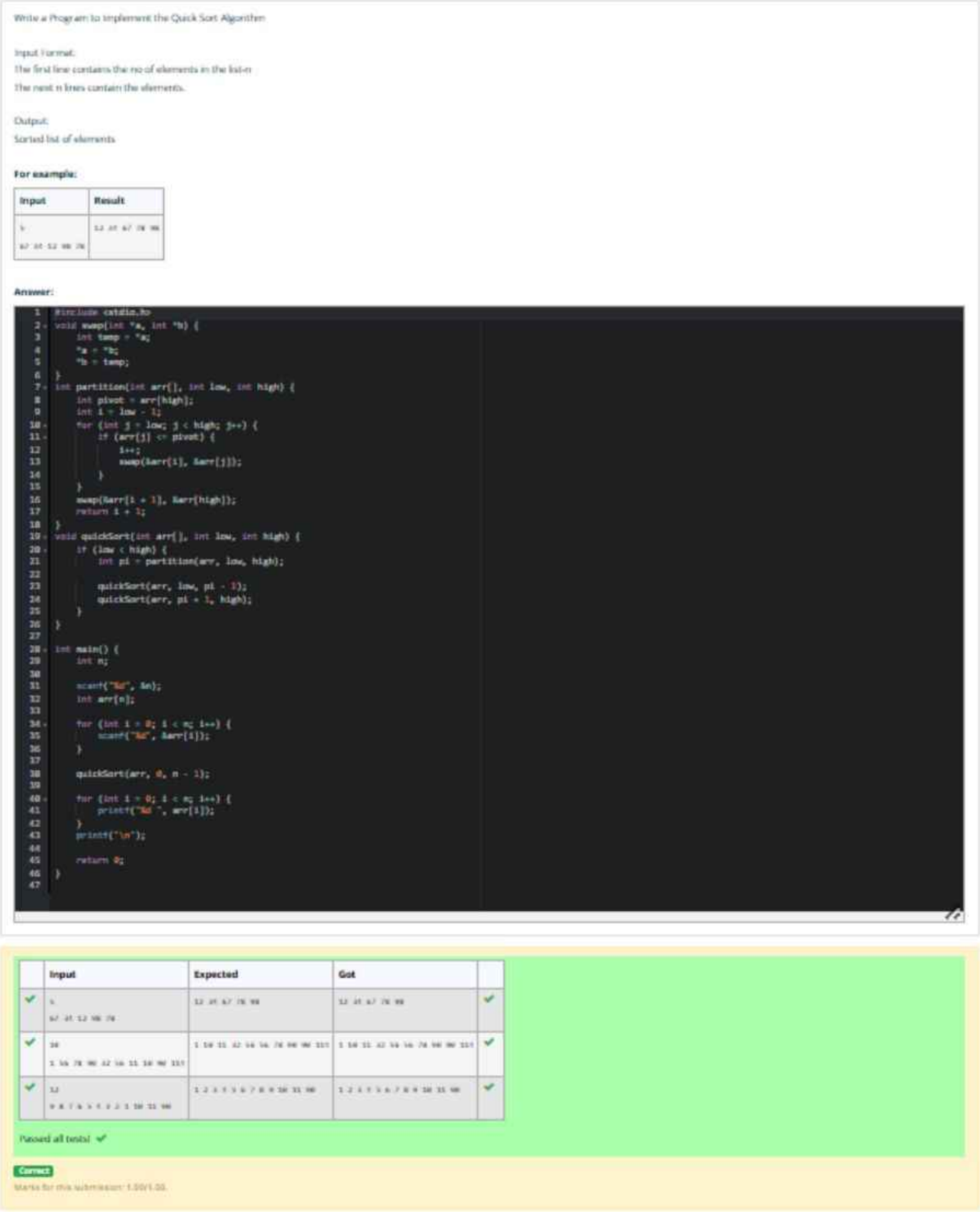
```c
#include <stdio.h>

void find_pair(int arr[], int low, int high, int x) {
    if (low >= high) {
        printf("No\n");
        return;
    }

    int sum = arr[low] + arr[high];

    if (sum == x) {
        printf("%d\n%d\n", arr[low], arr[high]);
        return;
    }
    else if (sum < x) {
        find_pair(arr, low + 1, high, x);
    }
    else {
        find_pair(arr, low, high - 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);

    find_pair(arr, 0, n - 1, x);

    return 0;
}
```

	Input	Expected	Got	
✓	4 2 4 8 10 14	4 10	4 10	✓
✓	5 2 4 6 8 10 100	No	No	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



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

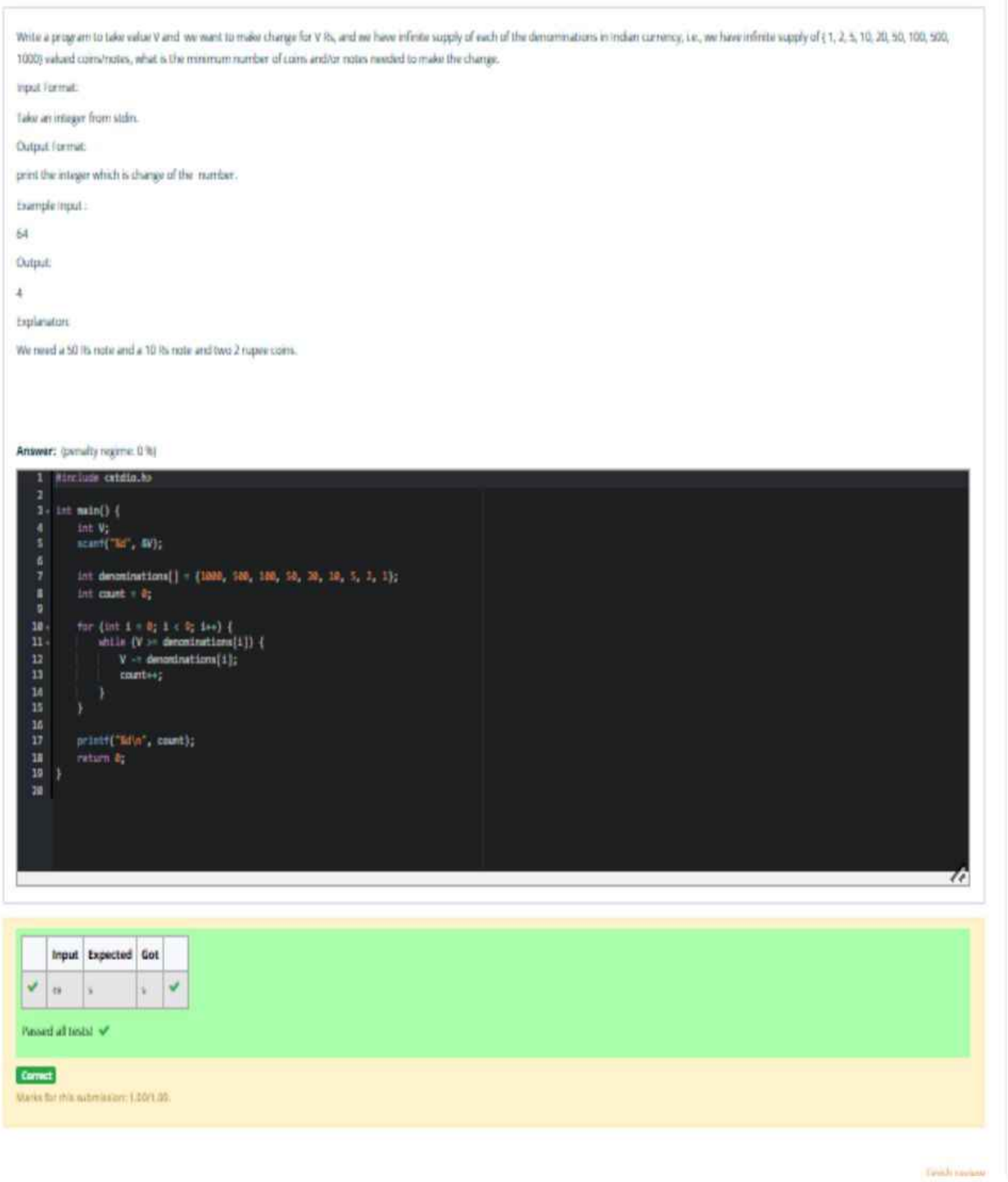
Answer:

```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 90 111	1 10 11 32 56 56 78 90 90 111	1 10 11 32 56 56 78 90 90 111	✓
✓	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! ✓

Marks for this submission: 1.00/1.00.



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.

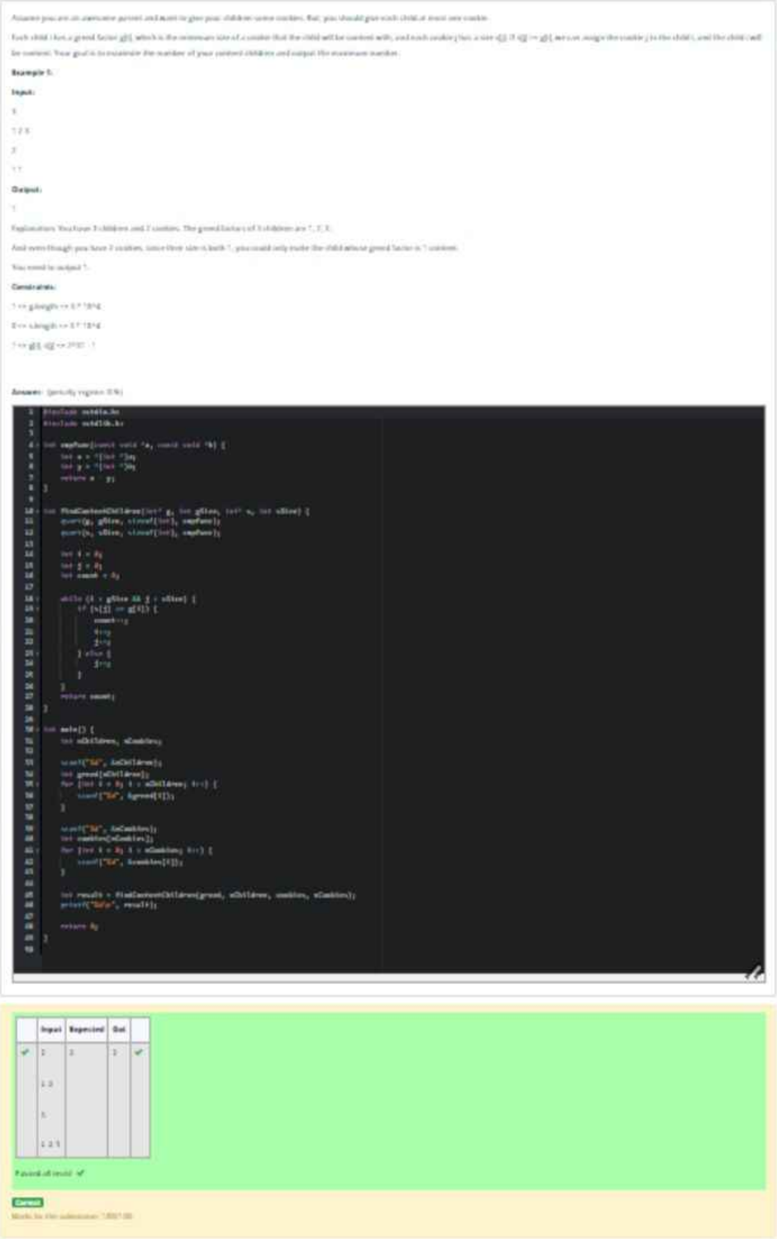
Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

int main() {
    int V;
    scanf("%d", &V);

    int denominations[] = {1000, 500, 100, 50, 20, 10, 5, 2, 1};
    int count = 0;

    for (int i = 0; i < 9; i++) {
        while (V >= denominations[i]) {
            V -= denominations[i];
            count++;
        }
    }

    printf("%d\n", count);
    return 0;
}
```

	Input	Expected	Got	
✓	64	4	4	✓

Passed all tests! ✓

Marks for this submission: 1.00/1.00.



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 %)

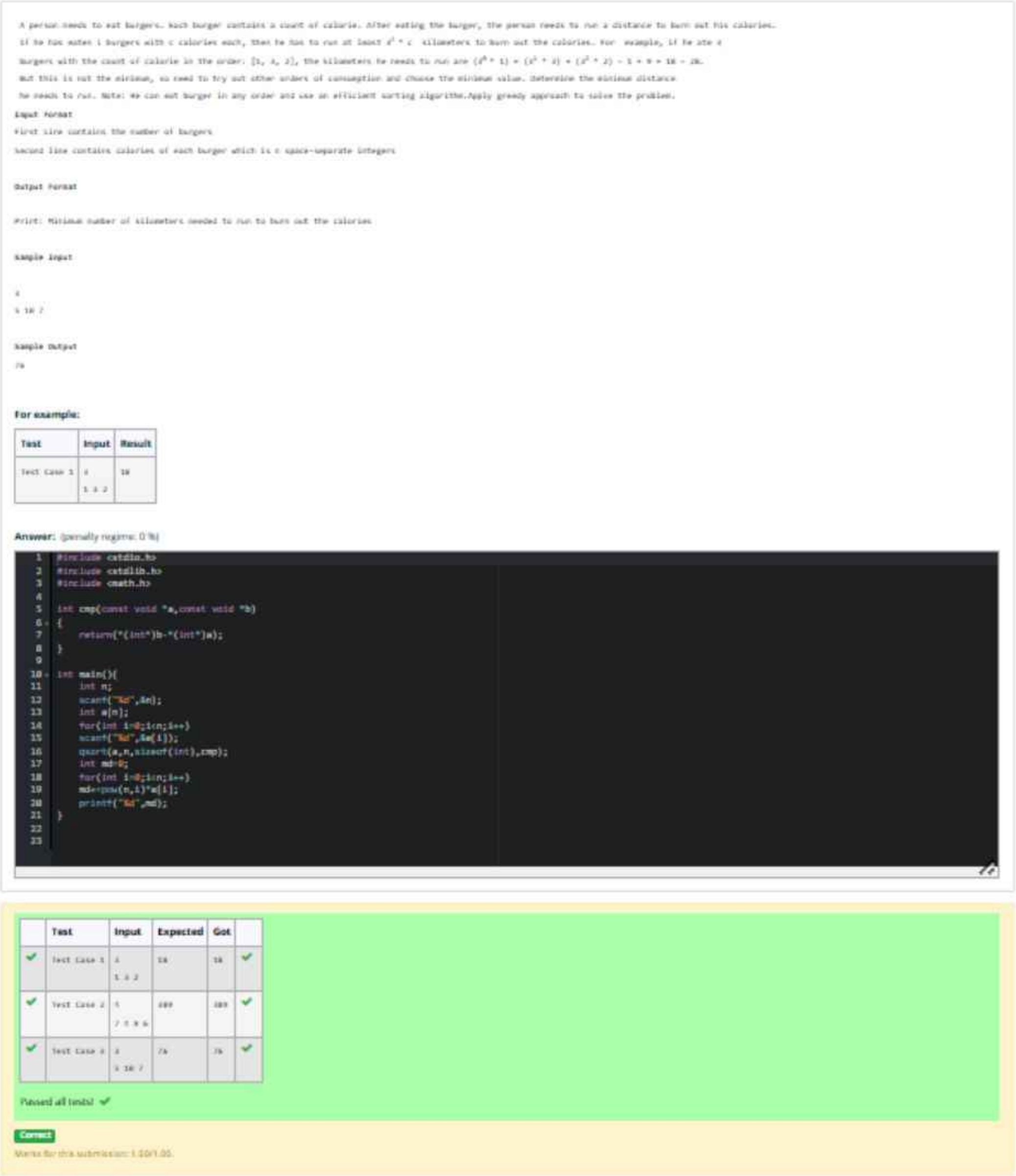
```c
#include <stdio.h>
#include <stdlib.h>

int cmpfunc(const void *a, const void *b) {
    int x = *(int *)a;
    int y = *(int *)b;
    return x - y;
}

int findContentChildren(int* g, int gSize, int* s, int sSize) {
    qsort(g, gSize, sizeof(int), cmpfunc);
    qsort(s, sSize, sizeof(int), cmpfunc);

    int i = 0;
    int j = 0;
    int count = 0;

    while (i < gSize && j < sSize) {
        if (s[j] >= g[i]) {
            count++;
            i++;
            j++;
        } else {
            j++;
        }
    }
    return count;
}

int main() {
    int nChildren, nCookies;

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

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

    int result = findContentChildren(greed, nChildren, cookies, nCookies);
    printf("%d\n", result);

    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.



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 %)

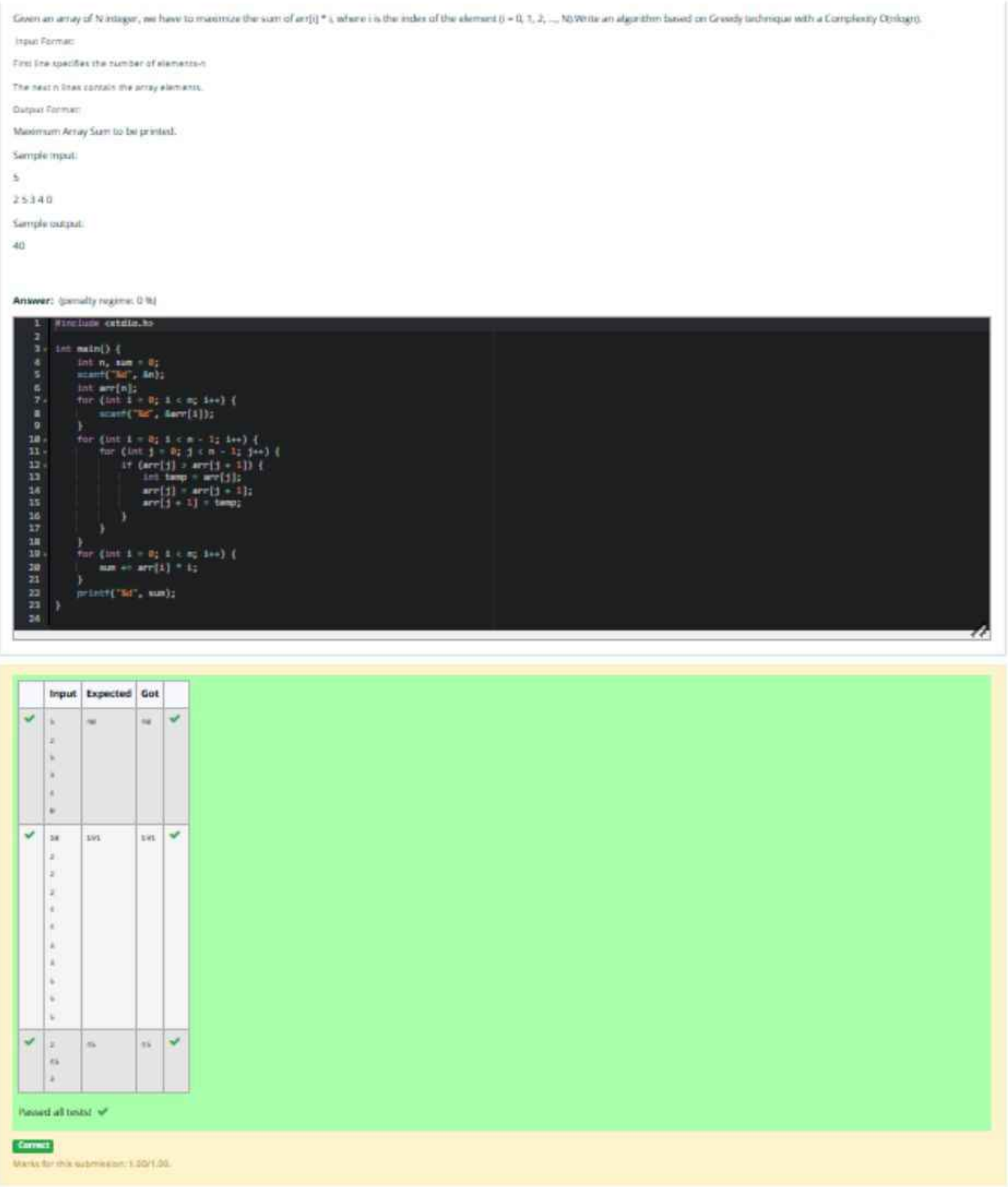
```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int cmp(const void *a,const void *b)
{
    return(*(int*)b-*(int*)a);
}

int main(){
    int n;
    scanf("%d",&n);
    int a[n];
    for(int i=0;i<n;i++)
    scanf("%d",&a[i]);
    qsort(a,n,sizeof(int),cmp);
    int md=0;
    for(int i=0;i<n;i++)
    md+=pow(n,i)*a[i];
    printf("%d",md);
}
```

	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.



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 %)

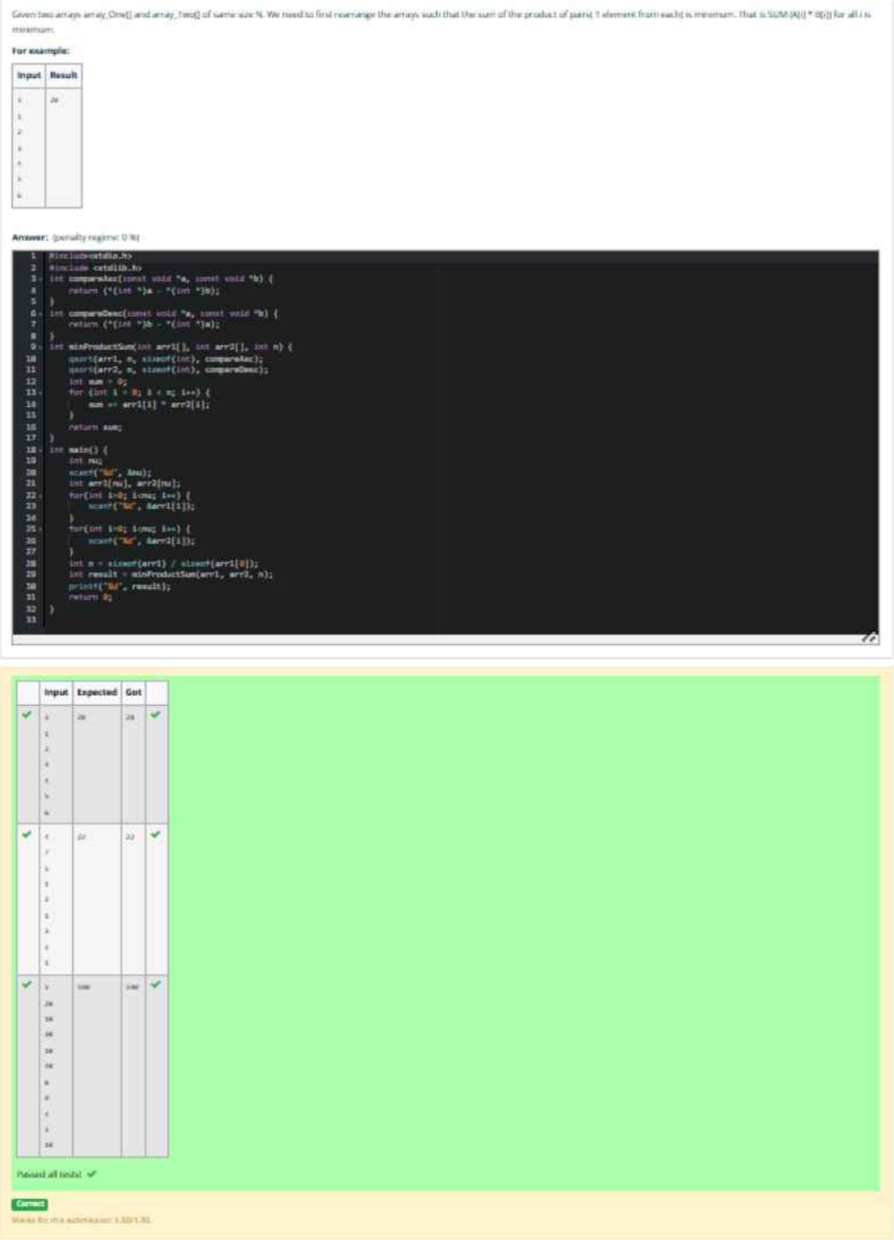
```
#include <stdio.h>

int main() {
    int n, sum = 0;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
    for (int i = 0; i < n; i++) {
        sum += arr[i] * i;
    }
    printf("%d", sum);
}
```

	Input	Expected	Got	
✓	5 2 5 3 4 0	40	40	✓
✓	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.



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 %)

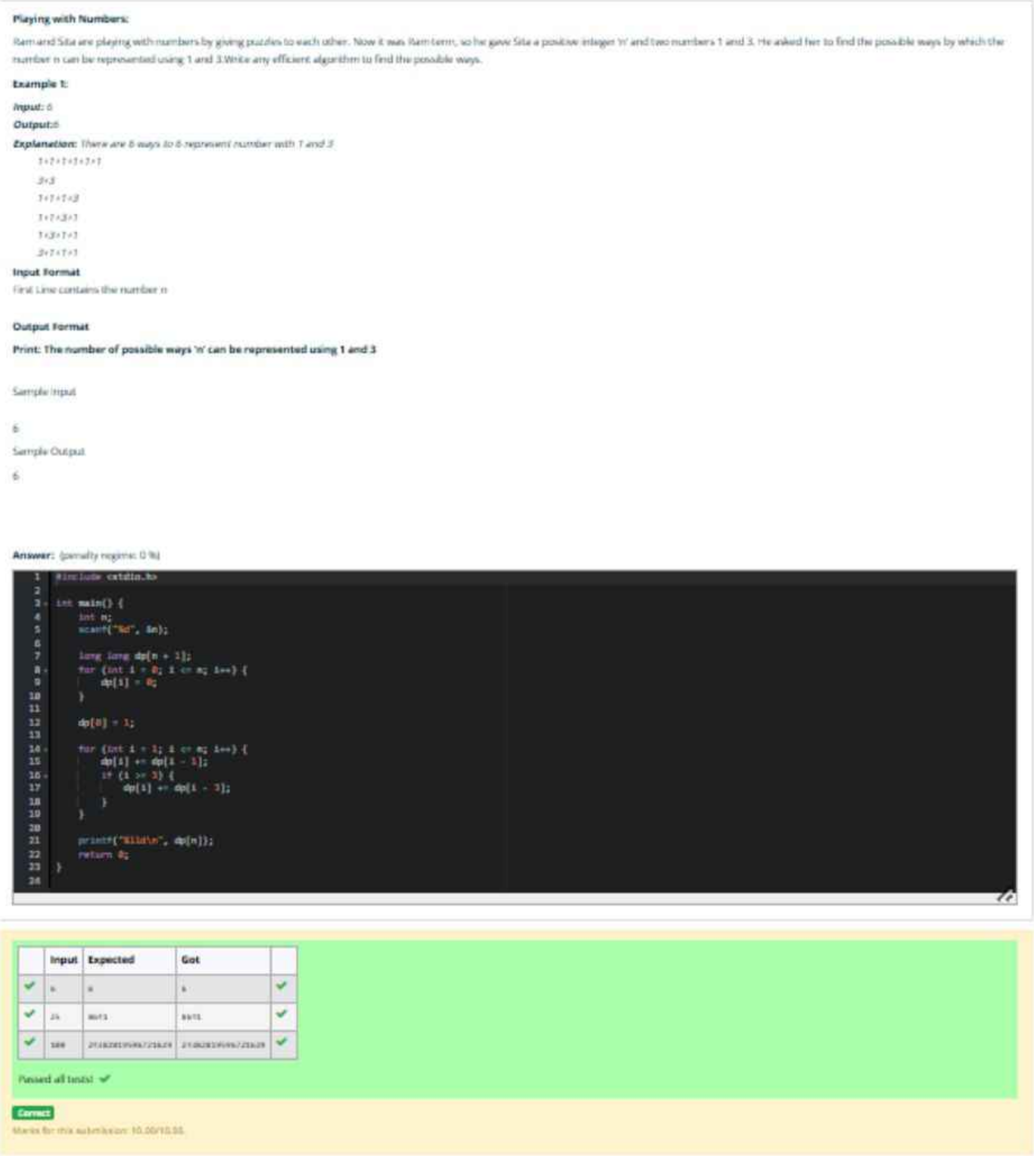
```c
#include<stdio.h>
#include <stdlib.h>
int compareAsc(const void *a, const void *b) {
    return (*(int *)a - *(int *)b);
}
int compareDesc(const void *a, const void *b) {
    return (*(int *)b - *(int *)a);
}
int minProductSum(int arr1[], int arr2[], int n) {
    qsort(arr1, n, sizeof(int), compareAsc);
    qsort(arr2, n, sizeof(int), compareDesc);
    int sum = 0;
    for (int i = 0; i < n; i++) {
        sum += arr1[i] * arr2[i];
    }
    return sum;
}
int main() {
    int nu;
    scanf("%d", &nu);
    int arr1[nu], arr2[nu];
    for(int i=0; i<nu; i++) {
        scanf("%d", &arr1[i]);
    }
    for(int i=0; i<nu; i++) {
        scanf("%d", &arr2[i]);
    }
    int n = sizeof(arr1) / sizeof(arr1[0]);
    int result = minProductSum(arr1, arr2, n);
    printf("%d", result);
    return 0;
}

```

	Input	Expected	Got	
✓	3 1 2 3 4 5 6	28	28	✓
✓	4 7 5 1 2 1 3 4 1	22	22	✓
✓	5 20 10 30 10 70 8 9 4 5 10	[illegible]	[illegible]	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



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 %)

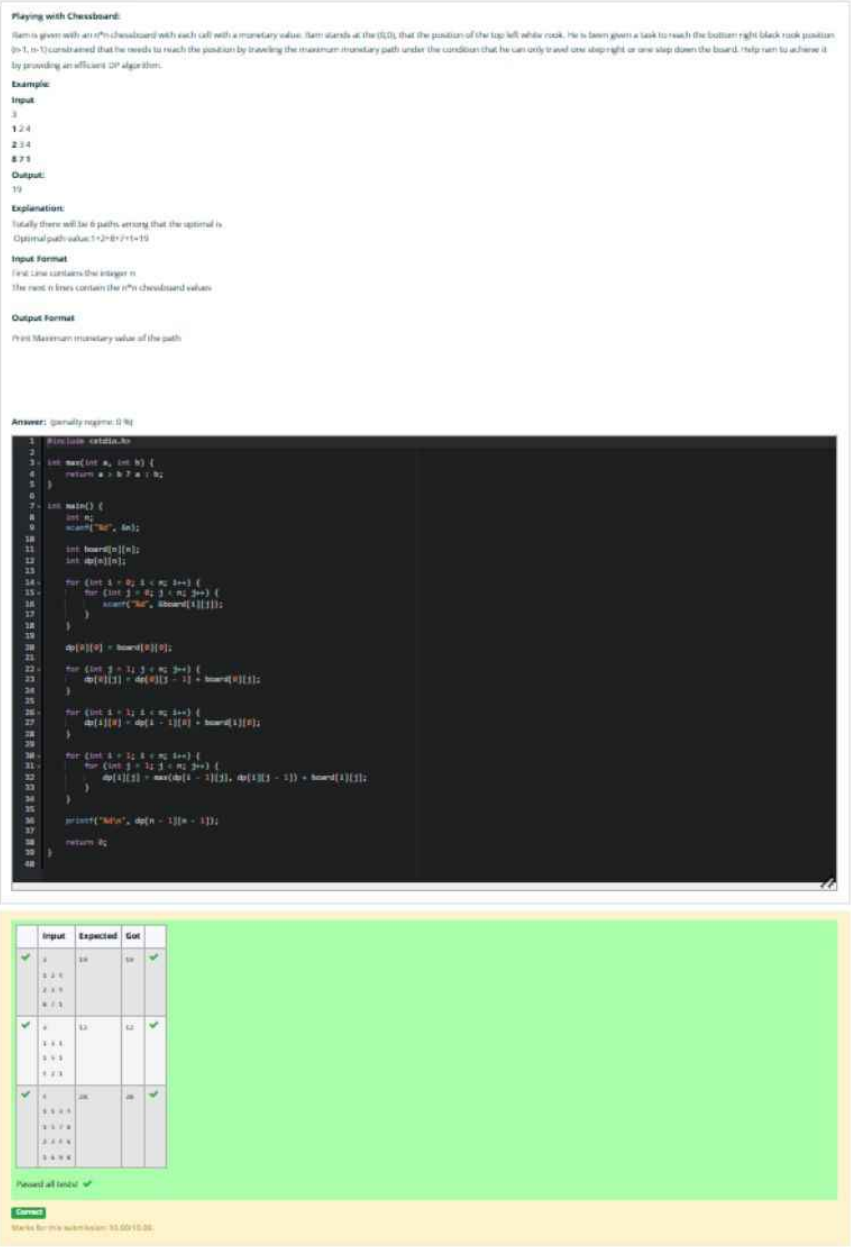
```
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);

    long long dp[n + 1];
    for (int i = 0; i <= n; i++) {
        dp[i] = 0;
    }

    dp[0] = 1;

    for (int i = 1; i <= n; i++) {
        dp[i] += dp[i - 1];
        if (i >= 3) {
            dp[i] += dp[i - 3];
        }
    }

    printf("%lld\n", dp[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.



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

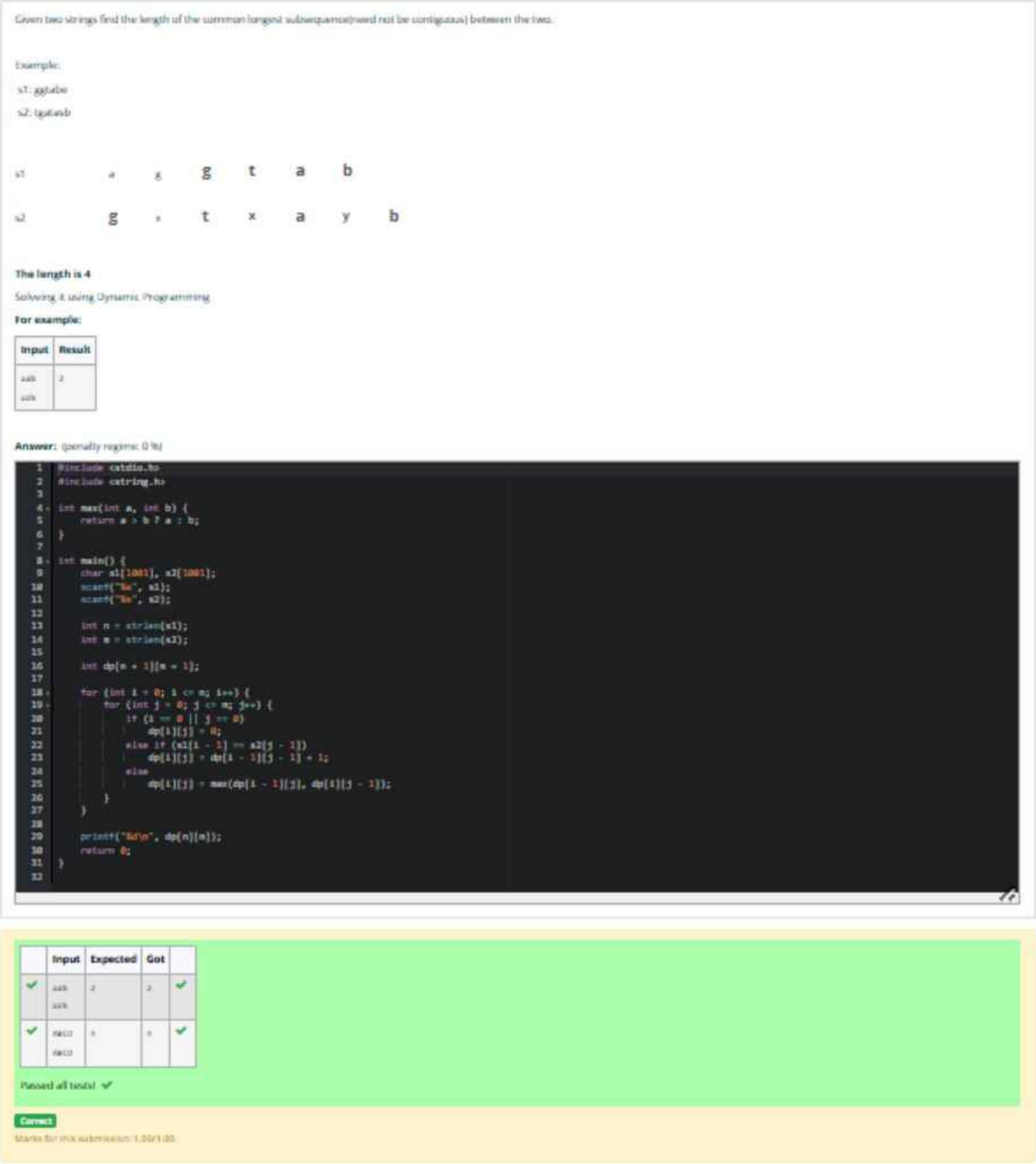
Answer: (penalty regime: 0 %)

```
#include <stdio.h>

int max(int a, int b) {
    return a > b ? a : b;
}

int main() {
    int n;
    scanf("%d", &n);

    int board[n][n];
    int dp[n][n];

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

    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] = max(dp[i - 1][j], dp[i][j - 1]) + board[i][j];
        }
    }

    printf("%d\n", dp[n - 1][n - 1]);

    return 0;
}
```

	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! ✓

Marks for this submission: 10.00/10.00.



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

Example:

s1: aggtab

s2: gxtxayb

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

The length is 4

Solving it using Dynamic Programming

For example:

Input	Result
aab azb	2

Answer: (penalty regime: 0 %)

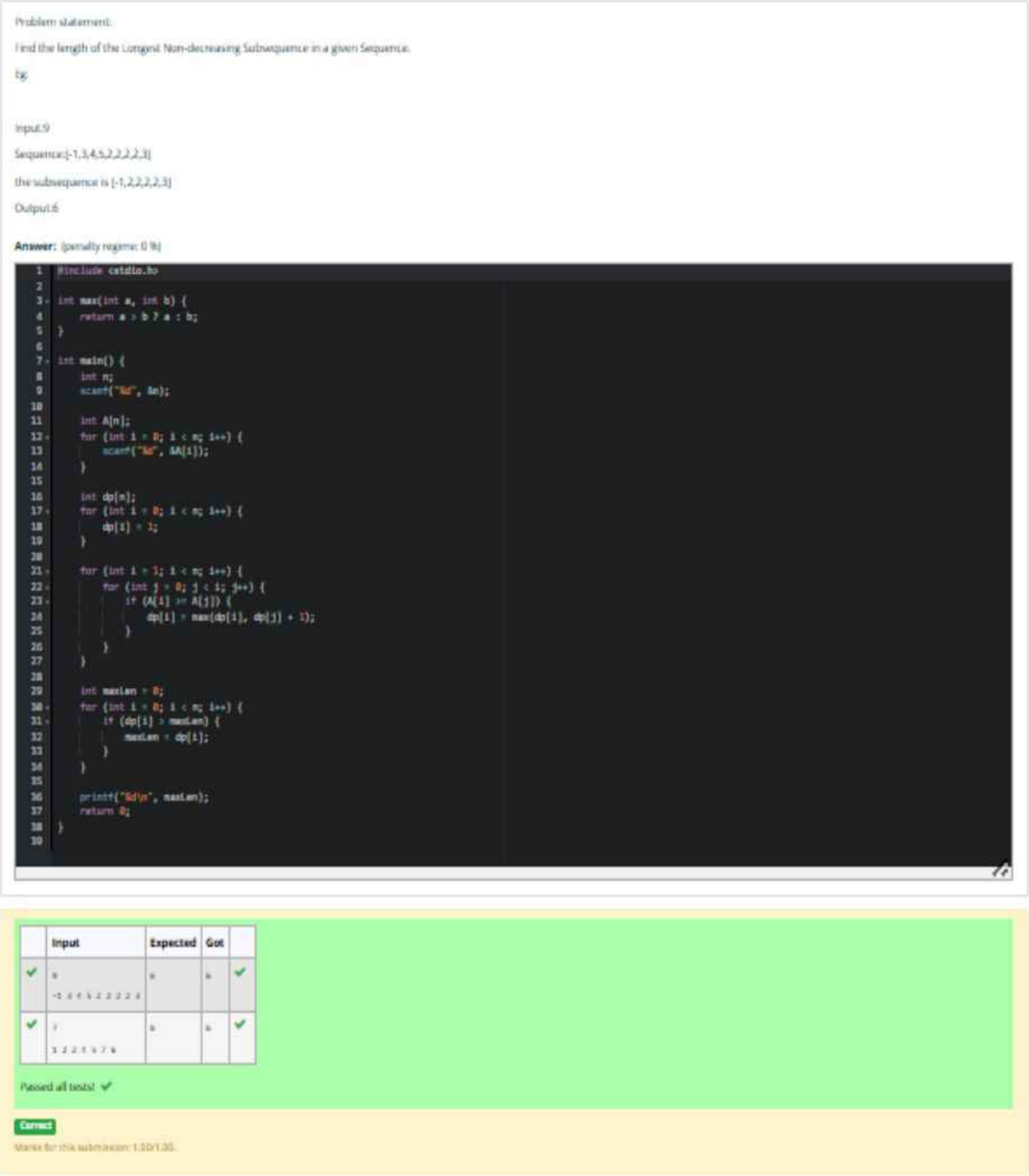
```c
#include <stdio.h>
#include <string.h>

int max(int a, int b) {
    return a > b ? a : b;
}

int main() {
    char s1[1001], s2[1001];
    scanf("%s", s1);
    scanf("%s", s2);

    int n = strlen(s1);
    int m = strlen(s2);

    int dp[n + 1][m + 1];

    for (int i = 0; i <= n; i++) {
        for (int j = 0; j <= m; 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] = max(dp[i - 1][j], dp[i][j - 1]);
        }
    }

    printf("%d\n", dp[n][m]);
    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.



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 %)

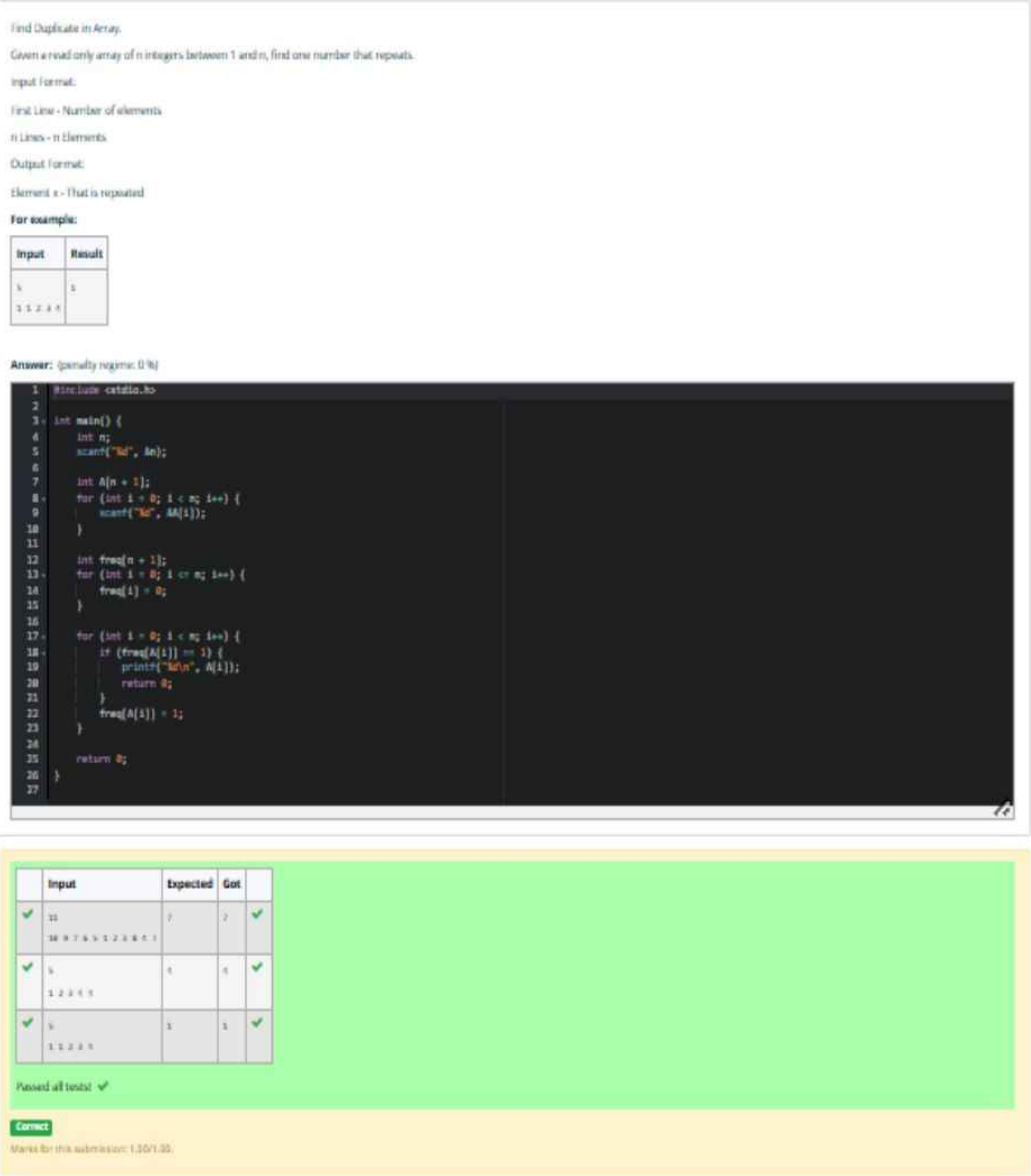
```
#include <stdio.h>

int max(int a, int b) {
    return a > b ? a : b;
}

int main() {
    int n;
    scanf("%d", &n);

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

    int dp[n];
    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 (A[i] >= A[j]) {
                dp[i] = max(dp[i], dp[j] + 1);
            }
        }
    }

    int maxLen = 0;
    for (int i = 0; i < n; i++) {
        if (dp[i] > maxLen) {
            maxLen = dp[i];
        }
    }

    printf("%d\n", maxLen);
    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.



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 %)

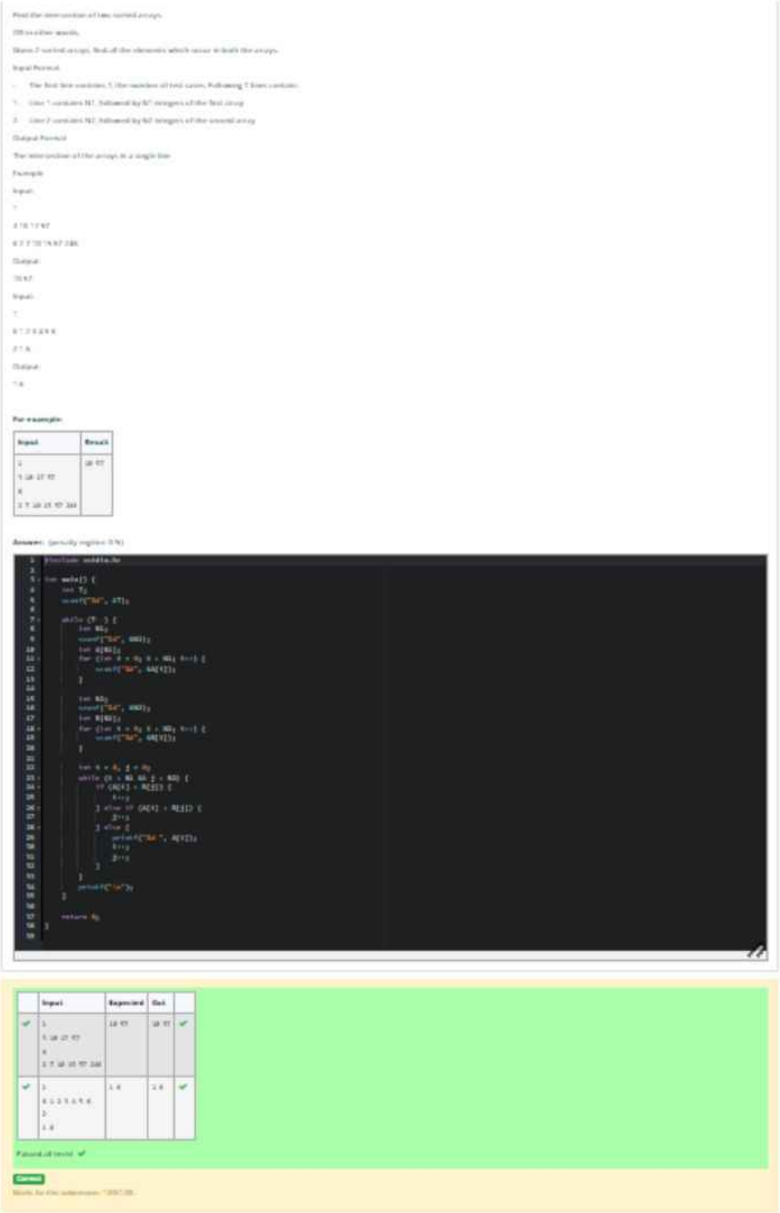
```c
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);

    int A[n + 1];
    for (int i = 0; i < n; i++) {
        scanf("%d", &A[i]);
    }

    int freq[n + 1];
    for (int i = 0; i <= n; i++) {
        freq[i] = 0;
    }

    for (int i = 0; i < n; i++) {
        if (freq[A[i]] == 1) {
            printf("%d\n", A[i]);
            return 0;
        }
        freq[A[i]] = 1;
    }

    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.



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 %)

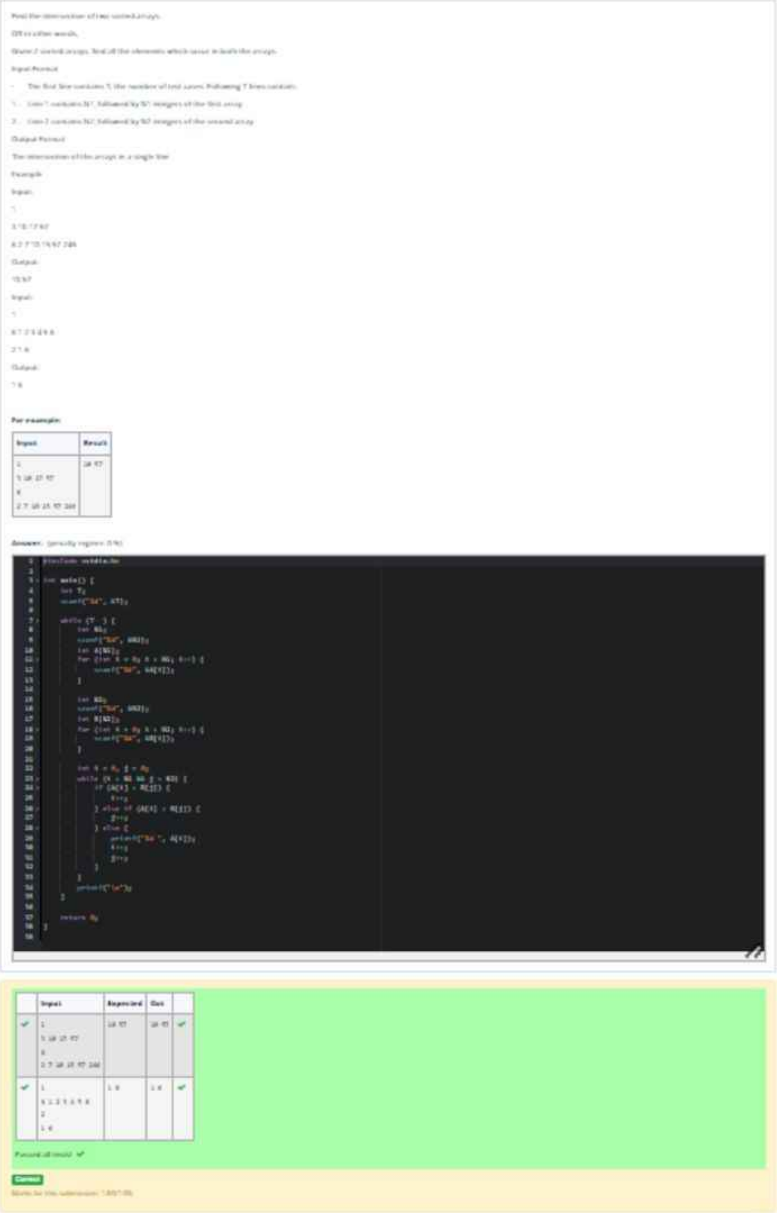
```c
#include <stdio.h>

int main() {
    int T;
    scanf("%d", &T);

    while (T--) {
        int N1;
        scanf("%d", &N1);
        int A[N1];
        for (int i = 0; i < N1; i++) {
            scanf("%d", &A[i]);
        }

        int N2;
        scanf("%d", &N2);
        int B[N2];
        for (int i = 0; i < N2; i++) {
            scanf("%d", &B[i]);
        }

        int i = 0, j = 0;
        while (i < N1 && j < N2) {
            if (A[i] < B[j]) {
                i++;
            } else if (A[i] > B[j]) {
                j++;
            } else {
                printf("%d ", A[i]);
                i++;
                j++;
            }
        }
        printf("\n");
    }

    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.



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

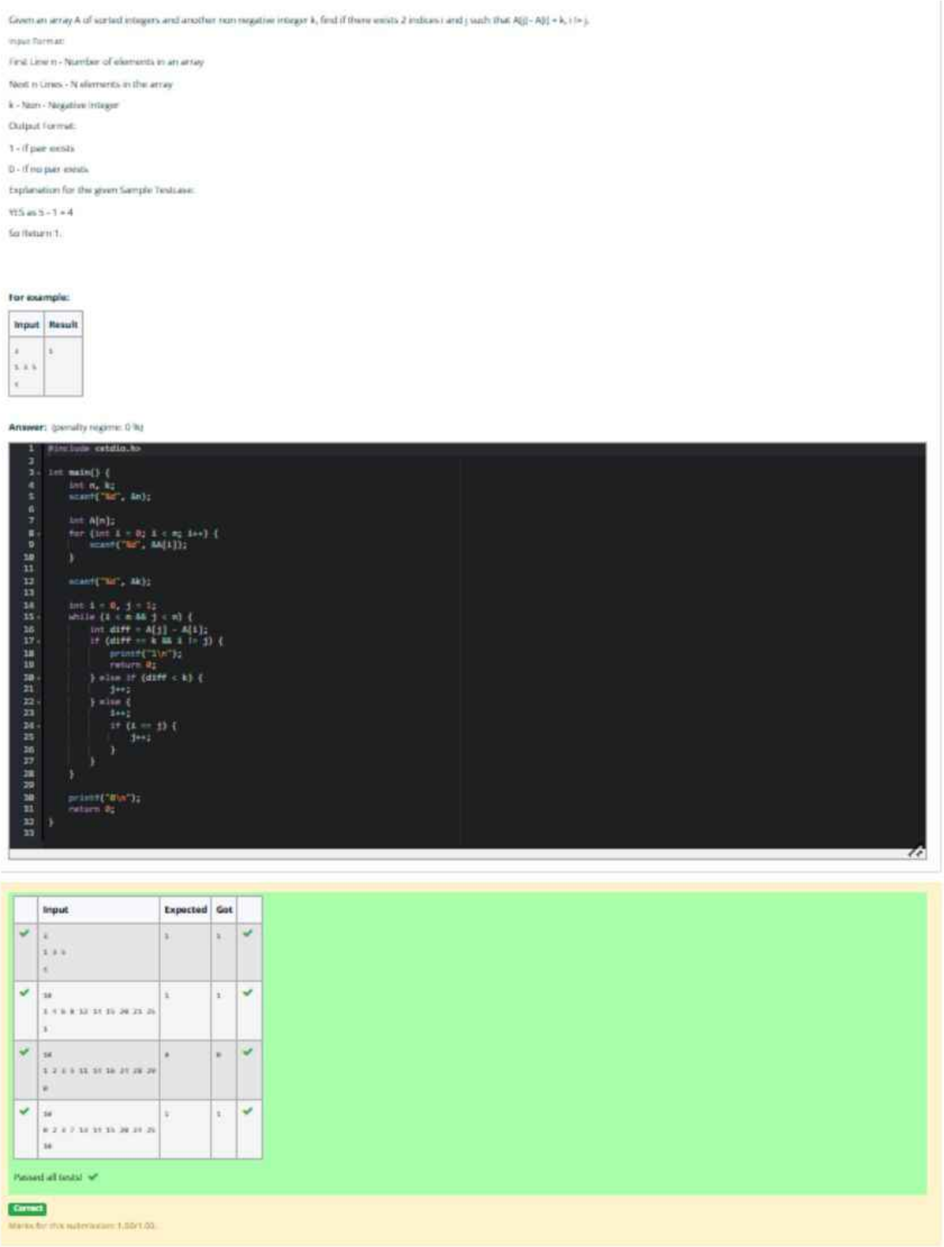
Answer: (penalty regime: 0 %)

```
#include <stdio.h>

int main() {
    int T;
    scanf("%d", &T);

    while (T--) {
        int N1;
        scanf("%d", &N1);
        int A[N1];
        for (int i = 0; i < N1; i++) {
            scanf("%d", &A[i]);
        }

        int N2;
        scanf("%d", &N2);
        int B[N2];
        for (int i = 0; i < N2; i++) {
            scanf("%d", &B[i]);
        }

        int i = 0, j = 0;
        while (i < N1 && j < N2) {
            if (A[i] < B[j]) {
                i++;
            } else if (A[i] > B[j]) {
                j++;
            } else {
                printf("%d ", A[i]);
                i++;
                j++;
            }
        }
        printf("\n");
    }

    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! ✓

Marks for this submission: 1.00/1.00.



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 %)

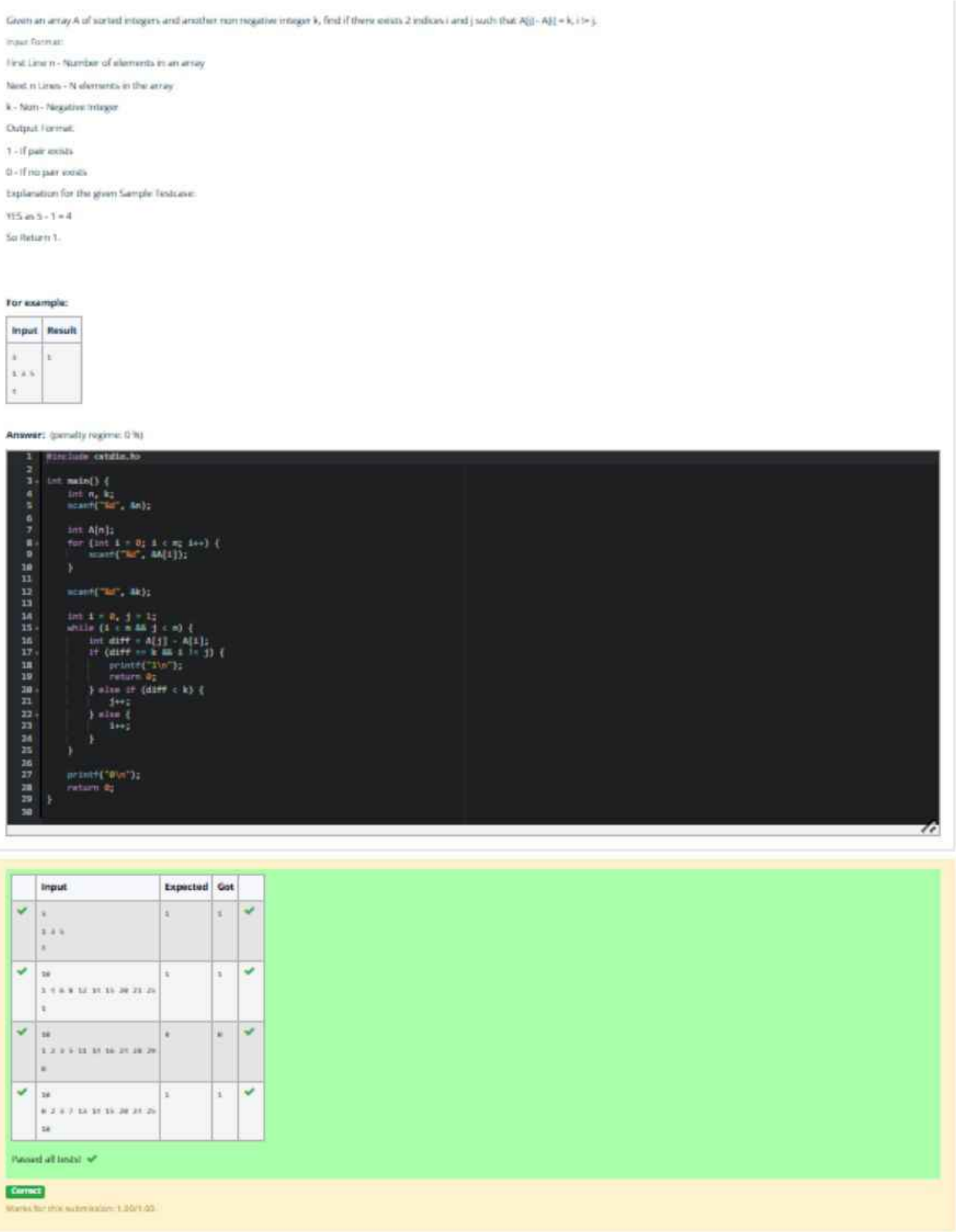
```
#include <stdio.h>

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);

    int i = 0, j = 1;
    while (i < n && j < n) {
        int diff = A[j] - A[i];
        if (diff == k && i != j) {
            printf("1\n");
            return 0;
        } else if (diff < k) {
            j++;
        } else {
            i++;
            if (i == j) {
                j++;
            }
        }
    }

    printf("0\n");
    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.



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 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 i = 0, j = 1;
    while (i < n && j < n) {
        int diff = A[j] - A[i];
        if (diff == k && i != j) {
            printf("1\n");
            return 0;
        } else if (diff < k) {
            j++;
        } else {
            i++;
        }
    }

    printf("0\n");
    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 21 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.