RITTVIK S 2024-CSE

R2

# Basic C programming-practice

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,temp;
    scanf("%d %d",&a,&b);
    temp = a;
    a = b;
    b = temp;
    printf("%d %d\n",a,b);
    return 0;
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 10 20 | 20 10 | 20 10 | ✓ |

Passed all tests! ✓

Correct

**Question 2** | Correct Mark 1.00 out of 1.00 Flag question

Write a C program to find the eligibility of admission for a professional course based on the following criteria:

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
```

**Answer:** (penalty regime: 0 %)

```c
#include <stdio.h>
int main(){
    int Maths,Physics,Chemistry;
    scanf("%d %d %d",&Maths,&Physics,&Chemistry);
    int total = Maths + Physics + Chemistry;
    if ((Maths >= 65 && Physics >= 55 && Chemistry >= 50) || total >= 180)
    printf("The candidate is eligible\n");
    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**

**Question 3** | Correct Mark 1.00 out of 1.00 Flag question

Malini goes to BestSave hyper market to buy grocery items. BestSave hyper market provides 10% discount on the bill amount B when ever the bill

The bill amount B is passed as the input to the program. The program must print the final amount A payable by Malini.

Input Format:

The first line denotes the value of B.

Output Format:

The first line contains the value of the final payable amount A.

Example Input/Output 1:

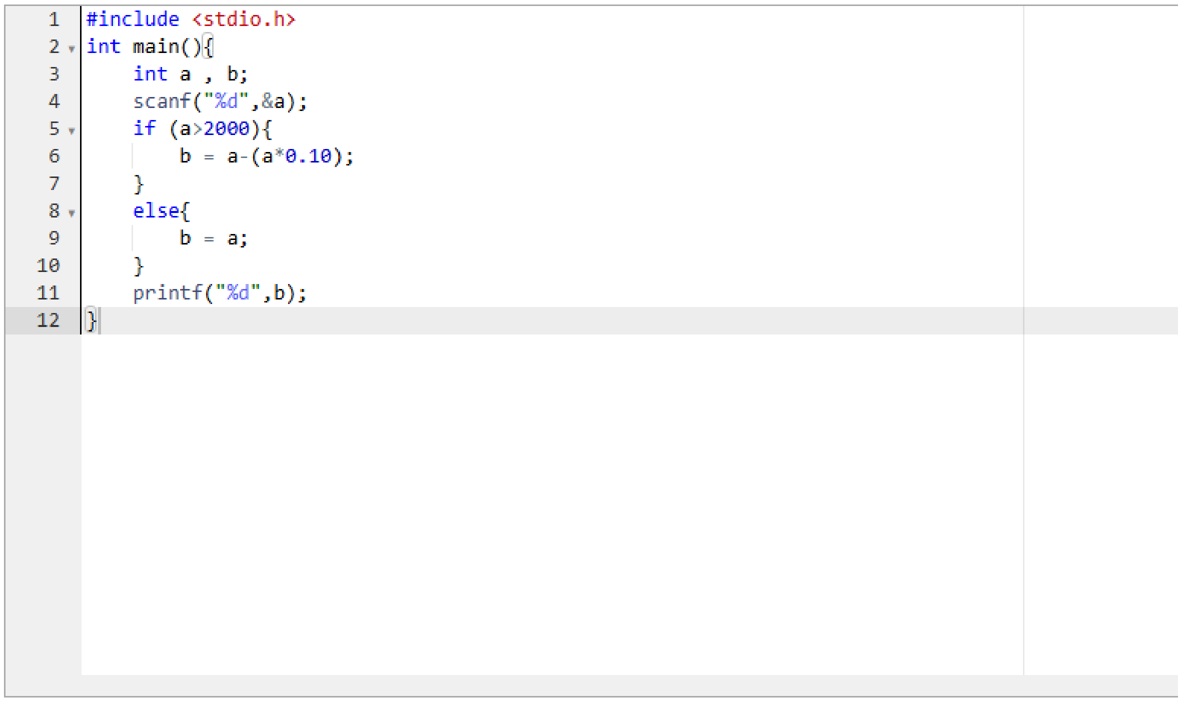
Input:

```
1900
```

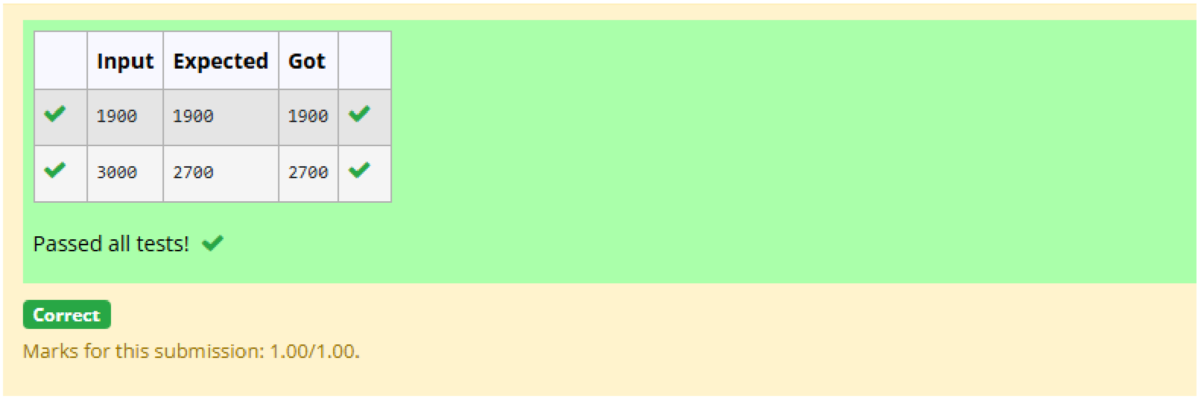
Output:

```
1900
```

**Answer:** (penalty regime: 0 %)



```
#include <stdio.h>
int main(){
    int a , b;
    scanf("%d",&a);
    if (a>2000){
        b = a-(a*0.10);
    }
    else{
        b = a;
    }
    printf("%d",b);
}
```

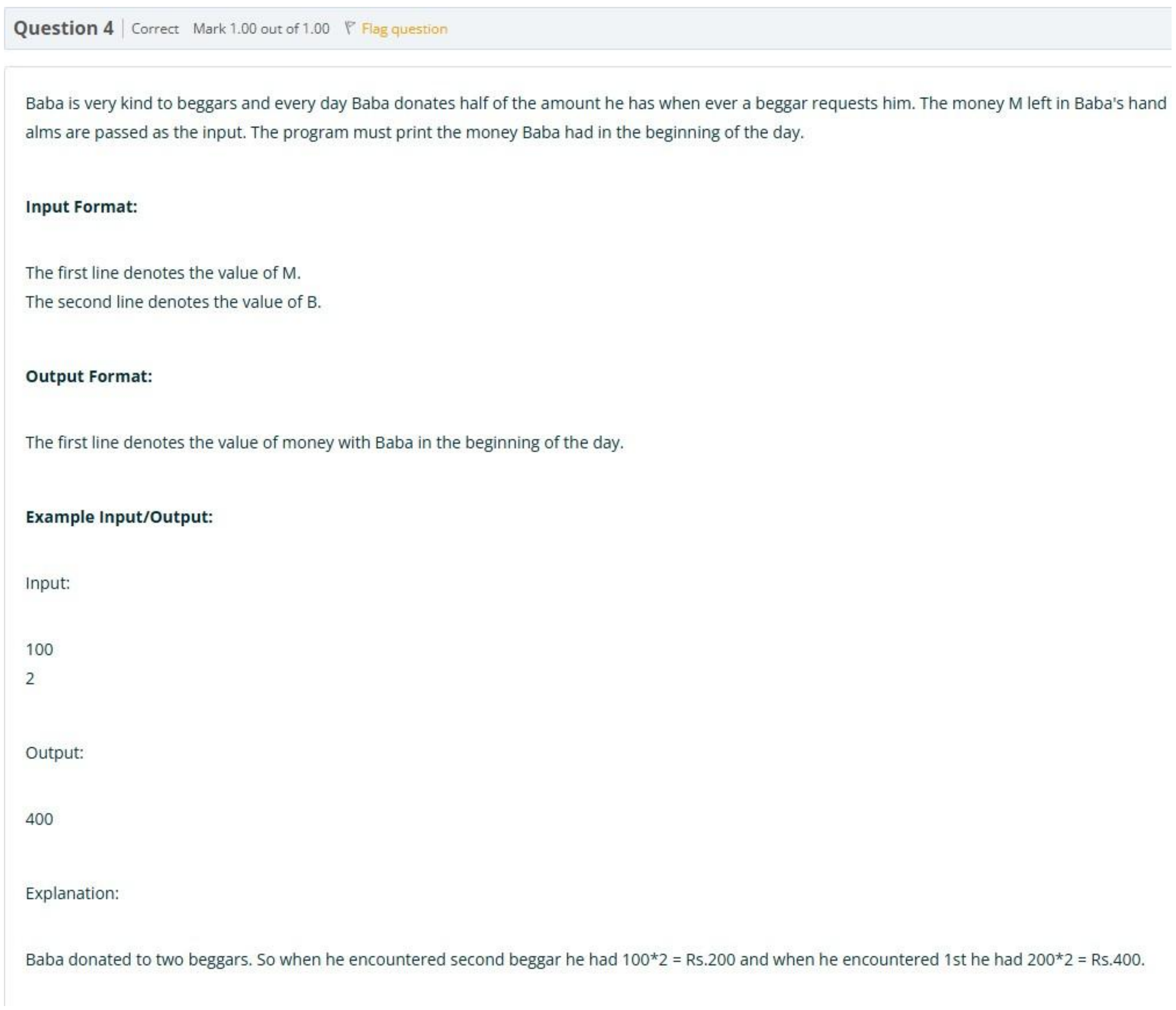


| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 1900 | 1900 | 1900 | ✓ |
| ✓ | 3000 | 2700 | 2700 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



**Question 4** | Correct Mark 1.00 out of 1.00 Flag question

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
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 %)

```c
#include <stdio.h>
#include <math.h>
int main(){
    int M,B,amt;
    scanf("%d%d",&M,&B);
    amt = M*pow(2,B);
    printf("%d",amt);
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 100<br>2 | 400 | 400 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

**Question 5** | Correct Mark 1.00 out of 1.00 Flag question

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
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 inp
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

**Answer:** (penalty regime: 0 %)

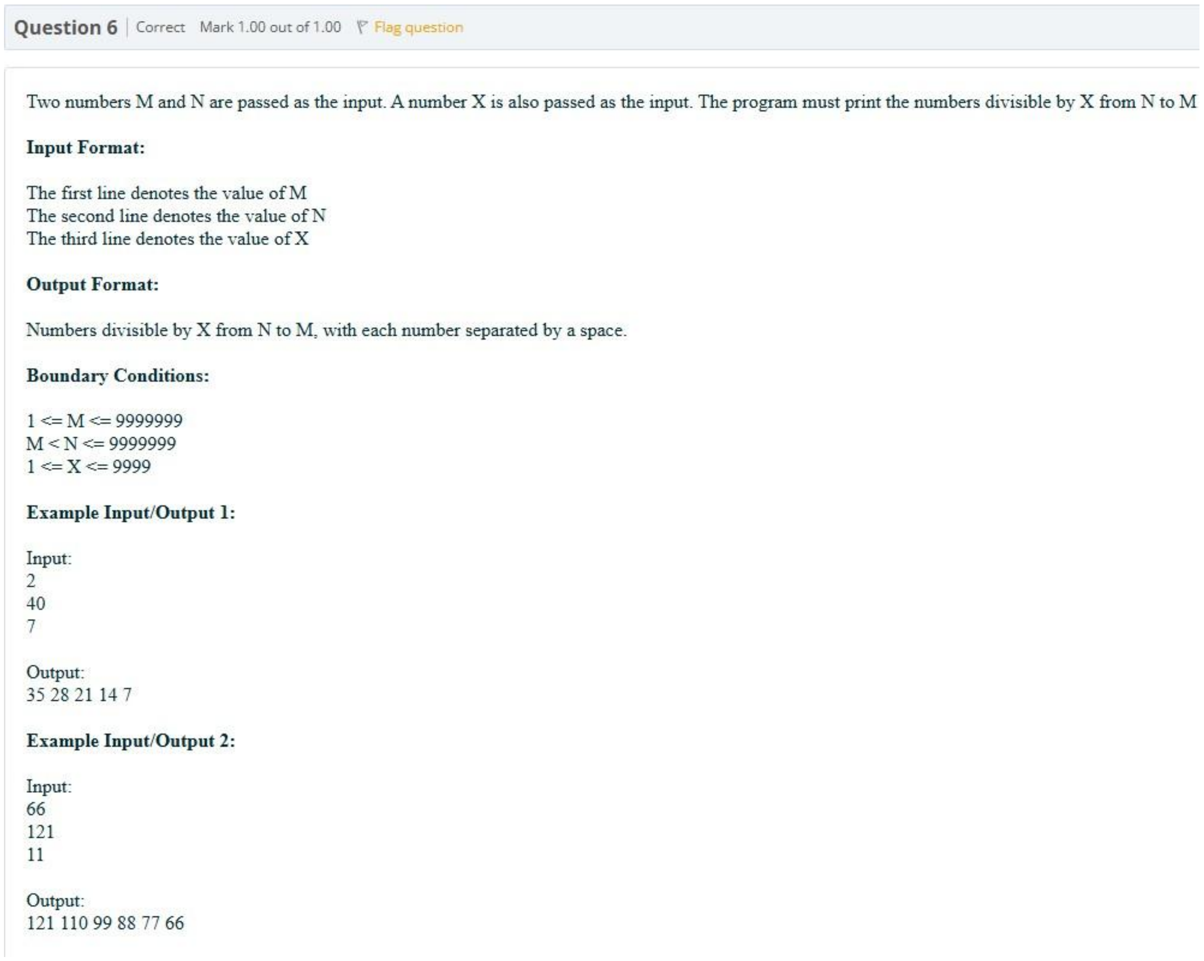
```c
#include <stdio.h>
int main(){
    int I,N,P = 0;
    scanf("%d %d",&I,&N);
    for (int i = 0;i<N;i++)
    P += I + (200*i);
    printf("%d",P);
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 500<br>3 | 2100 | 2100 | ✓ |
| ✓ | 100<br>3 | 900 | 900 | ✓ |

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.



**Question 6** | Correct Mark 1.00 out of 1.00 Flag question

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

**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:**

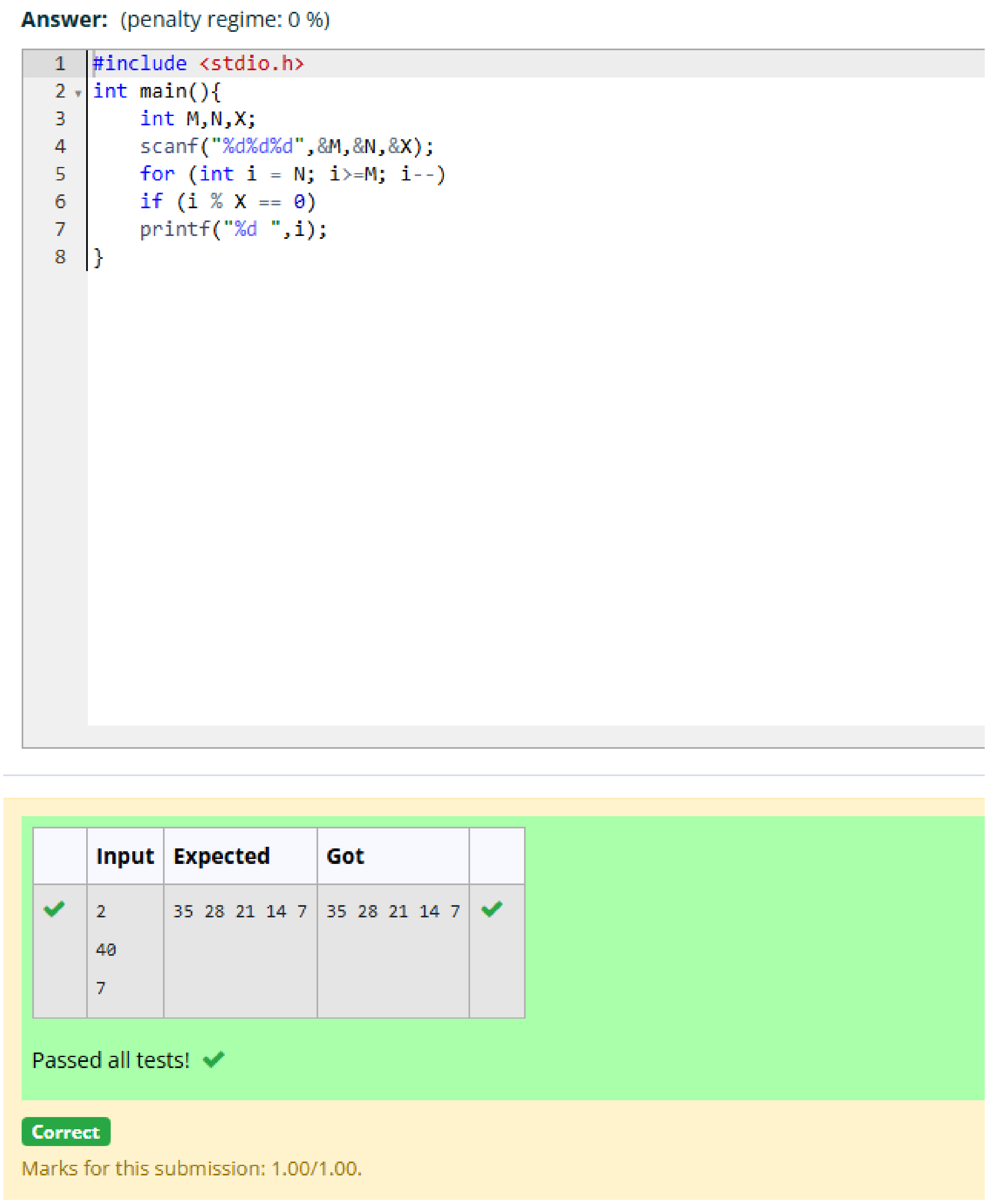
Input:
2
40
7

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 %)

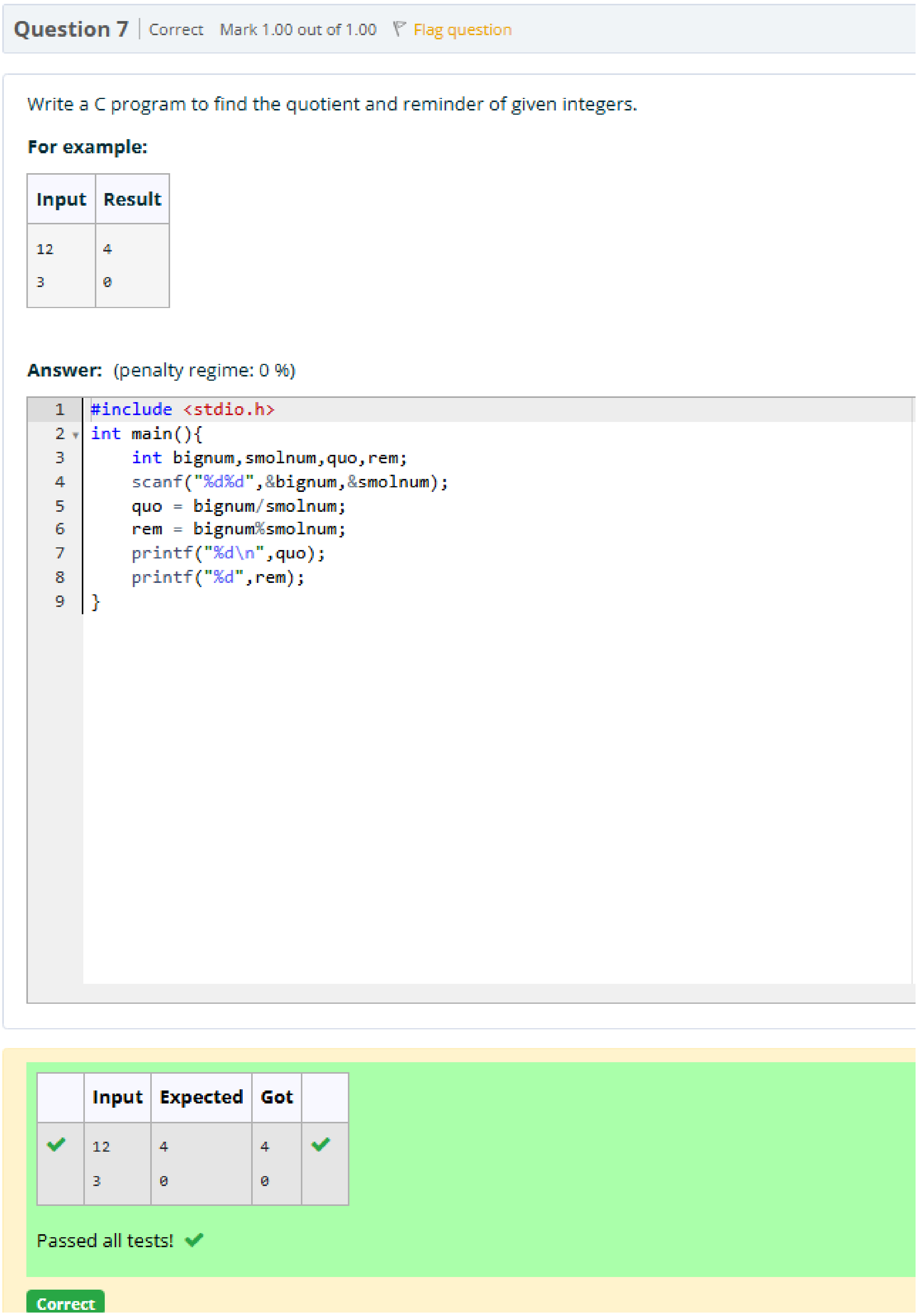
```c
#include <stdio.h>
int main(){
    int M,N,X;
    scanf("%d%d%d",&M,&N,&X);
    for (int i = N; i>=M; i--)
    if (i % X == 0)
    printf("%d ",i);
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 2<br>40<br>7 | `35 28 21 14 7` | `35 28 21 14 7` | ✓ |

Passed all tests! ✓

Correct

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<br>3 | 4<br>0 |

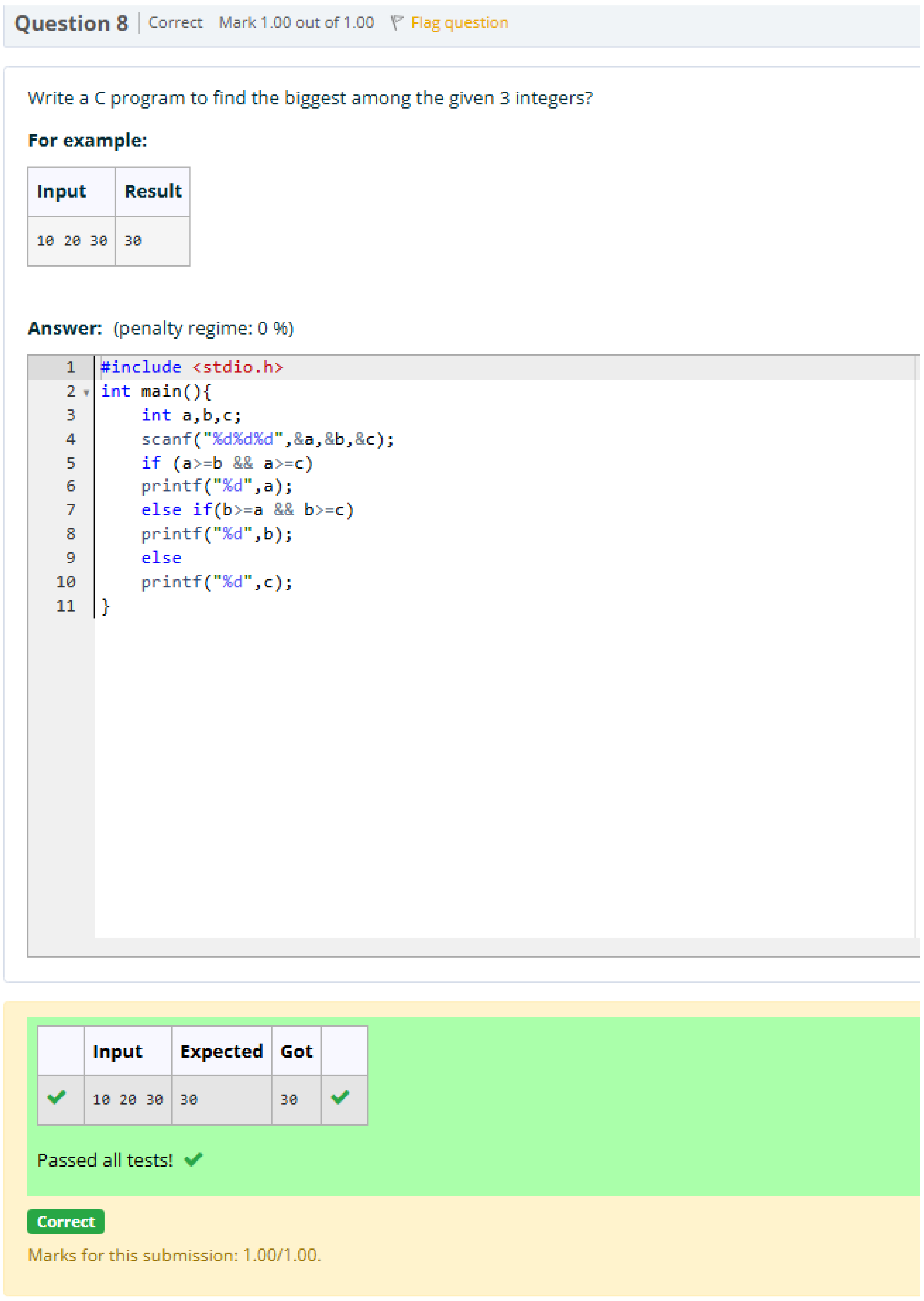
**Answer:** (penalty regime: 0 %)

```c
#include <stdio.h>
int main(){
    int bignum,smolnum,quo,rem;
    scanf("%d%d",&bignum,&smolnum);
    quo = bignum/smolnum;
    rem = bignum%smolnum;
    printf("%d\n",quo);
    printf("%d",rem);
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 12<br>3 | 4<br>0 | 4<br>0 | ✓ |

Passed all tests! ✓

Correct



**Question 8** | Correct Mark 1.00 out of 1.00 Flag question

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 %)

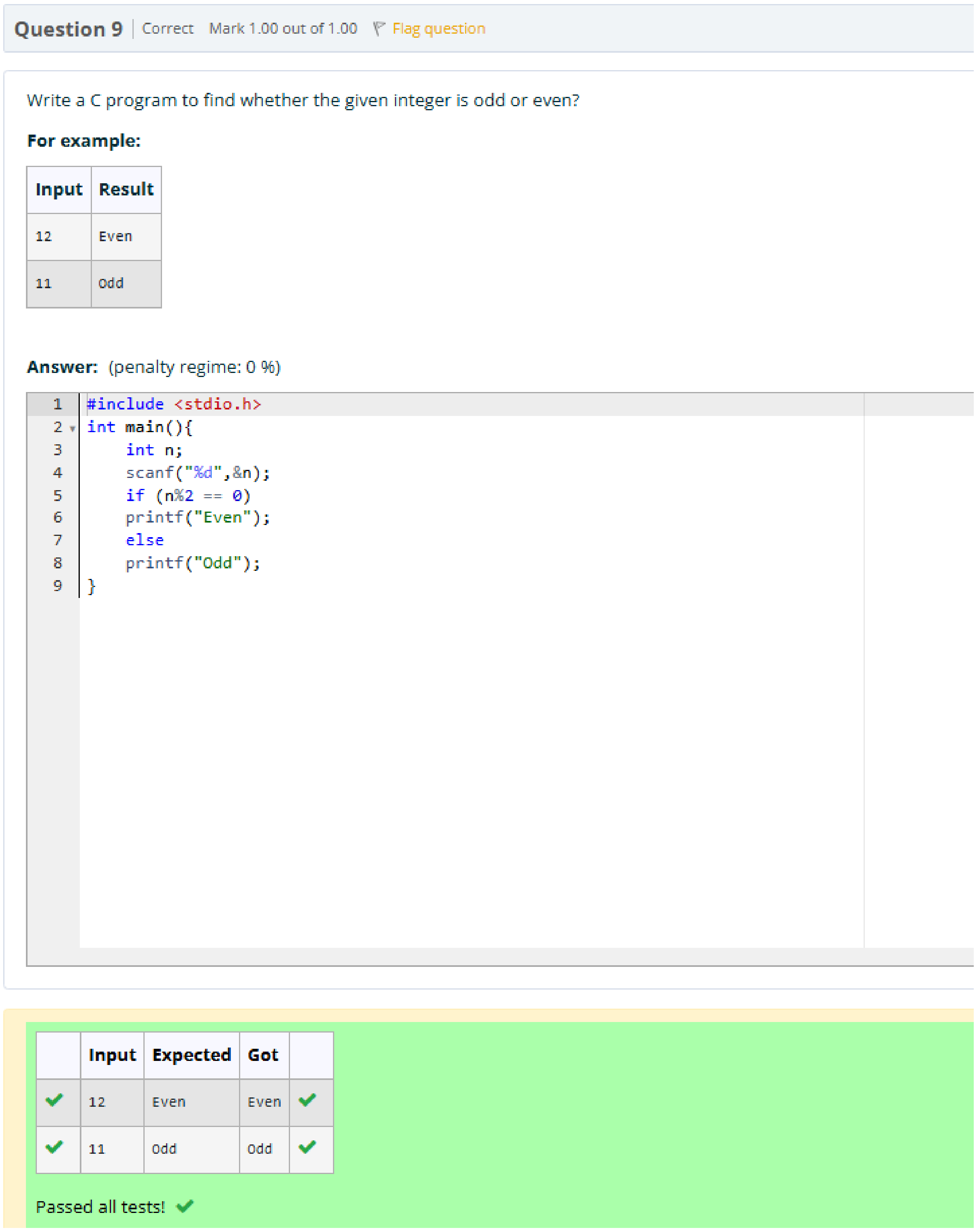
```c
#include <stdio.h>
int main(){
    int a,b,c;
    scanf("%d%d%d",&a,&b,&c);
    if (a>=b && a>=c)
    printf("%d",a);
    else if(b>=a && b>=c)
    printf("%d",b);
    else
    printf("%d",c);
}
```

|  | Input | Expected | Got |  |
|---|---|---|---|---|
| ✓ | 10 20 30 | 30 | 30 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



**Question 9** | Correct Mark 1.00 out of 1.00 Flag question

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 n;
    scanf("%d",&n);
    if (n%2 == 0)
    printf("Even");
    else
    printf("Odd");
}
```

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

Passed all tests! ✓

## Question 10

Correct Mark 1.00 out of 1.00 Flag question

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

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 5 | 120 | 120 | ✓ |

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.

## Question 11 | Correct Mark 1.00 out of 1.00 Flag question

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

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 3 | 6 | 6 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

**Question 12** | Correct Mark 1.00 out of 1.00 Flag question

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 %)

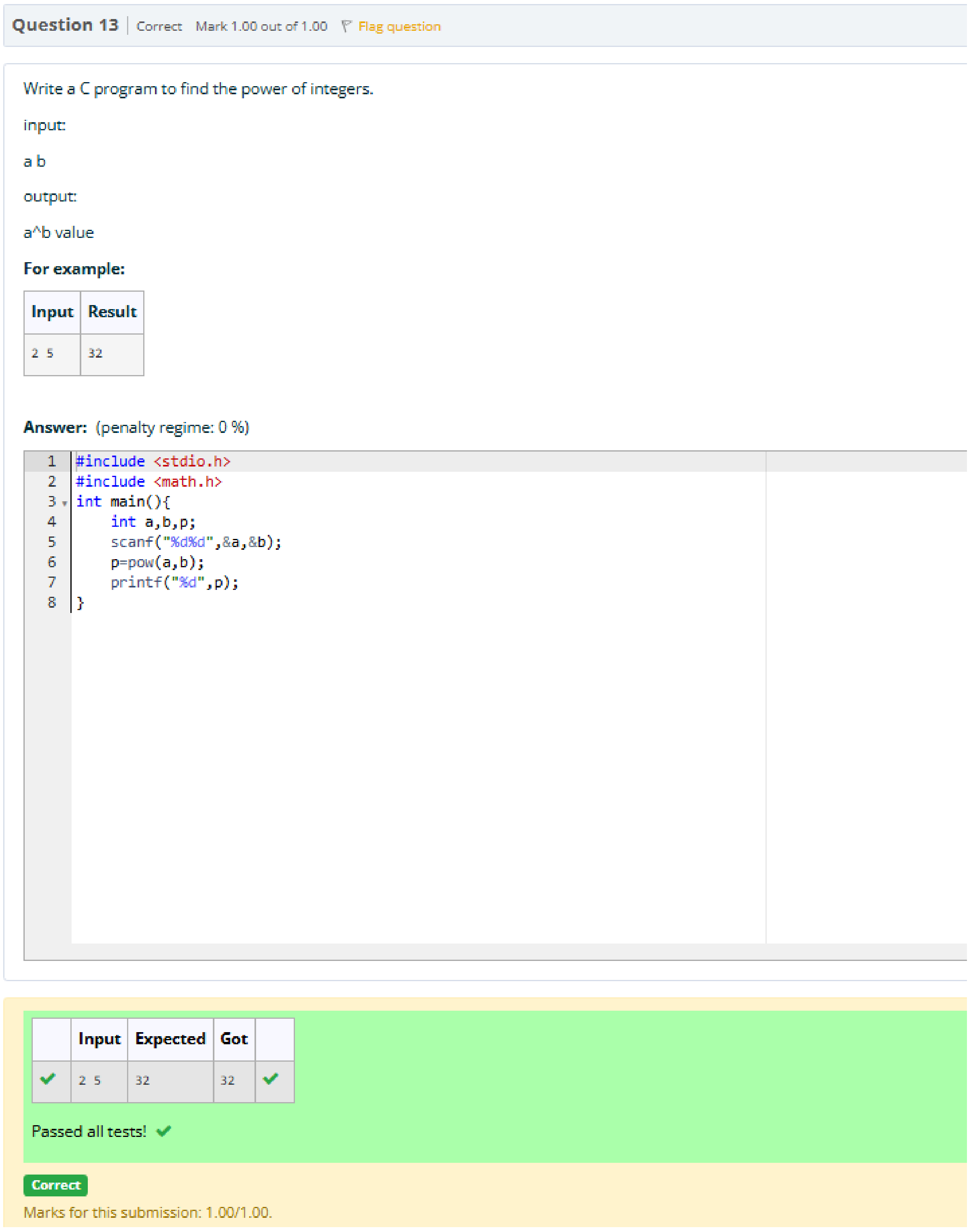
```
#include <stdio.h>
int main(){
    int n,i,t;
    scanf("%d",&n);
    int f = 0, s = 1;
    for (i = 0;i<=n;i++){
    t = f + s;
    if (i==n)
    printf("%d ",f);
    f = s;
    s = t;
    }
}
```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



**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 %)

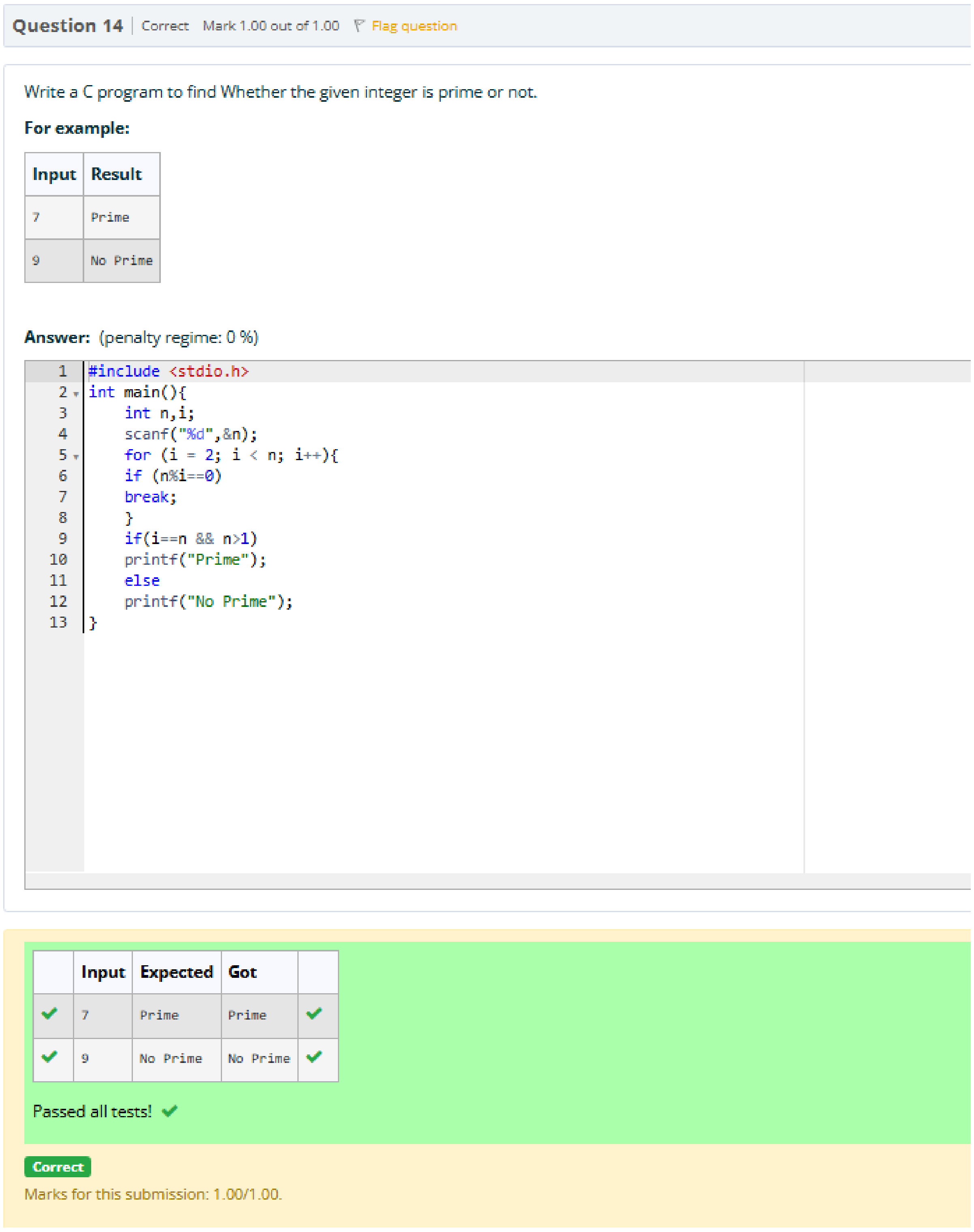
```c
#include <stdio.h>
#include <math.h>
int main(){
    int a,b,p;
    scanf("%d%d",&a,&b);
    p=pow(a,b);
    printf("%d",p);
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 2 5 | 32 | 32 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



**Question 14** | Correct Mark 1.00 out of 1.00 Flag question

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 %)

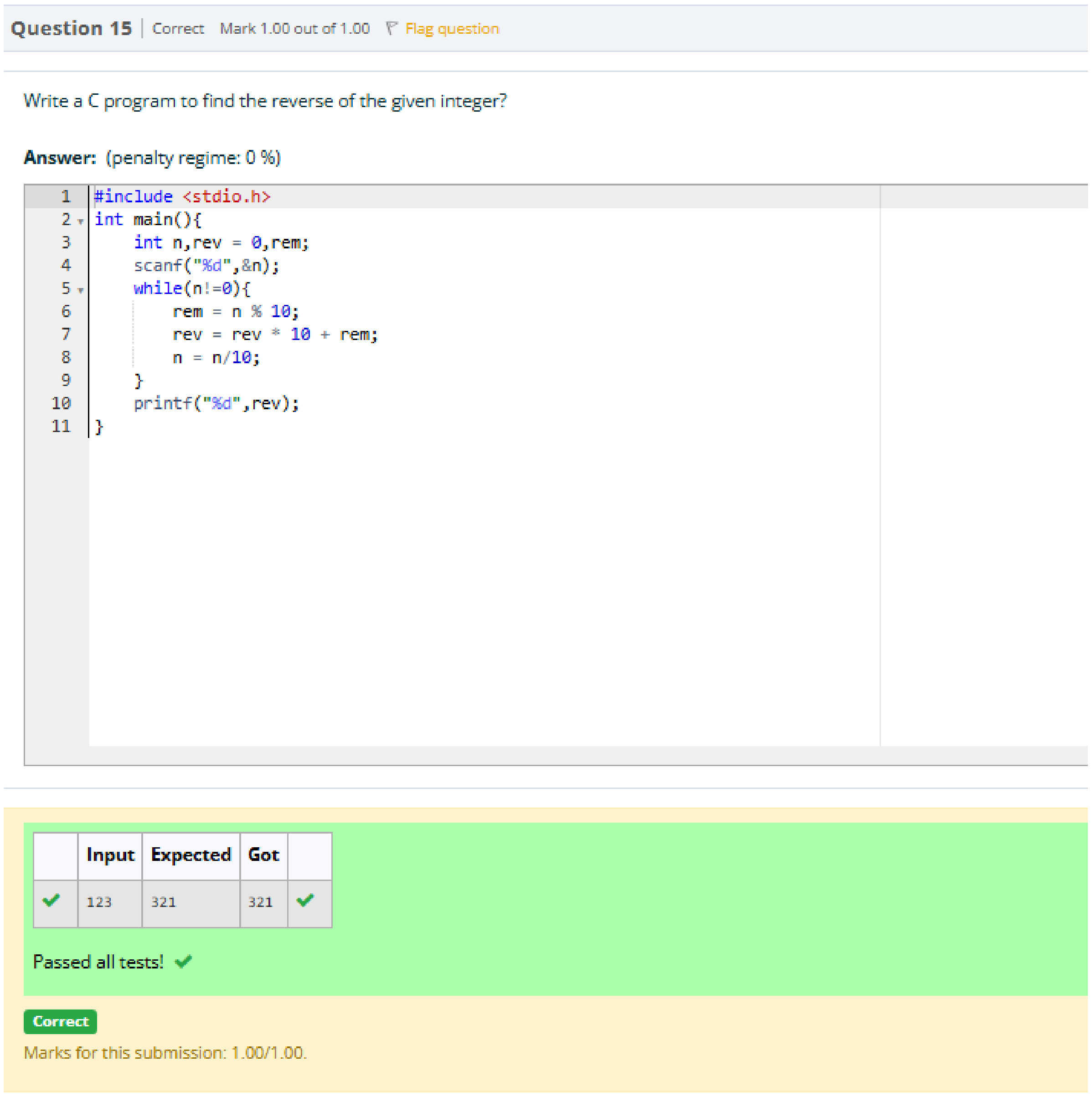
```c
#include <stdio.h>
int main(){
    int n,i;
    scanf("%d",&n);
    for (i = 2; i < n; i++){
    if (n%i==0)
    break;
    }
    if(i==n && n>1)
    printf("Prime");
    else
    printf("No Prime");
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 7 | Prime | Prime | ✓ |
| ✓ | 9 | No Prime | No Prime | ✓ |

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.


**Question 15** | Correct Mark 1.00 out of 1.00 Flag question

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

**Answer:** (penalty regime: 0 %)

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

|  | Input | Expected | Got |  |
|---|---|---|---|---|
| ✓ | 123 | 321 | 321 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

RITTVIK S 2024-CSE R2

## Finding time complexity of algorithms

## 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:**

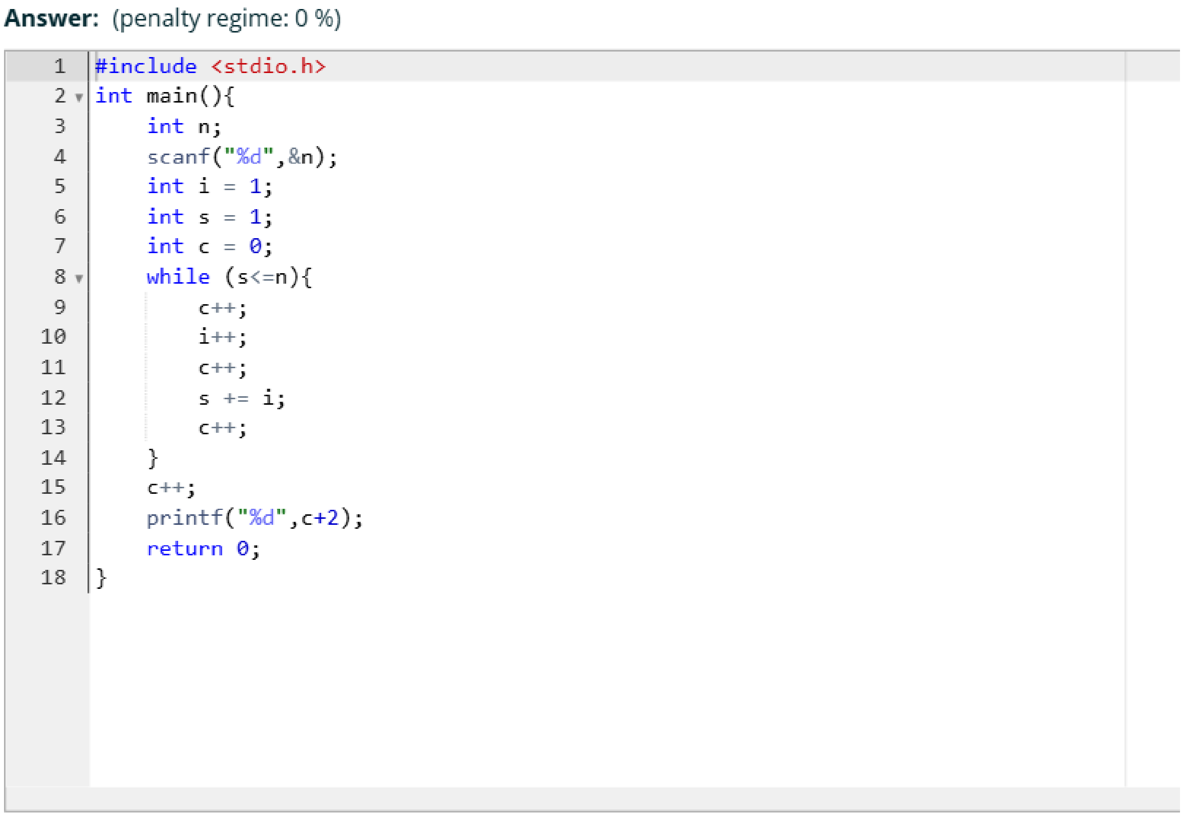
A positive Integer n

**Output:**

Print the value of the counter variable

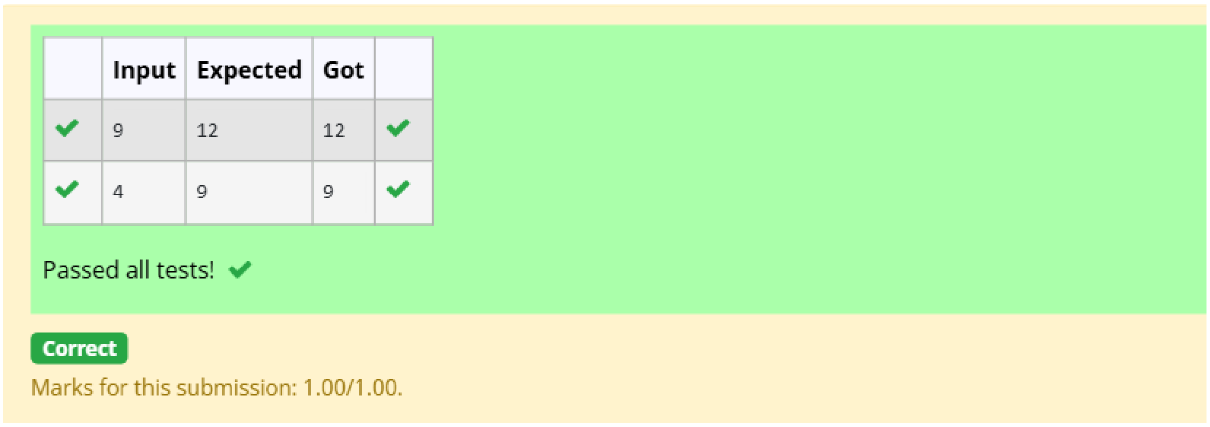
**For example:**

| Input | Result |
|---|---|
| 9 | 12 |



**Answer:** (penalty regime: 0 %)

```c
#include <stdio.h>
int main(){
    int n;
    scanf("%d",&n);
    int i = 1;
    int s = 1;
    int c = 0;
    while (s<=n){
        c++;
        i++;
        c++;
        s += i;
        c++;
    }
    c++;
    printf("%d",c+2);
    return 0;
}
```



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 9 | 12 | 12 | ✓ |
| ✓ | 4 | 9 | 9 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

## Problem 2: Finding Complexity using Counter method

## 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 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;
       }
     }
   }
 }
```

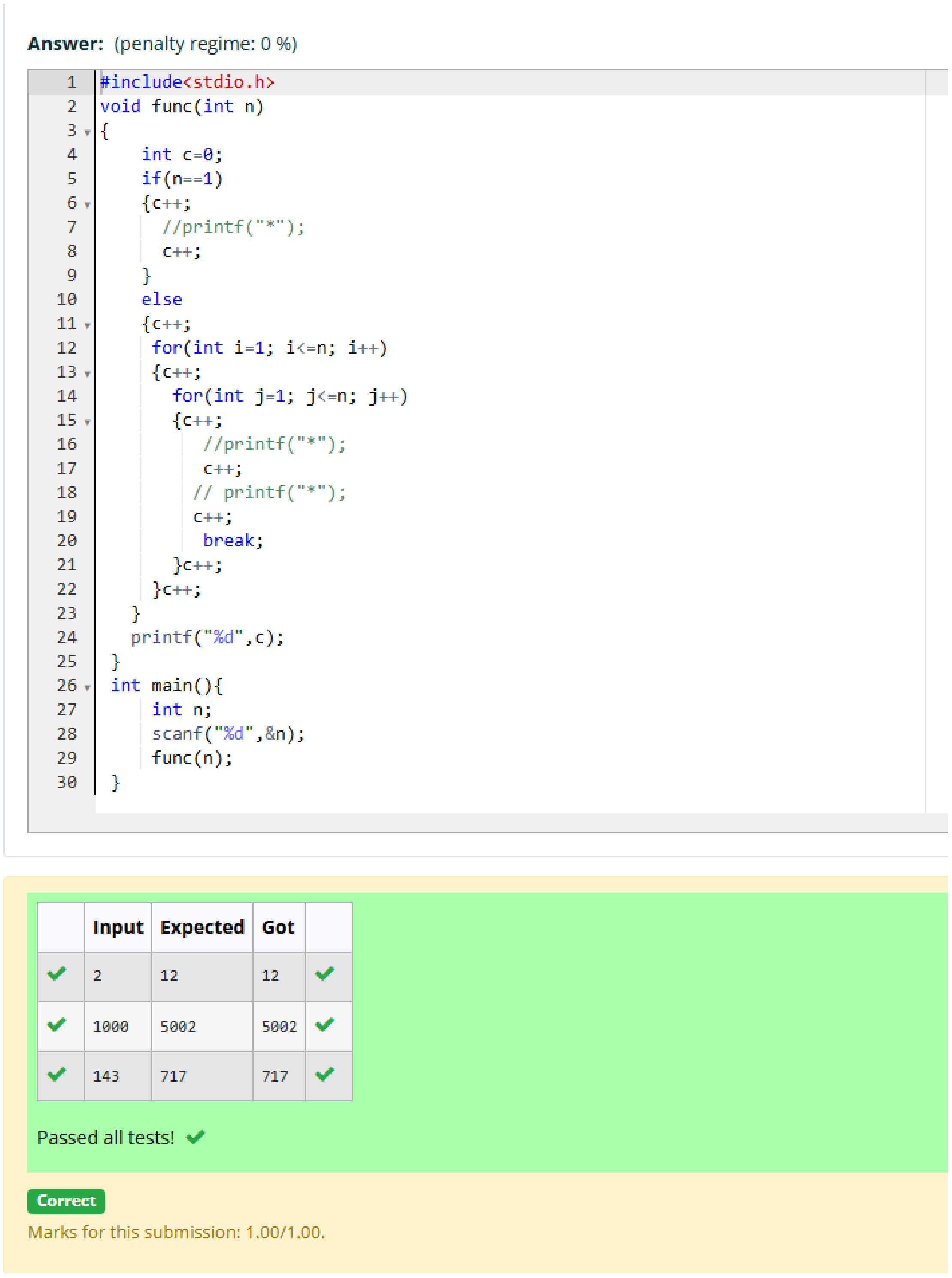
**Note:** No need of counter increment for declarations and scanf() and count variable printf() statements.

**Input:**

A positive Integer n

**Output:**

Print the value of the counter variable



**Answer:** (penalty regime: 0 %)

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

| | 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 3: Finding Complexity using Counter Method

**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 counter method.

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

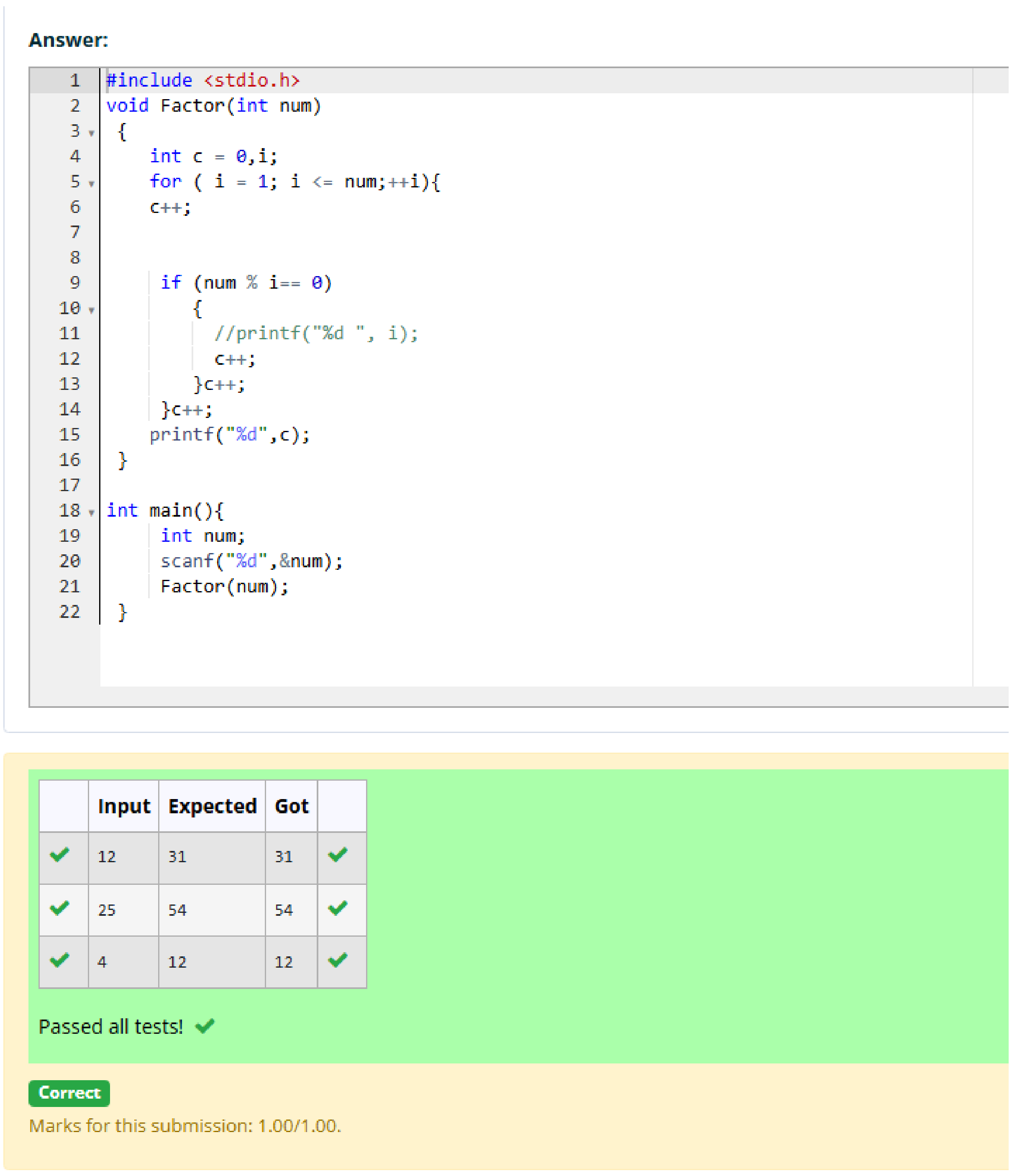
**Note:** No need of counter increment for declarations and scanf() and counter variable printf() statement.

**Input:**

A positive Integer n

**Output:**

Print the value of the counter variable



**Answer:**

```c
#include <stdio.h>
void Factor(int num)
 {
    int c = 0,i;
    for ( i = 1; i <= num;++i){
    c++;


      if (num % i== 0)
        {
          //printf("%d ", i);
          c++;
        }c++;
      }c++;
    printf("%d",c);
 }

int main(){
      int num;
      scanf("%d",&num);
      Factor(num);
 }
```

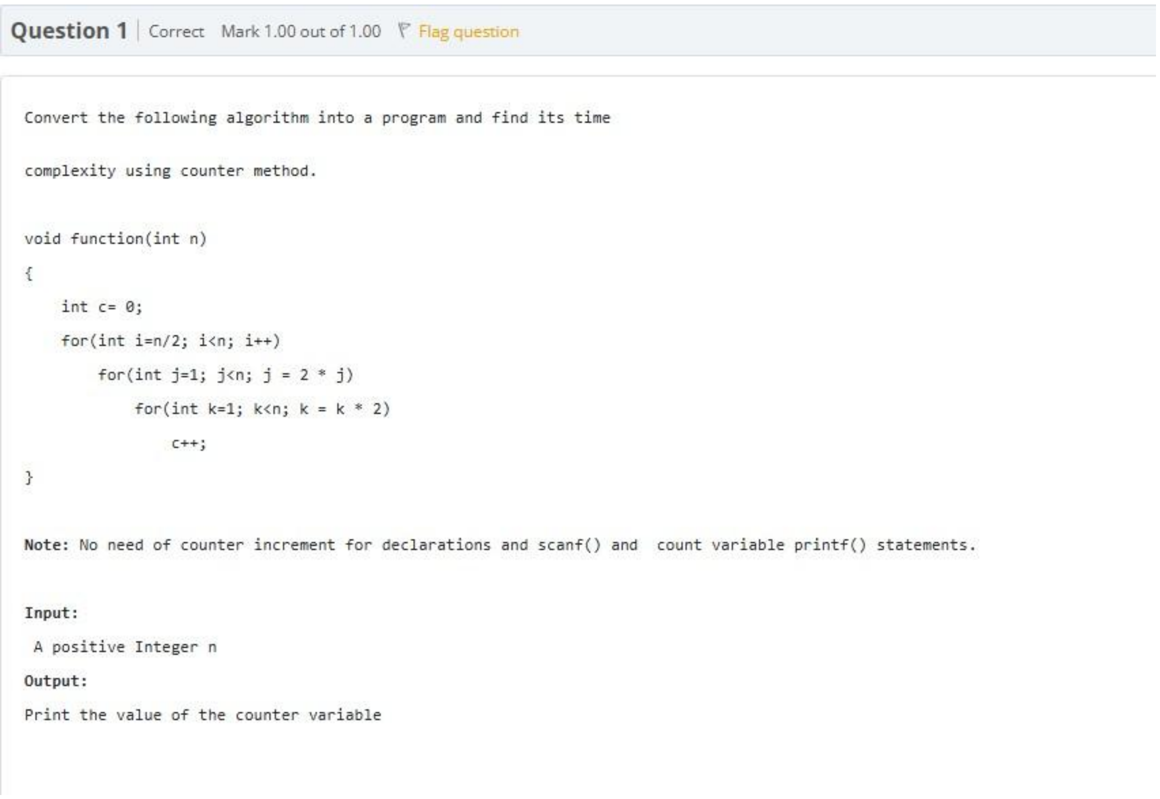
| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 12 | 31 | 31 | ✓ |
| ✓ | 25 | 54 | 54 | ✓ |
| ✓ | 4 | 12 | 12 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

**Problem 4: Finding Complexity using Counter Method**



**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 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++;
}
```

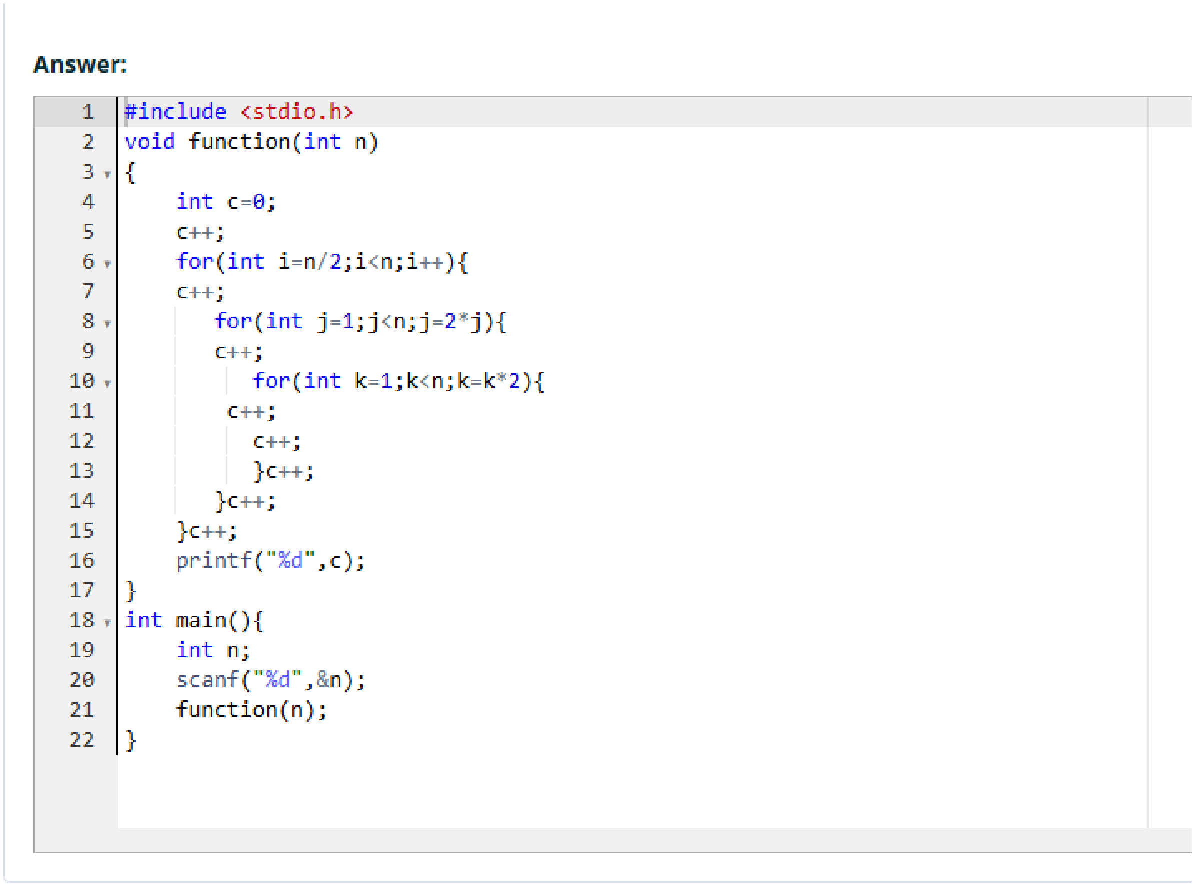
**Note:** No need of counter increment for declarations and scanf() and  count variable printf() statements.

**Input:**

A positive Integer n

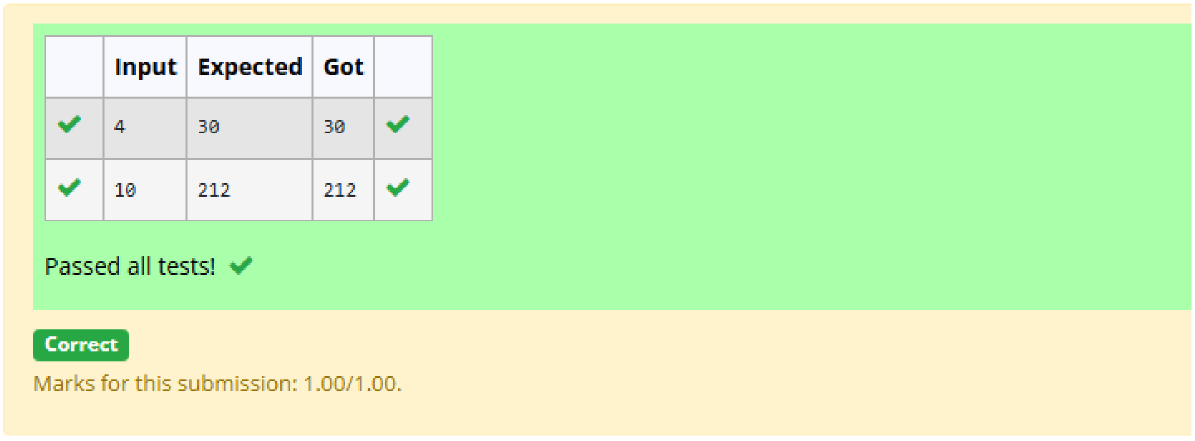
**Output:**

Print the value of the counter variable



**Answer:**

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



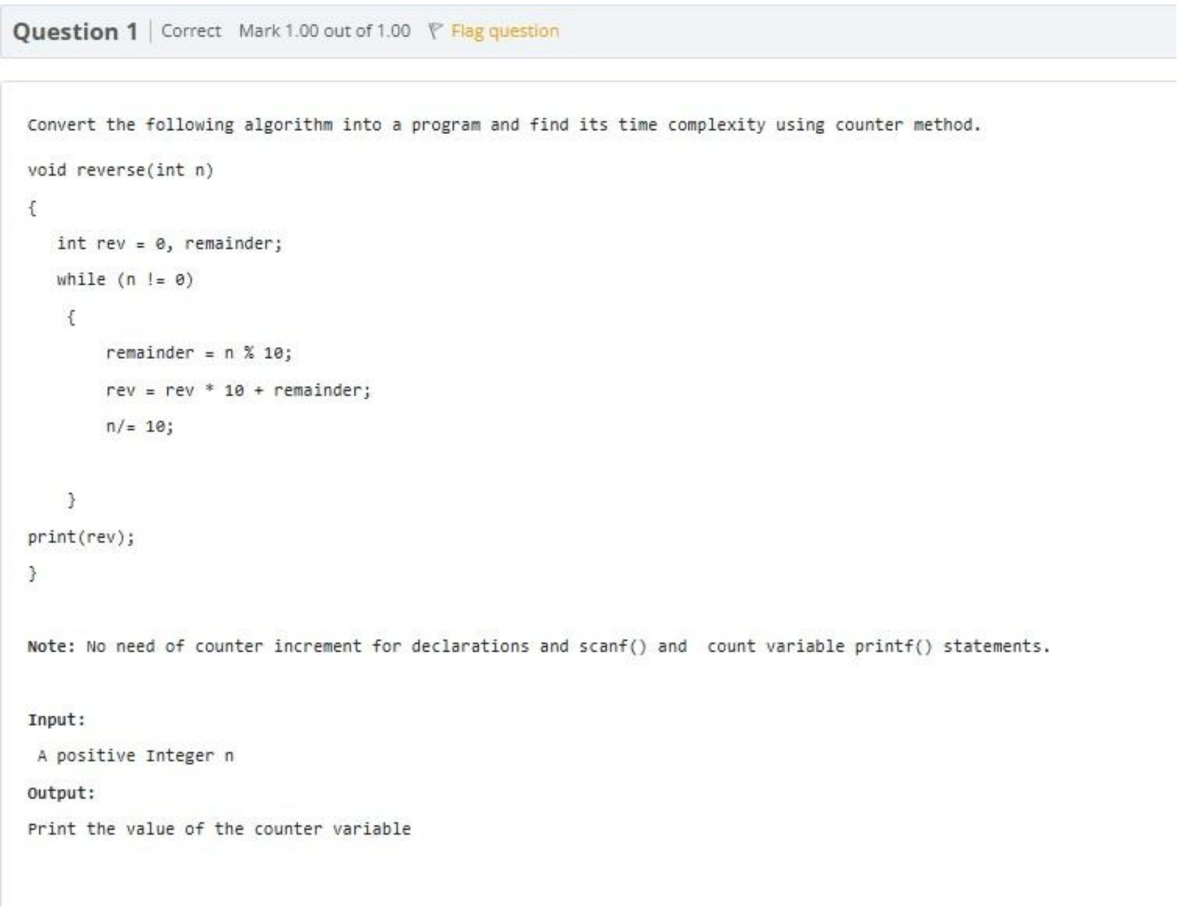
| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 4 | 30 | 30 | ✓ |
| ✓ | 10 | 212 | 212 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

## Problem 5: Finding Complexity using counter method



**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 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>
void reverse(int n)
{  int c=0;
   int rev = 0, remainder;
   c++;
   while (n != 0)
    {c++;
        remainder = n % 10;
        c++;
        rev = rev * 10 + remainder;
        c++;
        n/= 10;
        c++;
    }
    c++;
//printf(rev);
c++;
printf("%d",c);
}
int main(){
    int n;
    scanf("%d",&n);
    reverse(n);
}
```

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

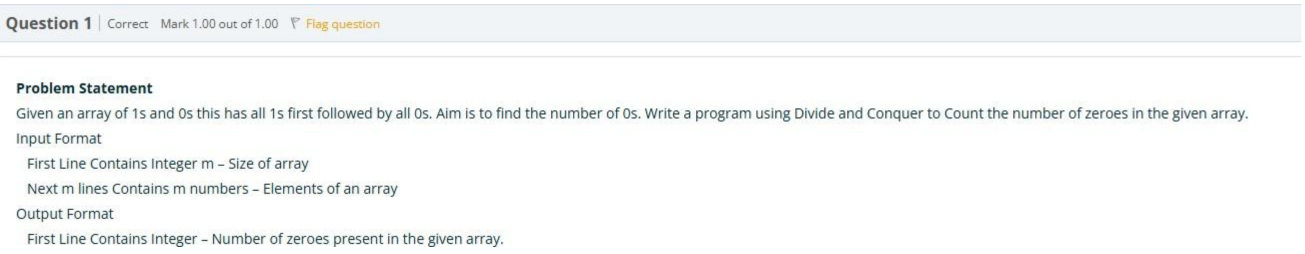
Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

RITTVIK S 2024-CSE R2

**Divide and Conquer**



**Question 1** | Correct Mark 1.00 out of 1.00 Flag question

**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

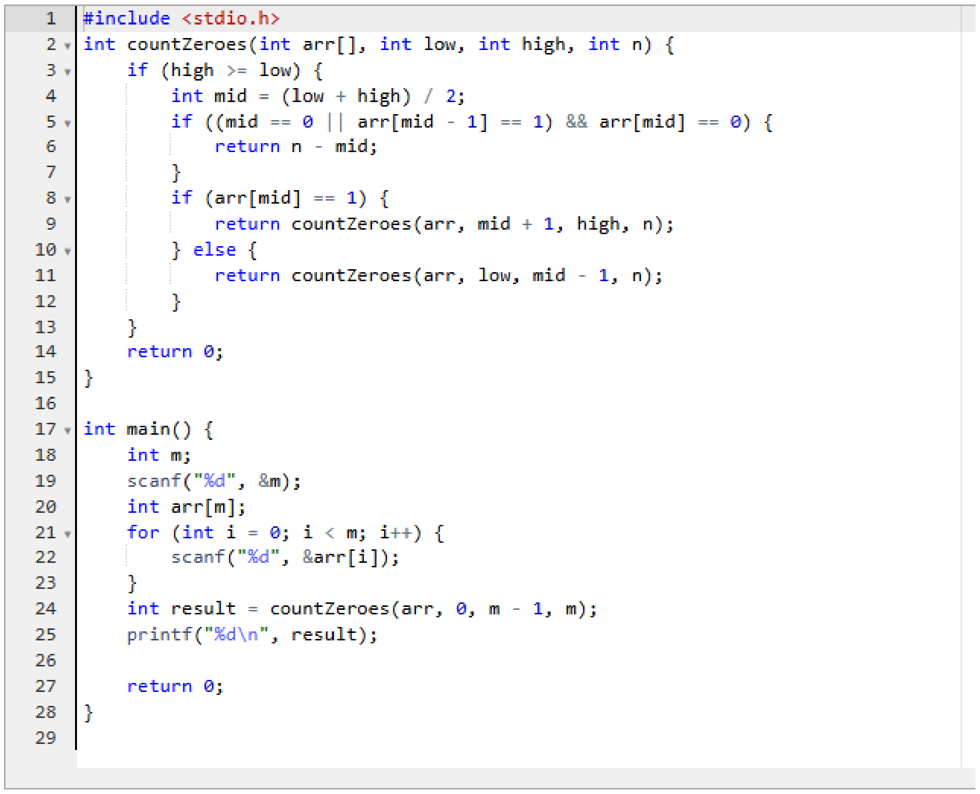
Next m lines Contains m numbers – Elements of an array

Output Format

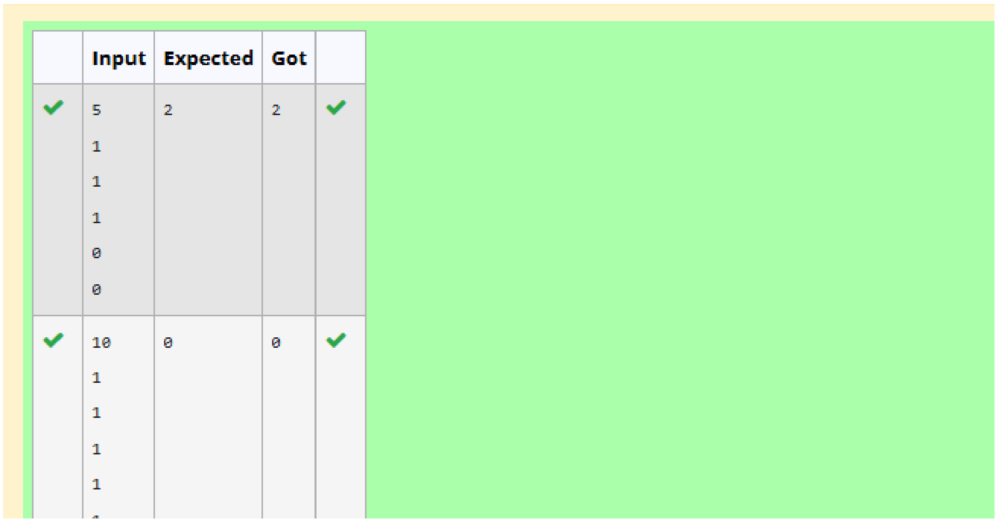
First Line Contains Integer – Number of zeroes present in the given array.

## 1-Number of Zeros in a Given Array

**Answer:** (penalty regime: 0 %)



```c
#include <stdio.h>
int countZeroes(int arr[], int low, int high, int n) {
    if (high >= low) {
        int mid = (low + high) / 2;
        if ((mid == 0 || arr[mid - 1] == 1) && arr[mid] == 0) {
            return n - mid;
        }
        if (arr[mid] == 1) {
            return countZeroes(arr, mid + 1, high, n);
        } else {
            return countZeroes(arr, low, mid - 1, n);
        }
    }
    return 0;
}

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<br>1<br>1<br>1<br>0<br>0 | 2 | 2 | ✓ |
| ✓ | 10<br>1<br>1<br>1<br>1 | 0 | 0 | ✓ |

## 2-Majority Element

**Question 1** | Correct Mark 1.00 out of 1.00 Flag question

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<br>3 2 3 | 3 |
| 7<br>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;
    int candidate = 0;

    for (int i = 0; i < n; i++) {
        if (count == 0) {
            candidate = nums[i];
        }
        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;
}
```

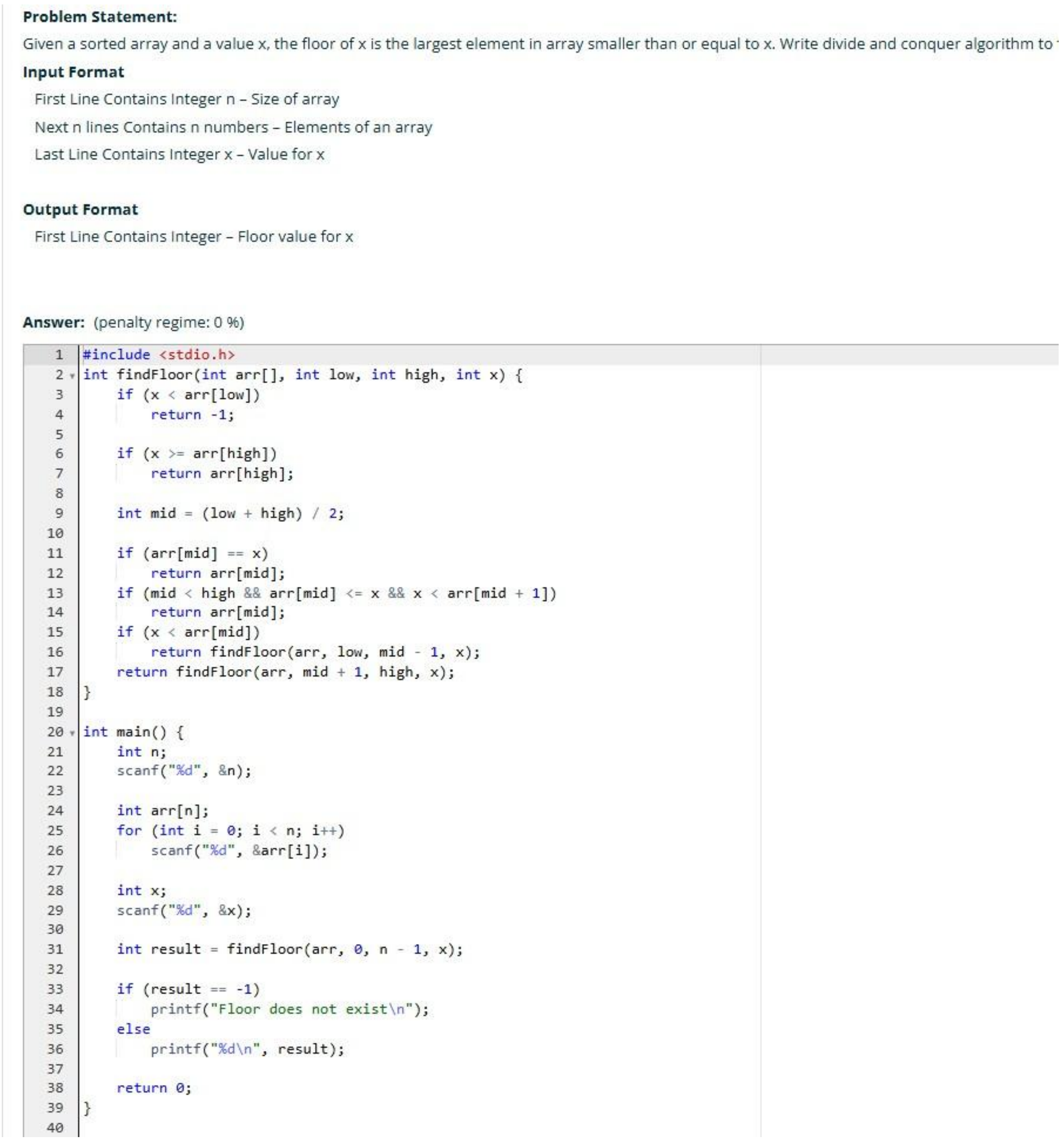
| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 3<br>3 2 3 | 3 | 3 | ✓ |

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.

## 3-Finding Floor Value



**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

**Input Format**

First Line Contains Integer n – Size of array

Next n lines Contains n numbers – Elements of an array

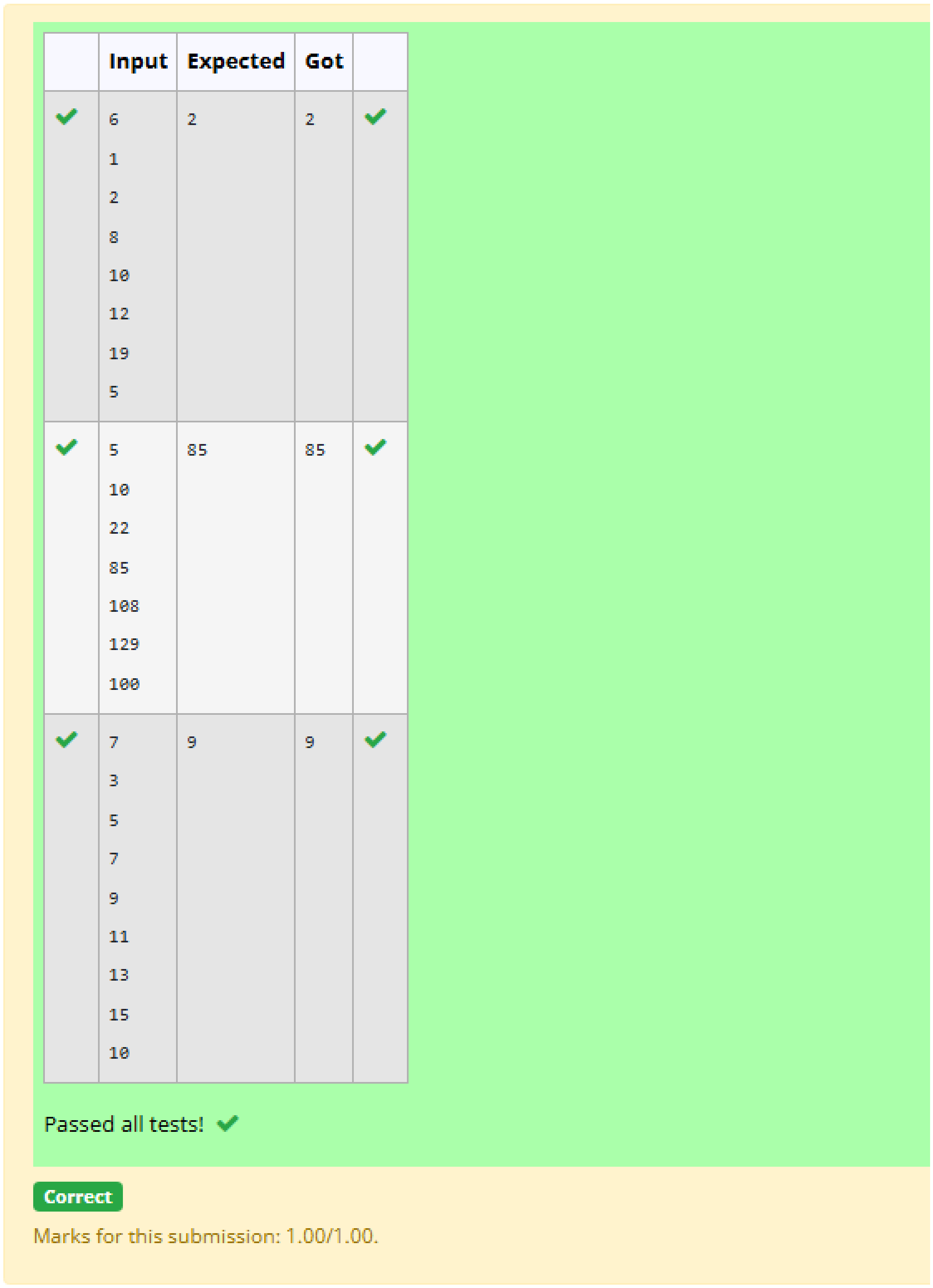
Last Line Contains Integer x – Value for x

**Output Format**

First Line Contains Integer – Floor value for x

**Answer:** (penalty regime: 0 %)

```c
#include <stdio.h>
int findFloor(int arr[], int low, int high, int x) {
    if (x < arr[low])
        return -1;

    if (x >= arr[high])
        return arr[high];

    int mid = (low + high) / 2;

    if (arr[mid] == x)
        return arr[mid];
    if (mid < high && arr[mid] <= x && x < arr[mid + 1])
        return arr[mid];
    if (x < arr[mid])
        return findFloor(arr, low, mid - 1, x);
    return findFloor(arr, mid + 1, high, 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);

    int result = findFloor(arr, 0, n - 1, x);

    if (result == -1)
        printf("Floor does not exist\n");
    else
        printf("%d\n", result);

    return 0;
}
```



|  | Input | Expected | Got |  |
|---|---|---|---|---|
| ✓ | 6<br>1<br>2<br>8<br>10<br>12<br>19<br>5 | 2 | 2 | ✓ |
| ✓ | 5<br>10<br>22<br>85<br>108<br>129<br>100 | 85 | 85 | ✓ |
| ✓ | 7<br>3<br>5<br>7<br>9<br>11<br>13<br>15<br>10 | 9 | 9 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

## 4-Two Elements sum to x

**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 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

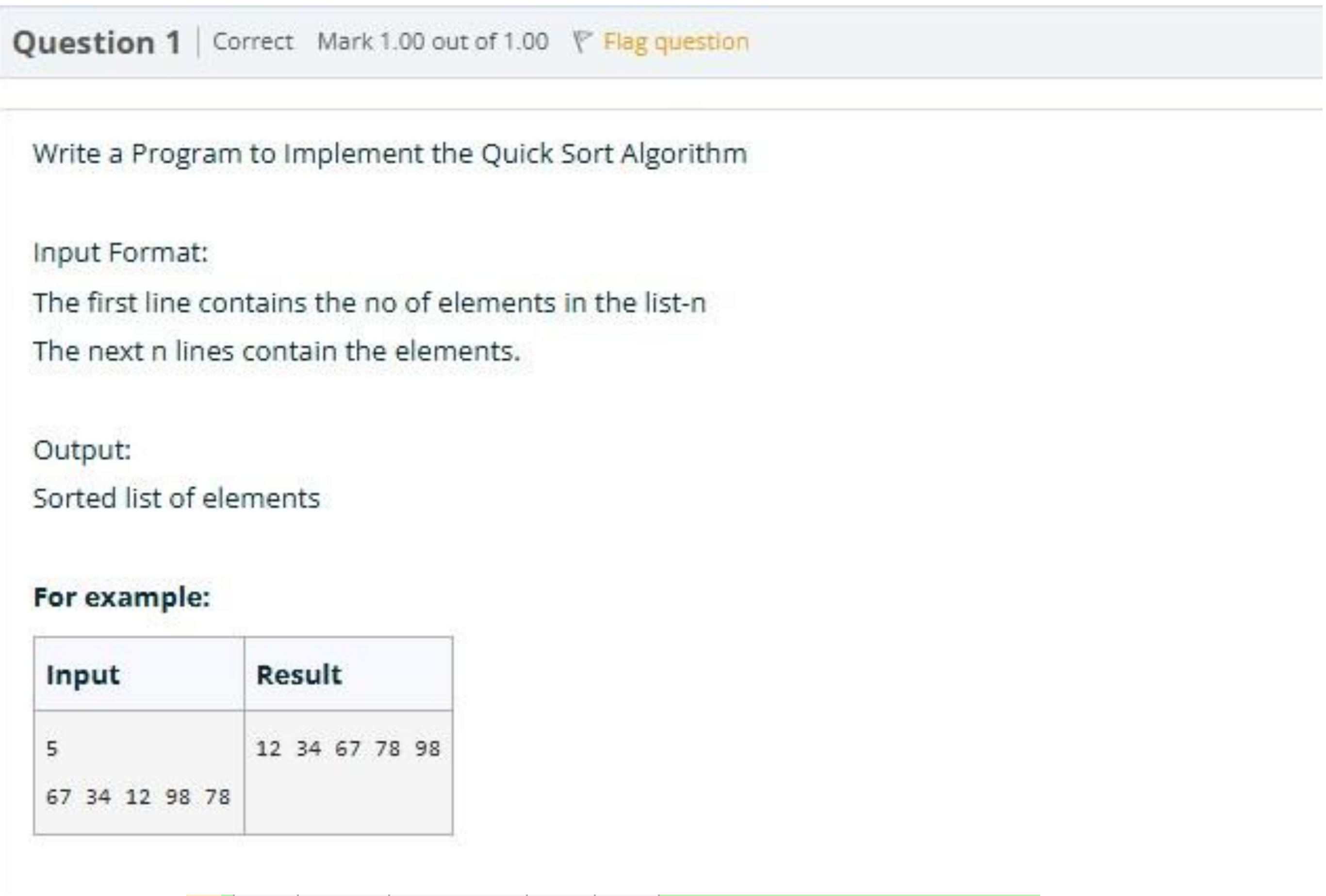
**Output Format**

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>

int findPair(int arr[], int left, int right, int x, int *a, int *b) {
    if (left >= right)
        return 0;

    int sum = arr[left] + arr[right];

    if (sum == x) {
        *a = arr[left];
        *b = arr[right];
        return 1;
    }

    if (sum > x)
        return findPair(arr, left, right - 1, x, a, b);
    else
        return findPair(arr, left + 1, right, x, a, b);
}

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

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

    scanf("%d", &x);

    int a, b;
    if (findPair(arr, 0, n - 1, x, &a, &b)) {
        printf("%d\n%d\n", a, b);
    } else {
        printf("No\n");
    }

    return 0;
```


Question 1 | Correct Mark 1.00 out of 1.00 Flag question

Write a Program to Implement the Quick Sort Algorithm

Input Format:

The first line contains the no of elements in the list-n

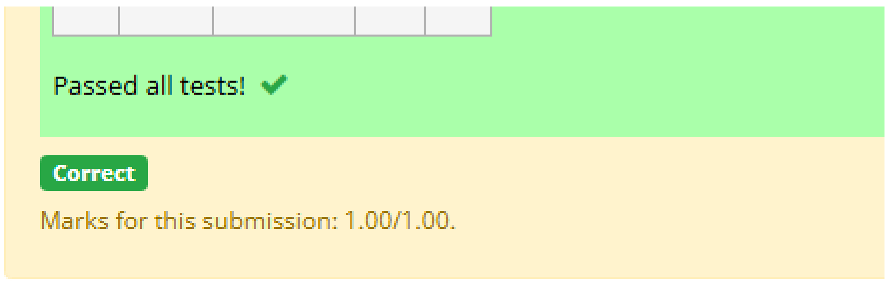
The next n lines contain the elements.

Output:

Sorted list of elements

**For example:**

| Input | Result |
|---|---|
| 5<br>67 34 12 98 78 | 12 34 67 78 98 |

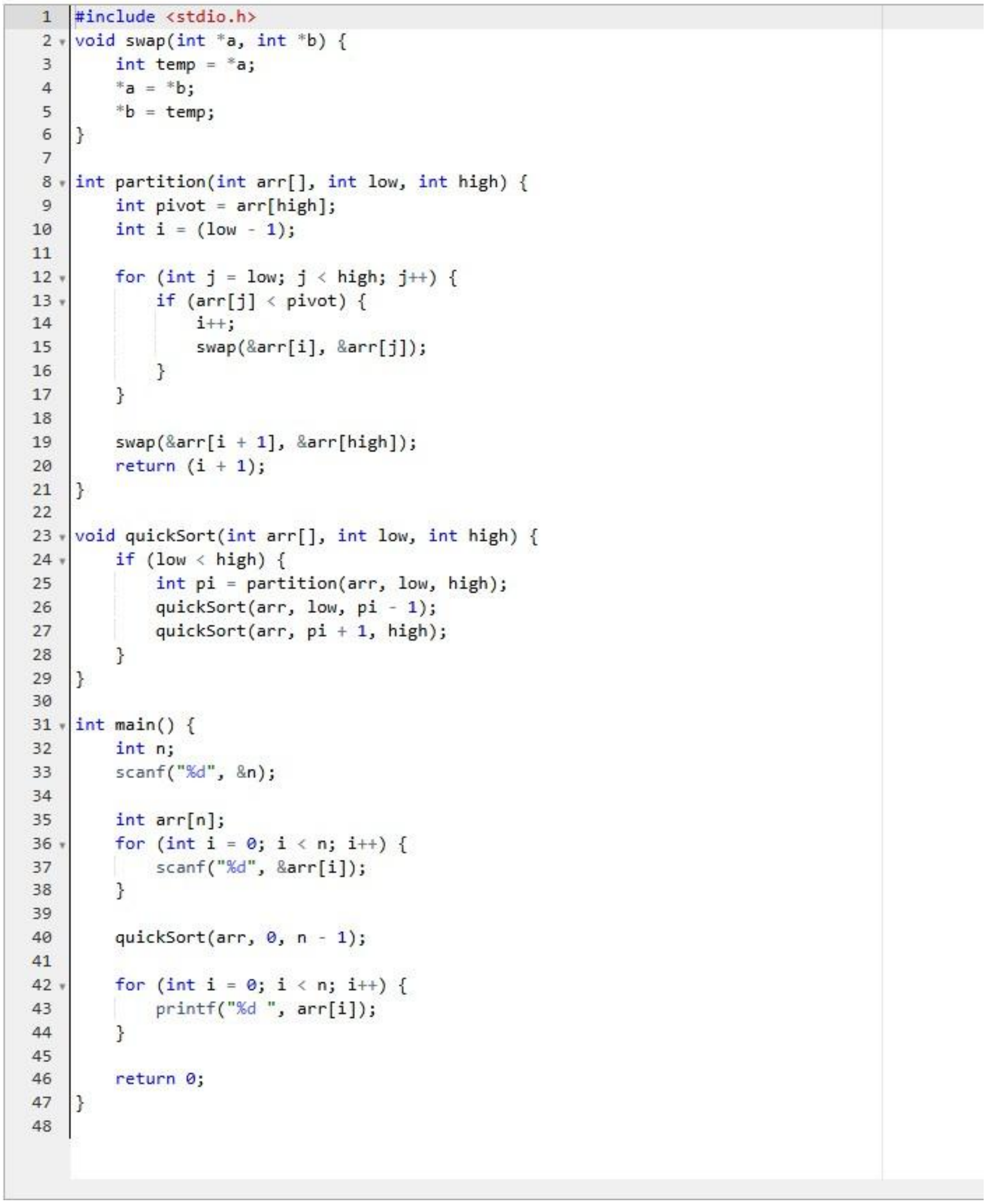

Passed all tests! ✓

Correct

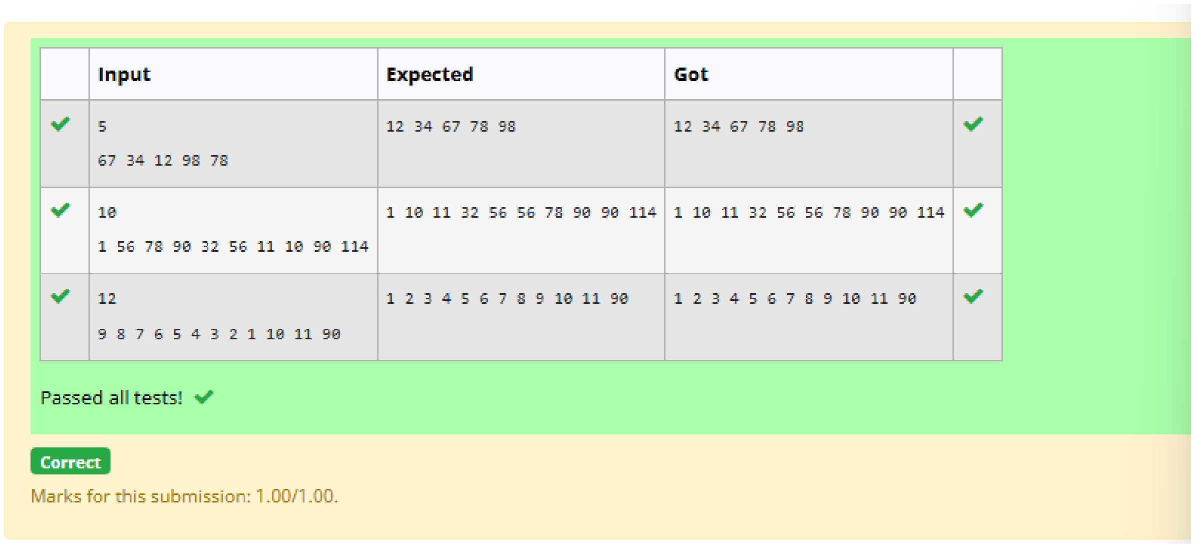
Marks for this submission: 1.00/1.00.

5-**Implementation of Quick Sort**

**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]);
    }

    return 0;
}
```



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 5<br>67 34 12 98 78 | 12 34 67 78 98 | 12 34 67 78 98 | ✓ |
| ✓ | 10<br>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<br>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! ✓

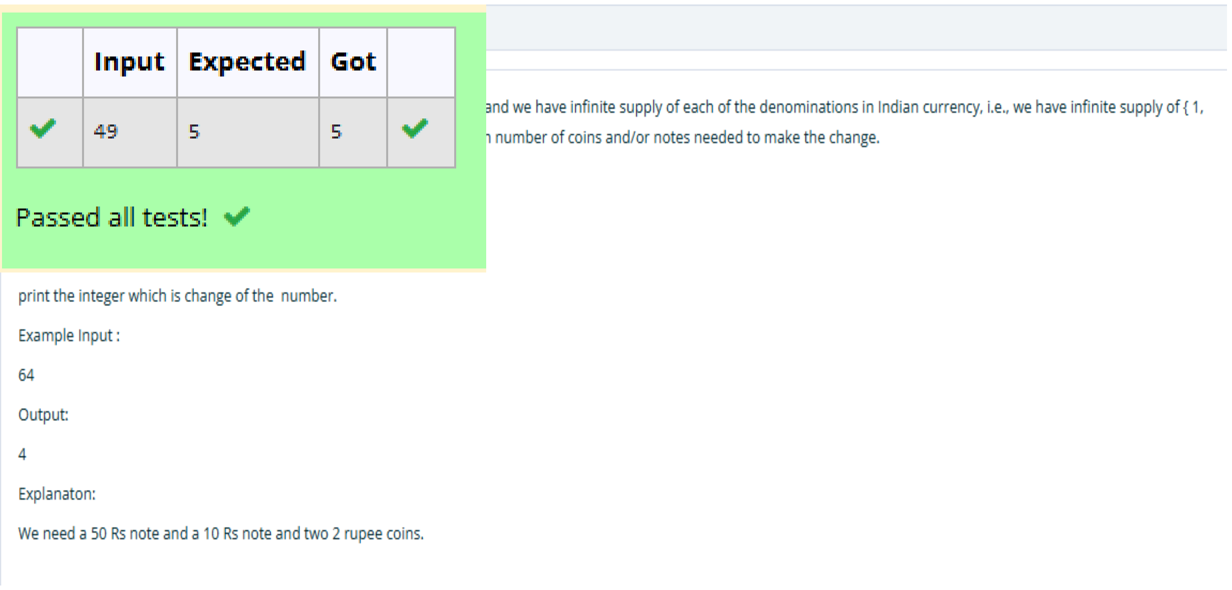
Correct

Marks for this submission: 1.00/1.00.

RITTVIK S 2024-CSE R2

# Greedy Algorithms

## 1-G-Coin Problem:



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 49 | 5 | 5 | ✓ |

Passed all tests! ✓

and we have infinite supply of each of the denominations in Indian currency, i.e., we have infinite supply of { 1,
n number of coins and/or notes needed to make the change.

print the integer which is change of the number.

Example Input :

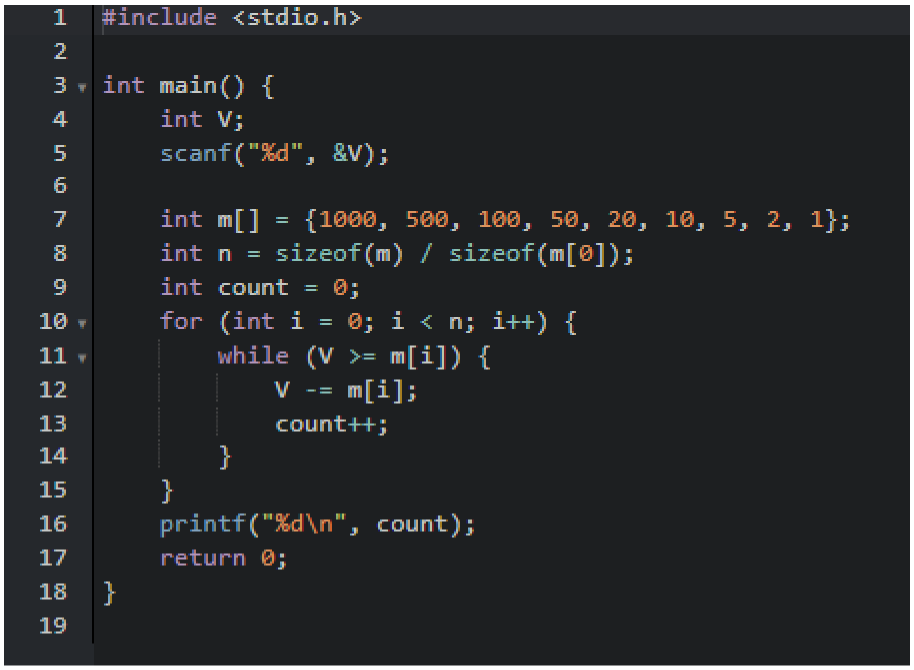
64

Output:

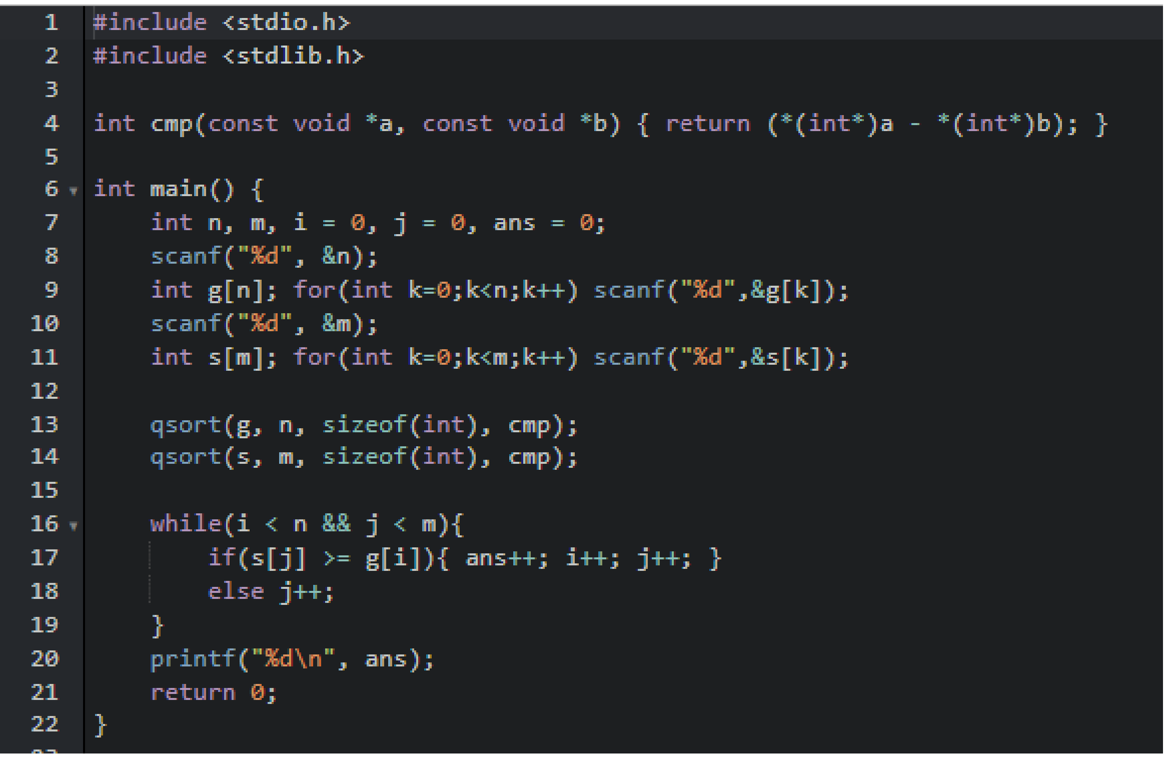
4

Explanaton:

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

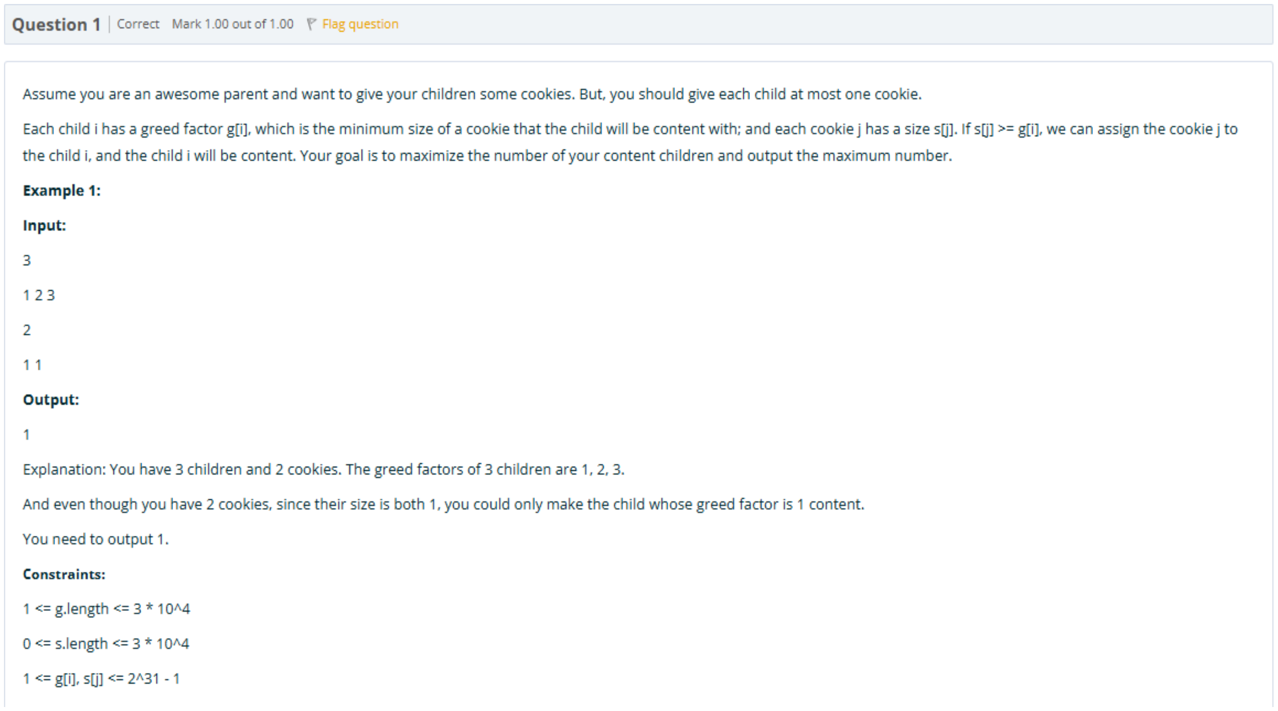


```c
#include <stdio.h>

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

    int m[] = {1000, 500, 100, 50, 20, 10, 5, 2, 1};
    int n = sizeof(m) / sizeof(m[0]);
    int count = 0;
    for (int i = 0; i < n; i++) {
        while (V >= m[i]) {
            V -= m[i];
            count++;
        }
    }
    printf("%d\n", count);
    return 0;
}
```



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

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

int main() {
    int n, m, i = 0, j = 0, ans = 0;
    scanf("%d", &n);
    int g[n]; for(int k=0;k<n;k++) scanf("%d",&g[k]);
    scanf("%d", &m);
    int s[m]; for(int k=0;k<m;k++) scanf("%d",&s[k]);

    qsort(g, n, sizeof(int), cmp);
    qsort(s, m, sizeof(int), cmp);

    while(i < n && j < m){
        if(s[j] >= g[i]){ ans++; i++; j++; }
        else j++;
    }
    printf("%d\n", ans);
    return 0;
}
```

## 2-G-Cookies Problem:



**Question 1** Correct Mark 1.00 out of 1.00 Flag question

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.

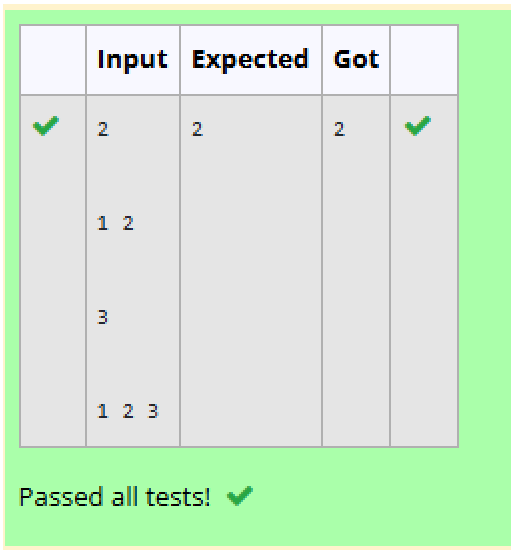
You need to output 1.

**Constraints:**

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

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

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



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 2<br><br>1 2<br><br>3<br><br>1 2 3 | 2 | 2 | ✓ |

Passed all tests! ✓

## 4-G-Array Sum max problem:

**Question 1** | Correct Mark 1.00 out of 1.00 Flag question

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:

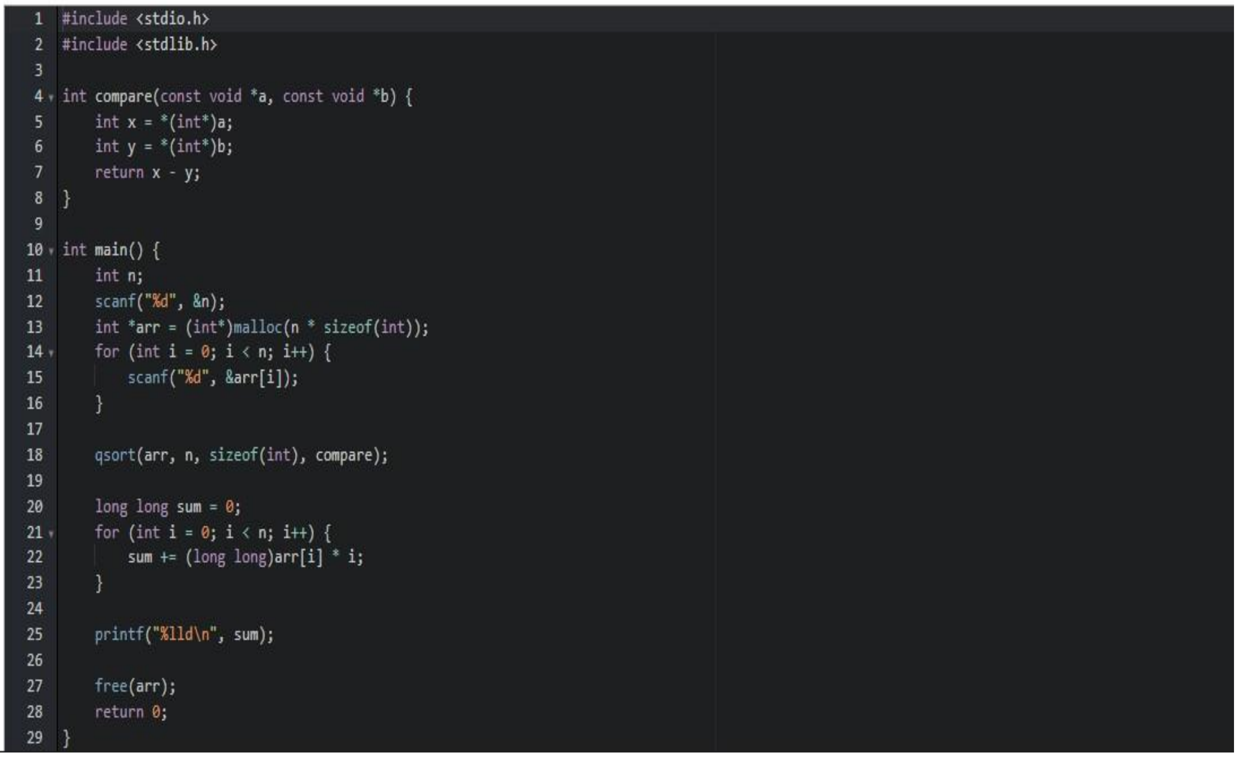
5

2 5 3 4 0

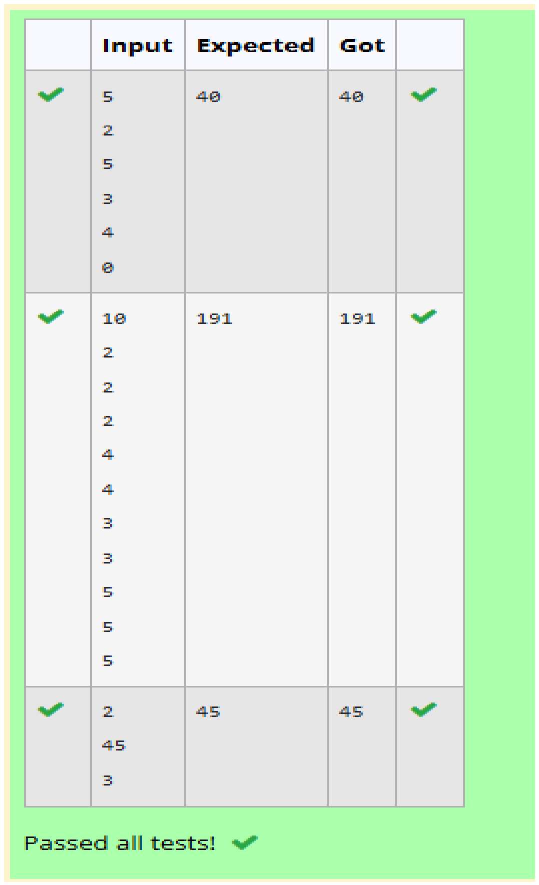
Sample output:

40

**Answer:** (penalty regime: 0 %)



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

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

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);

    long long sum = 0;
    for (int i = 0; i < n; i++) {
        sum += (long long)arr[i] * i;
    }

    printf("%lld\n", sum);

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



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 5<br>2<br>5<br>3<br>4<br>0 | 40 | 40 | ✓ |
| ✓ | 10<br>2<br>2<br>2<br>4<br>4<br>3<br>3<br>5<br>5<br>5 | 191 | 191 | ✓ |
| ✓ | 2<br>45<br>3 | 45 | 45 | ✓ |

Passed all tests! ✓

## 5-G-Product of Array elements-Minimum:

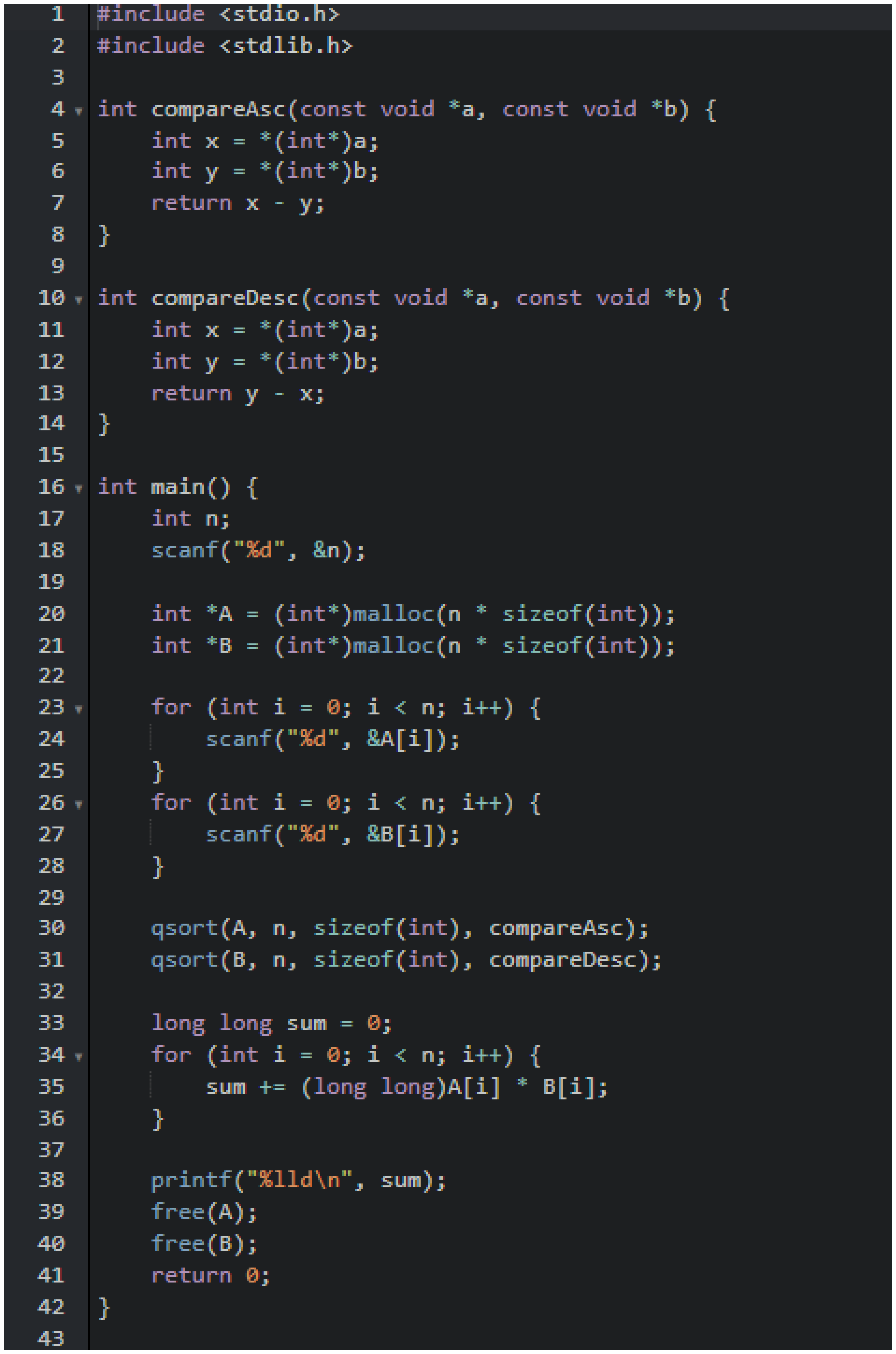
## Question 1

Correct Mark 1.00 out of 1.00 Flag question

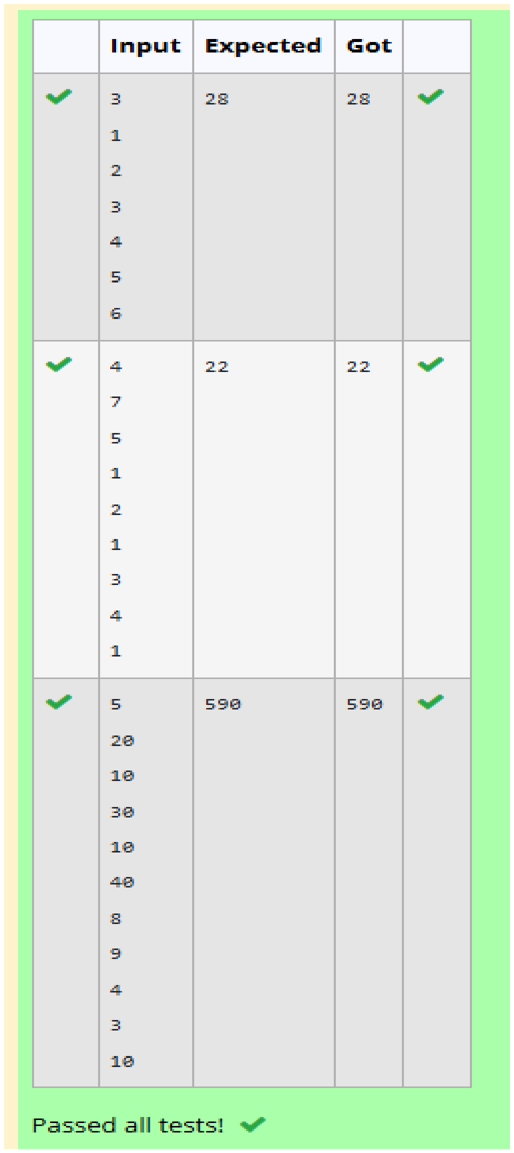
Given two arrays array_One[] and array_Two[] of same size N. We need to first rearrange the arrays such that the sum of the product of pairs( 1 element from each) is minimum. That is SUM (A[i] * B[i]) for all i is minimum.

**For example:**

| Input | Result |
|---|---|
| 3<br>1<br>2<br>3<br>4<br>5<br>6 | 28 |



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

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

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

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

    int *A = (int*)malloc(n * sizeof(int));
    int *B = (int*)malloc(n * sizeof(int));

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

    qsort(A, n, sizeof(int), compareAsc);
    qsort(B, n, sizeof(int), compareDesc);

    long long sum = 0;
    for (int i = 0; i < n; i++) {
        sum += (long long)A[i] * B[i];
    }

    printf("%lld\n", sum);
    free(A);
    free(B);
    return 0;
}

```



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 3<br>1<br>2<br>3<br>4<br>5<br>6 | 28 | 28 | ✓ |
| ✓ | 4<br>7<br>5<br>1<br>2<br>1<br>3<br>4<br>1 | 22 | 22 | ✓ |
| ✓ | 5<br>20<br>10<br>30<br>10<br>40<br>8<br>9<br>4<br>3<br>10 | 590 | 590 | ✓ |

Passed all tests! ✓

RITTVIK S 2024-CSE

R2

Competitive programming:

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

**Question 1** | Correct Mark 1.00 out of 1.00 Flag question

Find Duplicate in Array.

Given a read only array of n integers between 1 and n, find one number that repeats.

Input Format:

First Line - Number of elements

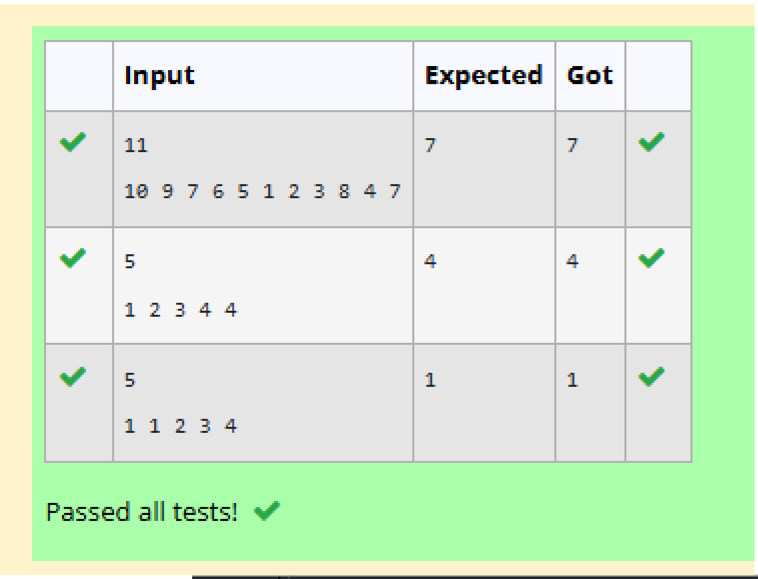
n Lines - n Elements

Output Format:

Element x - That is repeated

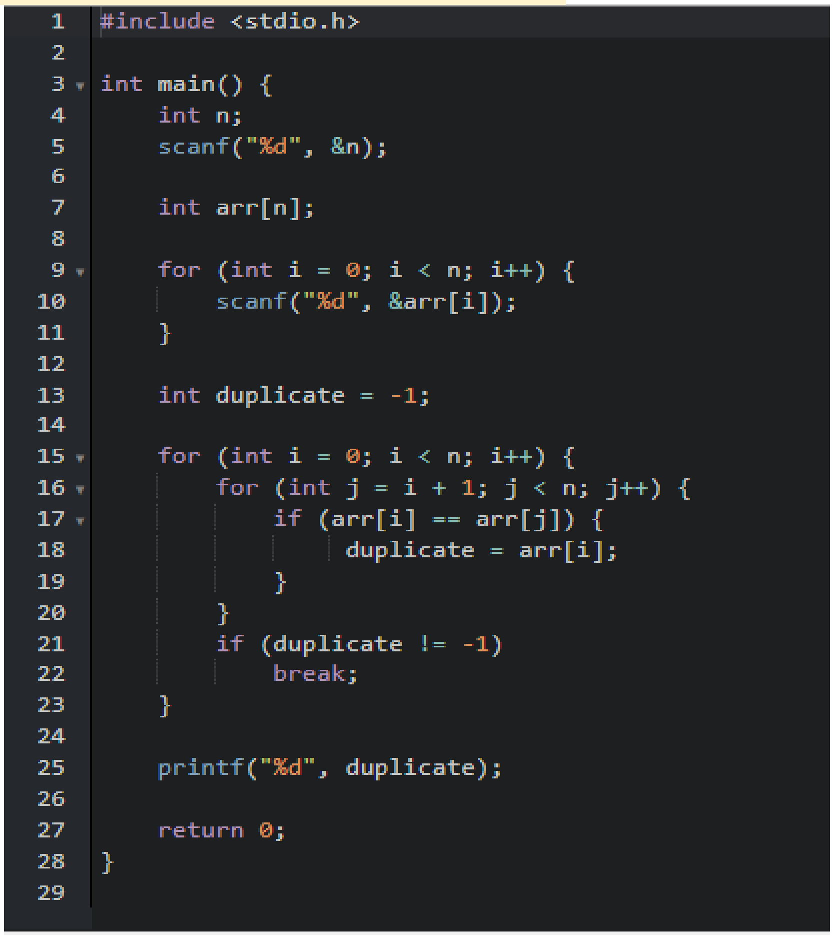
**For example:**

| Input | Result |
|---|---|
| 5<br>1 1 2 3 4 | 1 |



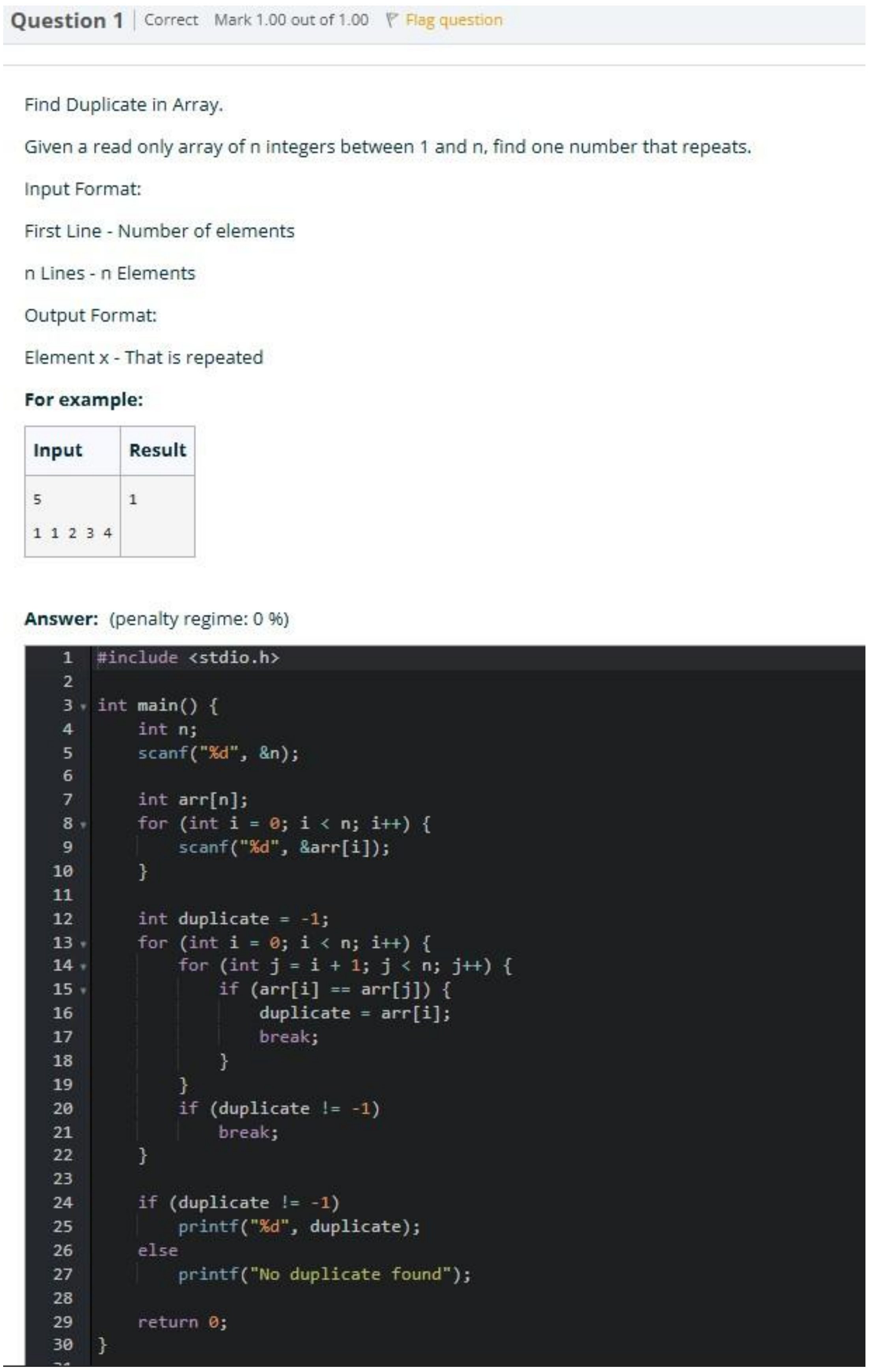
| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 11<br>10 9 7 6 5 1 2 3 8 4 7 | 7 | 7 | ✓ |
| ✓ | 5<br>1 2 3 4 4 | 4 | 4 | ✓ |
| ✓ | 5<br>1 1 2 3 4 | 1 | 1 | ✓ |

Passed all tests! ✓



```c
#include <stdio.h>

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

    int arr[n];

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

    int duplicate = -1;

    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            if (arr[i] == arr[j]) {
                duplicate = arr[i];
            }
        }
        if (duplicate != -1)
            break;
    }

    printf("%d", duplicate);

    return 0;
}
```

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



**Question 1** | Correct Mark 1.00 out of 1.00 Flag question

Find Duplicate in Array.

Given a read only array of n integers between 1 and n, find one number that repeats.

Input Format:

First Line - Number of elements

n Lines - n Elements

Output Format:

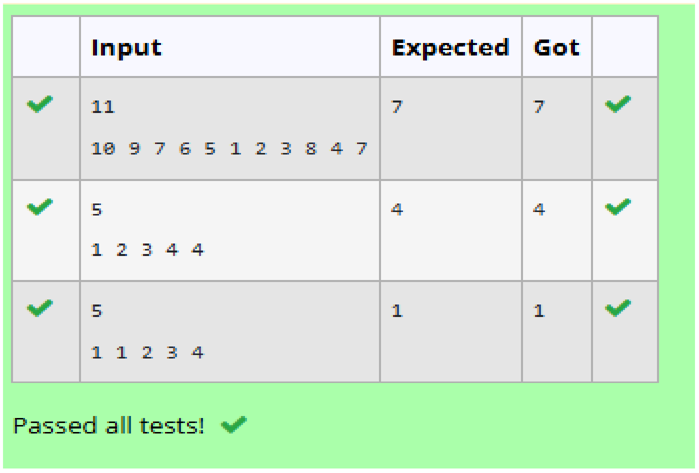
Element x - That is repeated

**For example:**

| Input | Result |
|---|---|
| 5<br>1 1 2 3 4 | 1 |

**Answer:** (penalty regime: 0 %)

```c
#include <stdio.h>

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

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

    int duplicate = -1;
    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            if (arr[i] == arr[j]) {
                duplicate = arr[i];
                break;
            }
        }
        if (duplicate != -1)
            break;
    }

    if (duplicate != -1)
        printf("%d", duplicate);
    else
        printf("No duplicate found");

    return 0;
}
```



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 11<br>10 9 7 6 5 1 2 3 8 4 7 | 7 | 7 | ✓ |
| ✓ | 5<br>1 2 3 4 4 | 4 | 4 | ✓ |
| ✓ | 5<br>1 1 2 3 4 | 1 | 1 | ✓ |

Passed all tests! ✓

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

**Question 1** | Correct Mark 1.00 out of 1.00 Flag question

Find the intersection of two sorted arrays.

OR in other words,

Given 2 sorted arrays, find all the elements which occur in both the arrays.

Input Format

- The first line contains T, the number of test cases. Following T lines contain:

1. Line 1 contains N1, followed by N1 integers of the first array
2. Line 2 contains N2, followed by N2 integers of the second array

Output Format

The intersection of the arrays in a single line

Example

Input:

```
1
3 10 17 57
6 2 7 10 15 57 246
```

Output:

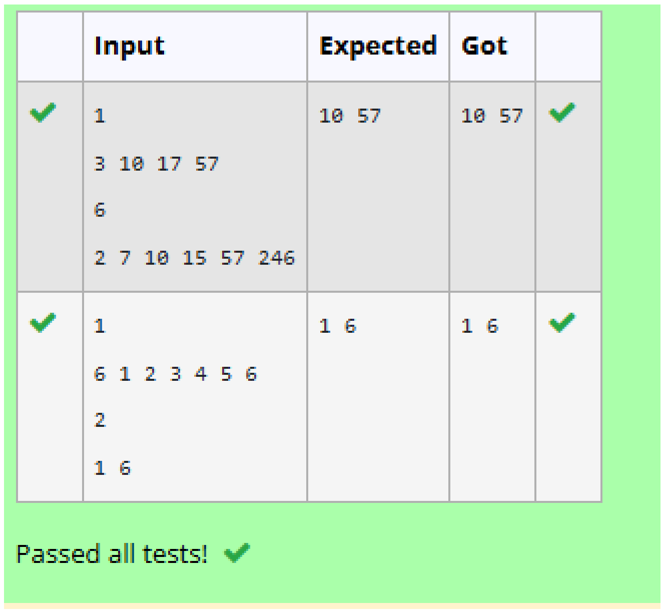
```
10 57
```

Input:

```
1
6 1 2 3 4 5 6
2 1 6
```

Output:

```
1 6
```



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 1<br>3 10 17 57<br>6<br>2 7 10 15 57 246 | 10 57 | 10 57 | ✓ |
| ✓ | 1<br>6 1 2 3 4 5 6<br>2<br>1 6 | 1 6 | 1 6 | ✓ |

Passed all tests! ✓

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

## Question 1

Correct Mark 1.00 out of 1.00 Flag question

Find the intersection of two sorted arrays.

OR in other words,

Given 2 sorted arrays, find all the elements which occur in both the arrays.

Input Format

- The first line contains T, the number of test cases. Following T lines contain:

1. Line 1 contains N1, followed by N1 integers of the first array
2. Line 2 contains N2, followed by N2 integers of the second array

Output Format

The intersection of the arrays in a single line

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

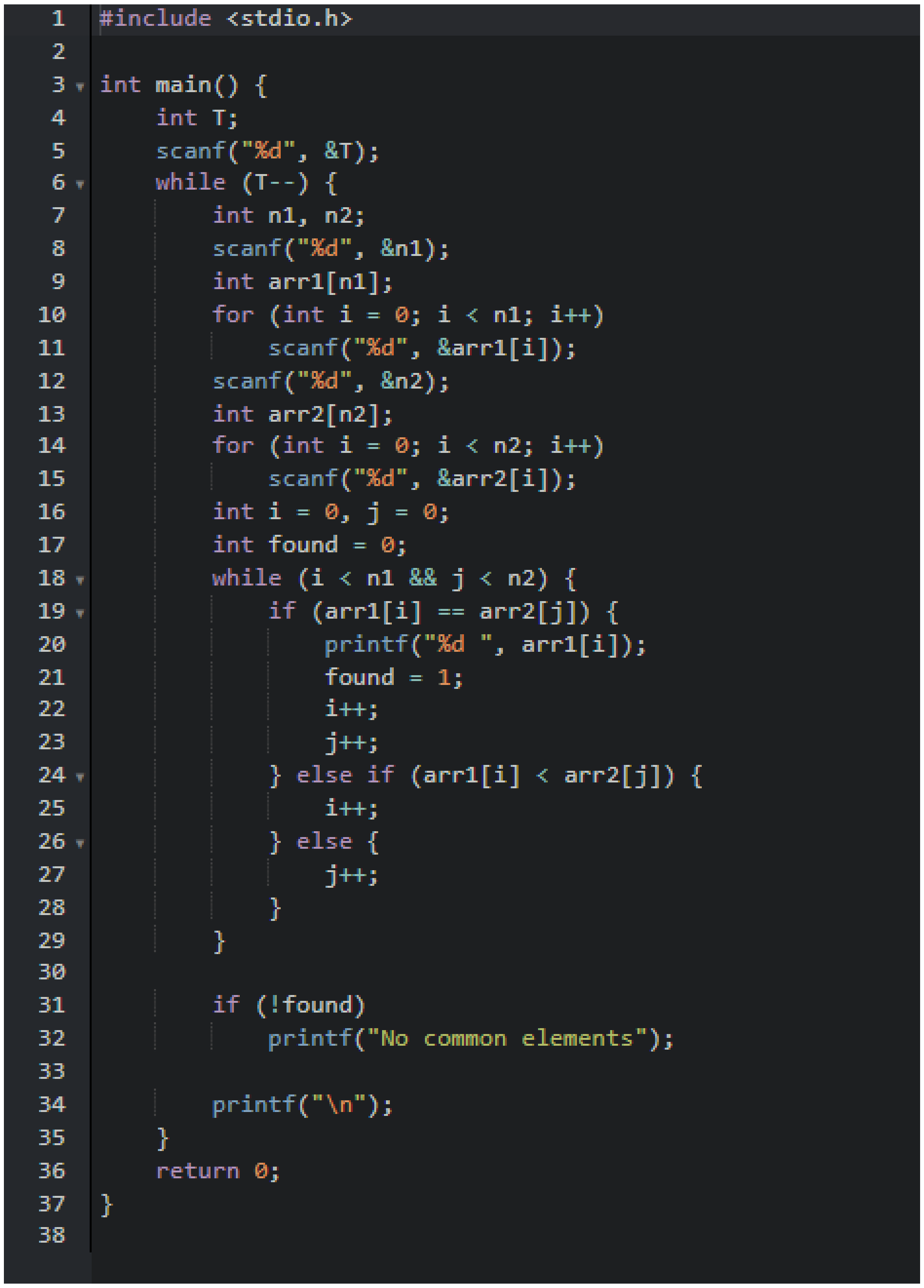
2 1 6

Output:

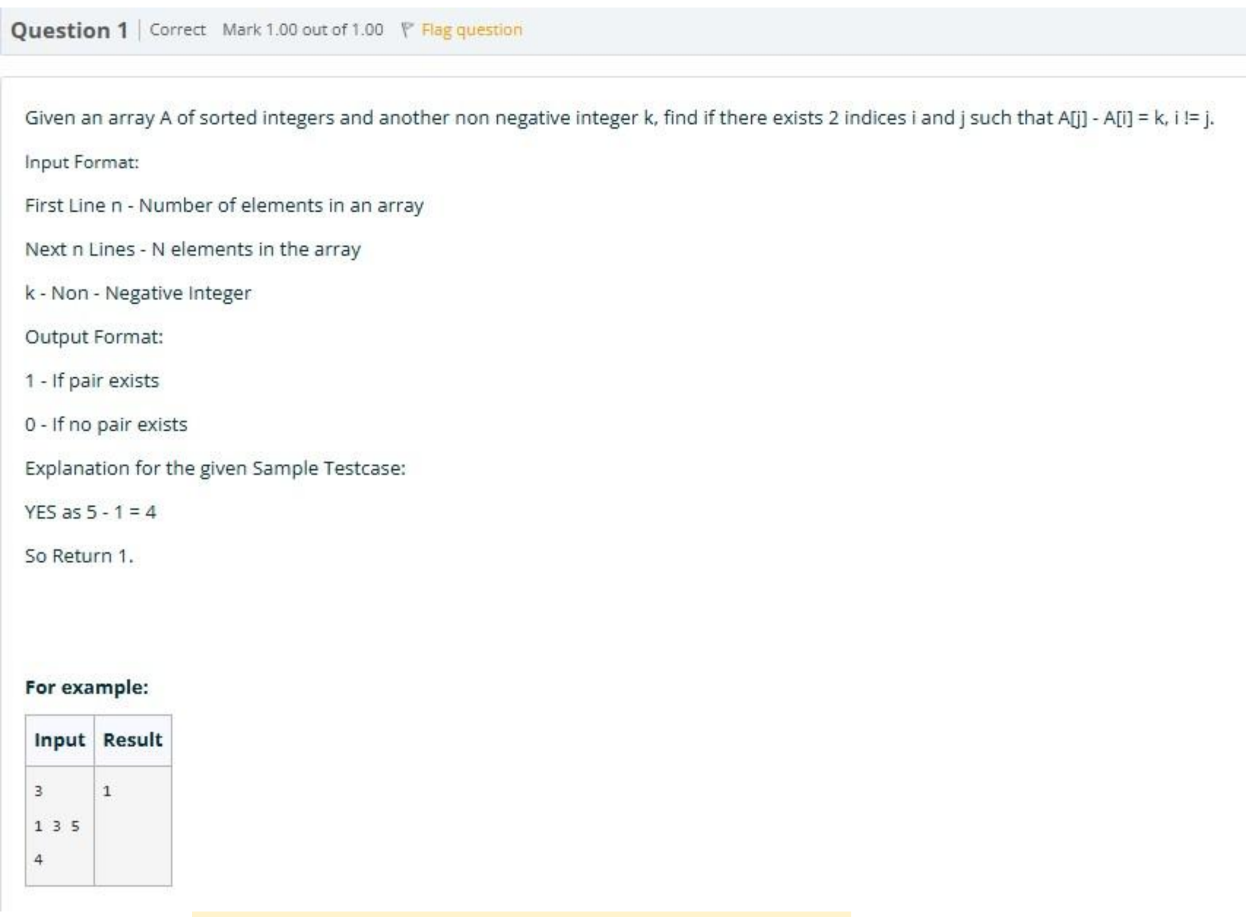
1 6

**For example:**

| Input | Result |
|---|---|
| `1`<br>`3 10 17 57`<br>`6`<br>`2 7 10 15 57 246` | `10 57` |



```c
#include <stdio.h>

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]);
        int i = 0, j = 0;
        int found = 0;
        while (i < n1 && j < n2) {
            if (arr1[i] == arr2[j]) {
                printf("%d ", arr1[i]);
                found = 1;
                i++;
                j++;
            } else if (arr1[i] < arr2[j]) {
                i++;
            } else {
                j++;
            }
        }

        if (!found)
            printf("No common elements");

        printf("\n");
    }
    return 0;
}
```


**Question 1** | Correct Mark 1.00 out of 1.00 Flag question

Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that A[j] - A[i] = k, i != 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

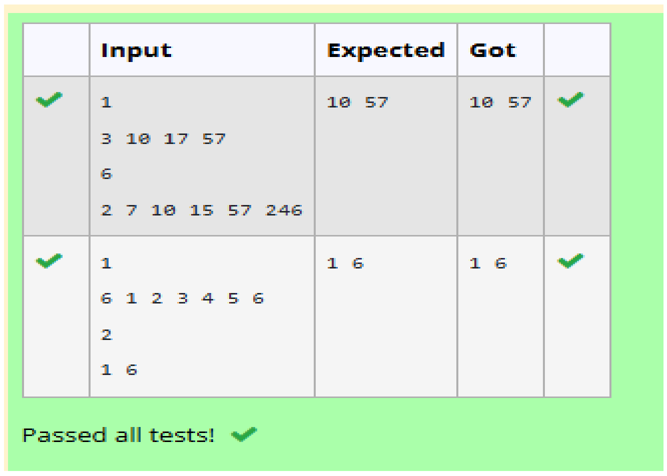
Explanation for the given Sample Testcase:

YES as 5 - 1 = 4

So Return 1.

**For example:**

| Input | Result |
|---|---|
| 3<br>1 3 5<br>4 | 1 |



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 1<br>3 10 17 57<br>6<br>2 7 10 15 57 246 | 10 57 | 10 57 | ✓ |
| ✓ | 1<br>6 1 2 3 4 5 6<br>2<br>1 6 | 1 6 | 1 6 | ✓ |

Passed all tests! ✓

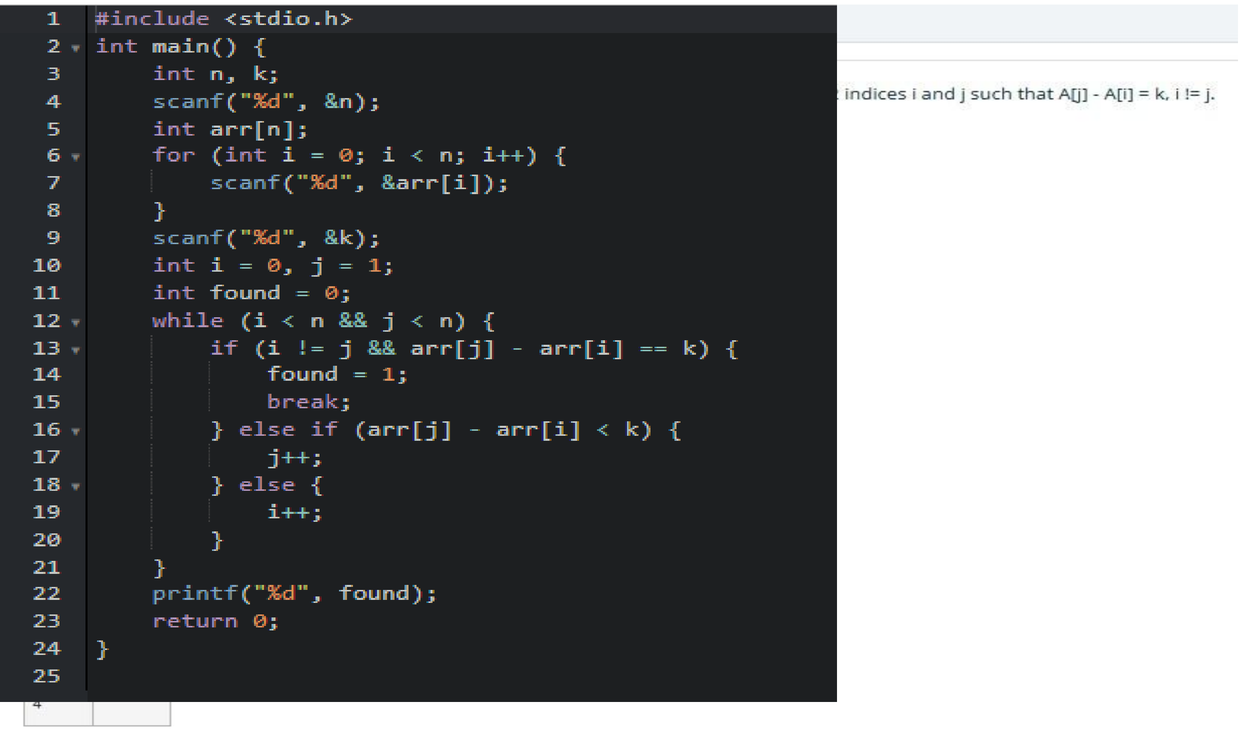
5- Pair with Difference-O(n^2)Time Complexity,O(1) Space Complexity:

**Answer:** (penalty regime: 0 %)

```c
#include <stdio.h>
int main() {
    int n, k;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    scanf("%d", &k);
    int i = 0, j = 1;
    int found = 0;
    while (i < n && j < n) {
        if (i != j && arr[j] - arr[i] == k) {
            found = 1;
            break;
        }
        else if (arr[j] - arr[i] < k) {
            j++;
        }
        else {
            i++;
        }
    }
    printf("%d", found);
    return 0;
}

```

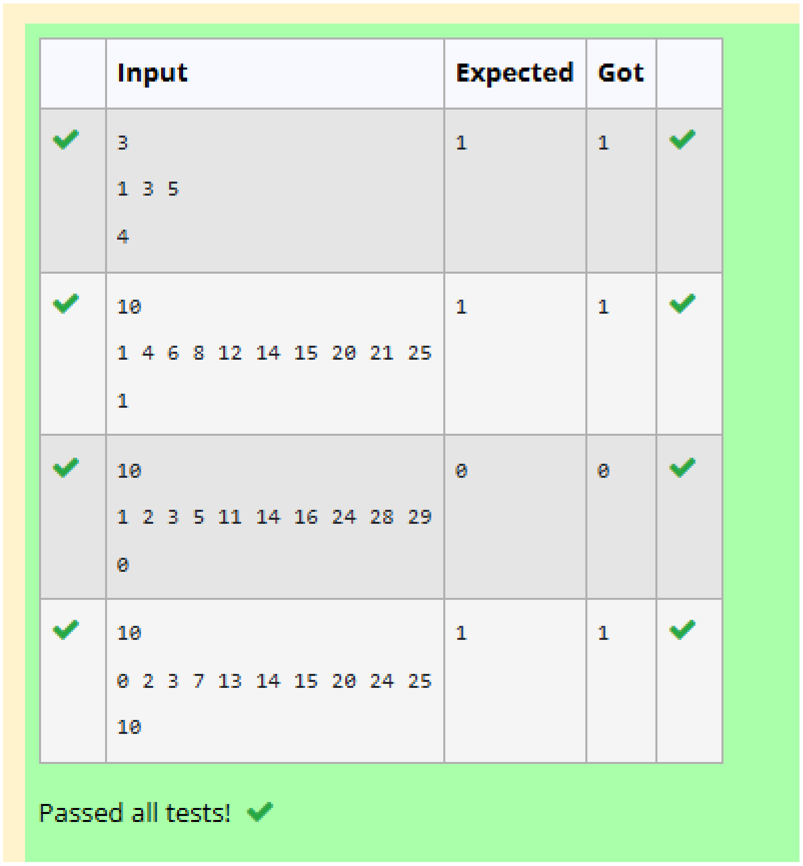
| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 3<br>1 3 5<br>4 | 1 | 1 | ✓ |
| ✓ | 10<br>1 4 6 8 12 14 15 20 21 25<br>1 | 1 | 1 | ✓ |
| ✓ | 10<br>1 2 3 5 11 14 16 24 28 29<br>0 | 0 | 0 | ✓ |
| ✓ | 10<br>0 2 3 7 13 14 15 20 24 25<br>10 | 1 | 1 | ✓ |

Passed all tests! ✓


indices i and j such that A[j] - A[i] = k, i != j.

```c
#include <stdio.h>
int main() {
    int n, k;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    scanf("%d", &k);
    int i = 0, j = 1;
    int found = 0;
    while (i < n && j < n) {
        if (i != j && arr[j] - arr[i] == k) {
            found = 1;
            break;
        } else if (arr[j] - arr[i] < k) {
            j++;
        } else {
            i++;
        }
    }
    printf("%d", found);
    return 0;
}
```

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



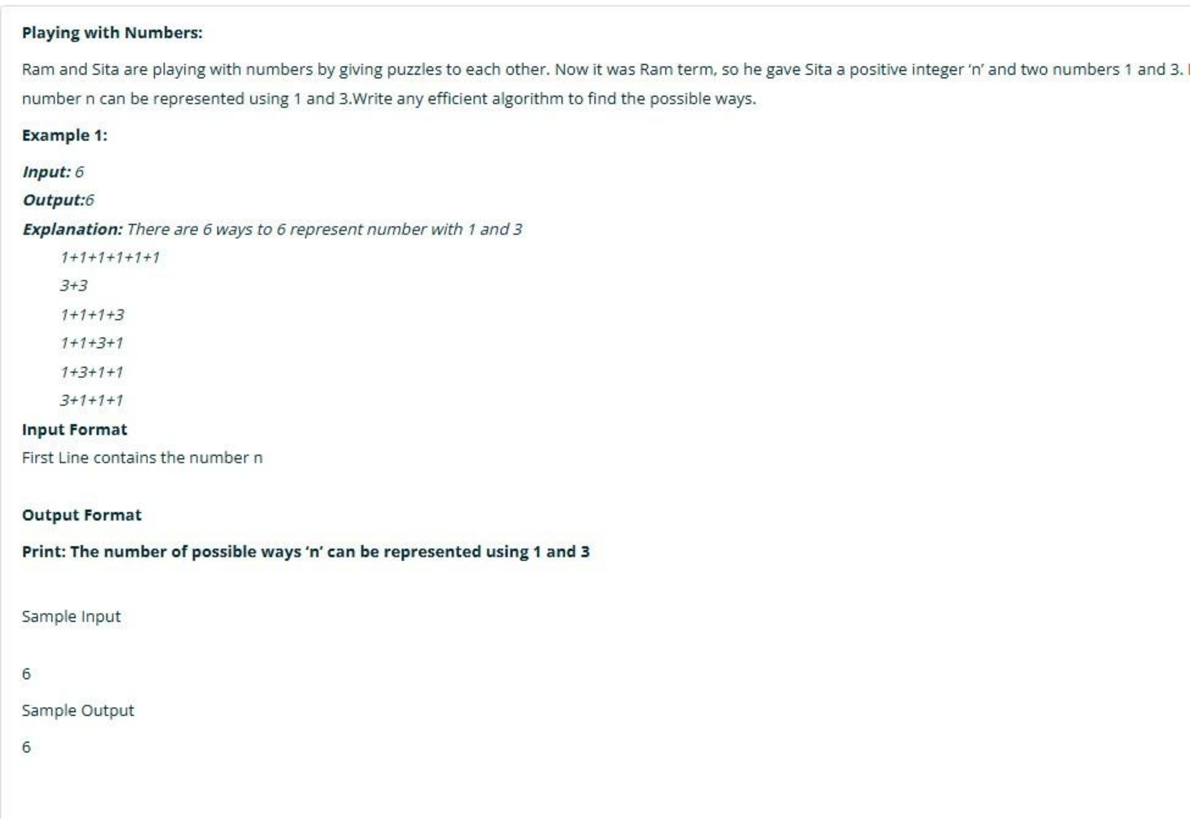
| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 3<br>1 3 5<br>4 | 1 | 1 | ✓ |
| ✓ | 10<br>1 4 6 8 12 14 15 20 21 25<br>1 | 1 | 1 | ✓ |
| ✓ | 10<br>1 2 3 5 11 14 16 24 28 29<br>0 | 0 | 0 | ✓ |
| ✓ | 10<br>0 2 3 7 13 14 15 20 24 25<br>10 | 1 | 1 | ✓ |

Passed all tests! ✓

RITTVIK S 2024-CSE R2

**Dynamic programming**

**1-DP-Playing with Numbers**



**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.
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**

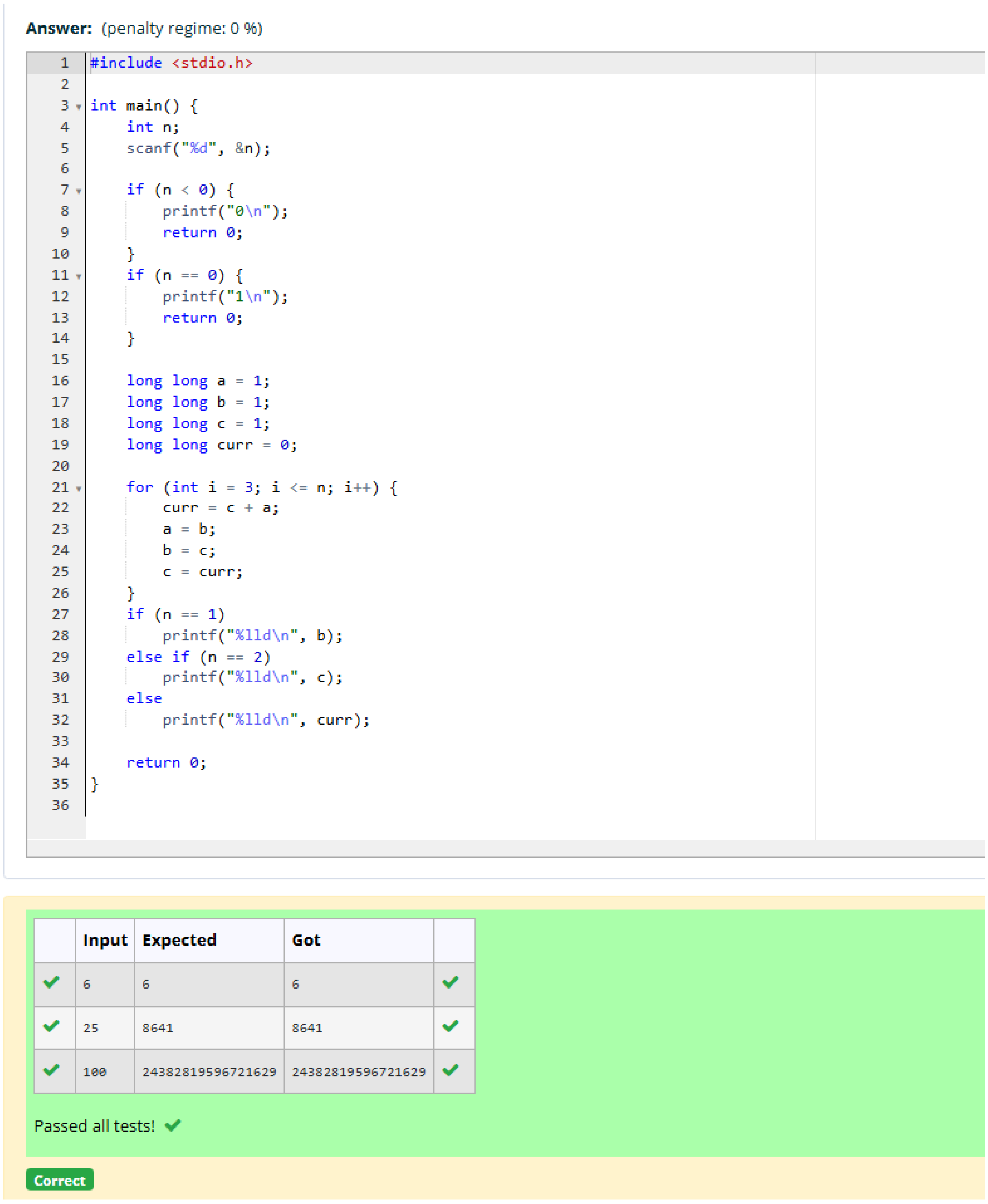
**Print: The number of possible ways 'n' can be represented using 1 and 3**

Sample Input

```
6
```

Sample Output

```
6
```


**Answer:** (penalty regime: 0 %)

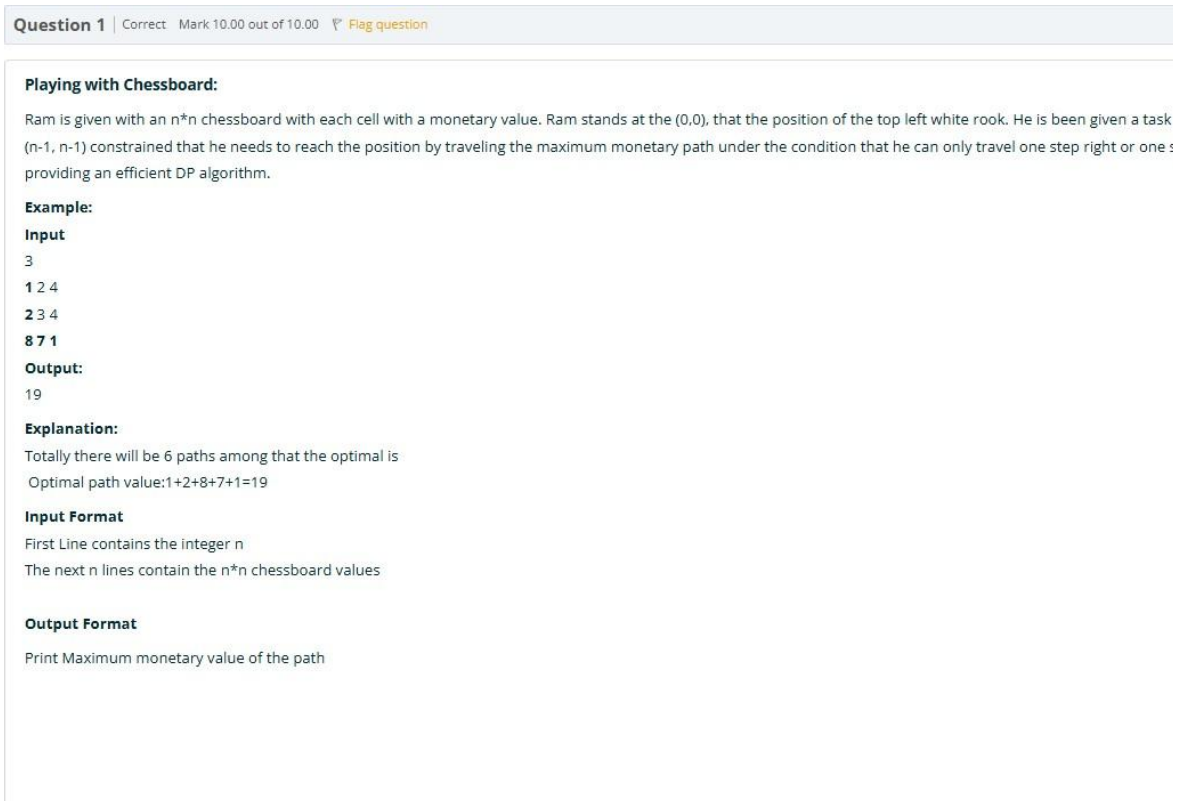
```c
#include <stdio.h>

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

    if (n < 0) {
        printf("0\n");
        return 0;
    }
    if (n == 0) {
        printf("1\n");
        return 0;
    }

    long long a = 1;
    long long b = 1;
    long long c = 1;
    long long curr = 0;

    for (int i = 3; i <= n; i++) {
        curr = c + a;
        a = b;
        b = c;
        c = curr;
    }
    if (n == 1)
        printf("%lld\n", b);
    else if (n == 2)
        printf("%lld\n", c);
    else
        printf("%lld\n", curr);

    return 0;
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 6 | 6 | 6 | ✓ |
| ✓ | 25 | 8641 | 8641 | ✓ |
| ✓ | 100 | 24382819596721629 | 24382819596721629 | ✓ |

Passed all tests! ✓

Correct

## 2-DP-Playing with chessboard



**Question 1** | Correct Mark 10.00 out of 10.00 Flag question

**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 (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 s 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

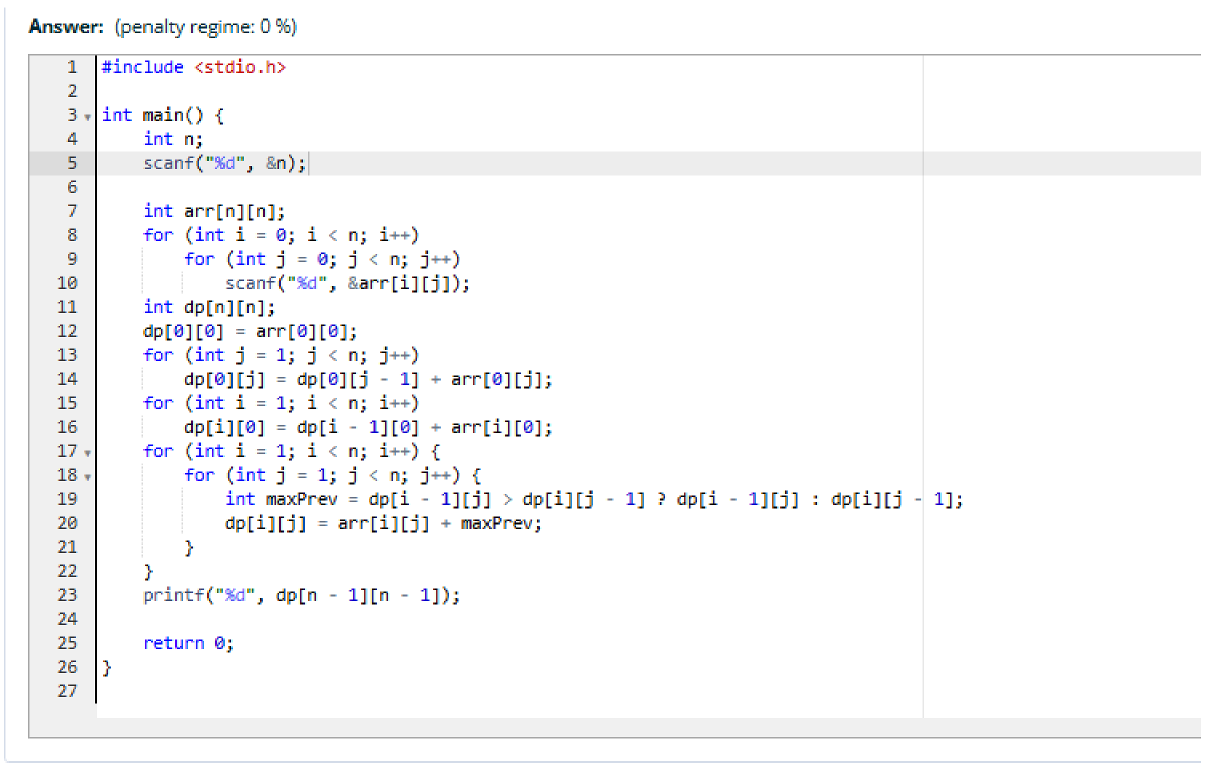
**Input Format**

First Line contains the integer n

The next n lines contain the n*n chessboard values

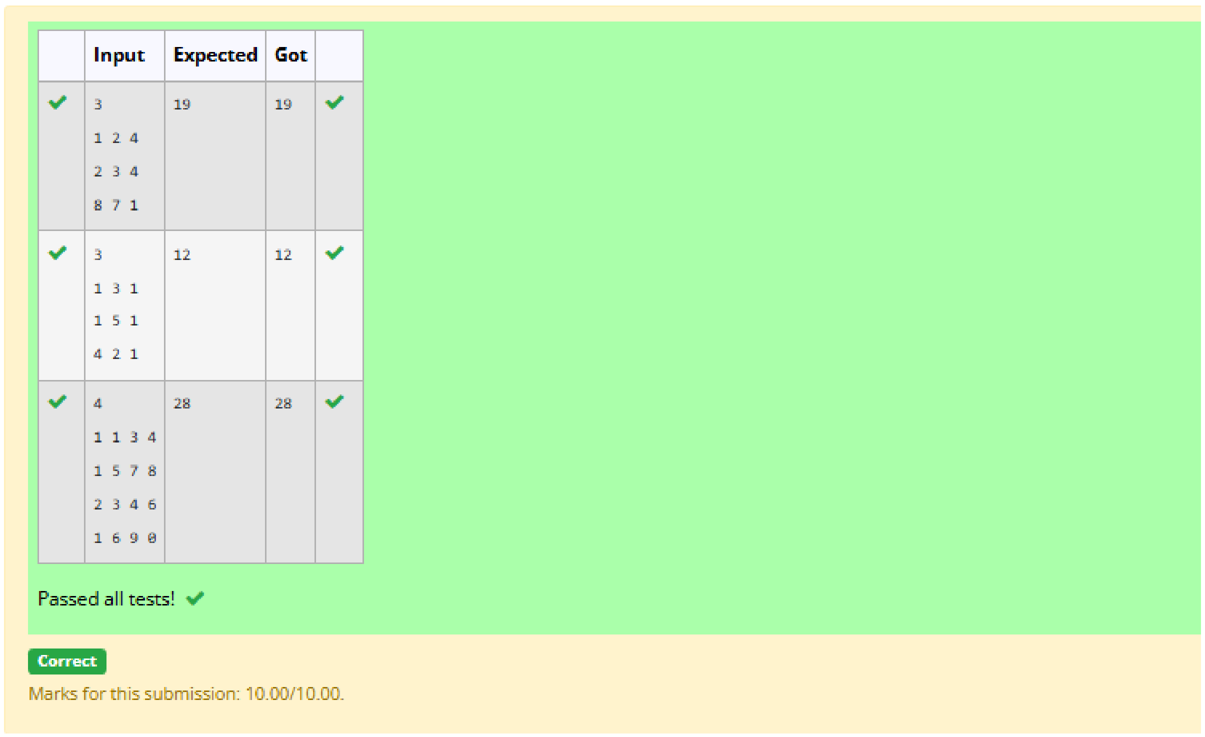
**Output Format**

Print Maximum monetary value of the path



**Answer:** (penalty regime: 0 %)

```c
#include <stdio.h>

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

    int arr[n][n];
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            scanf("%d", &arr[i][j]);
    int dp[n][n];
    dp[0][0] = arr[0][0];
    for (int j = 1; j < n; j++)
        dp[0][j] = dp[0][j - 1] + arr[0][j];
    for (int i = 1; i < n; i++)
        dp[i][0] = dp[i - 1][0] + arr[i][0];
    for (int i = 1; i < n; i++) {
        for (int j = 1; j < n; j++) {
            int maxPrev = dp[i - 1][j] > dp[i][j - 1] ? dp[i - 1][j] : dp[i][j - 1];
            dp[i][j] = arr[i][j] + maxPrev;
        }
    }
    printf("%d", dp[n - 1][n - 1]);

    return 0;
}
```



| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 3<br>1 2 4<br>2 3 4<br>8 7 1 | 19 | 19 | ✓ |
| ✓ | 3<br>1 3 1<br>1 5 1<br>4 2 1 | 12 | 12 | ✓ |
| ✓ | 4<br>1 1 3 4<br>1 5 7 8<br>2 3 4 6<br>1 6 9 0 | 28 | 28 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

## 3-DP-Longest Common Subsequence

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**

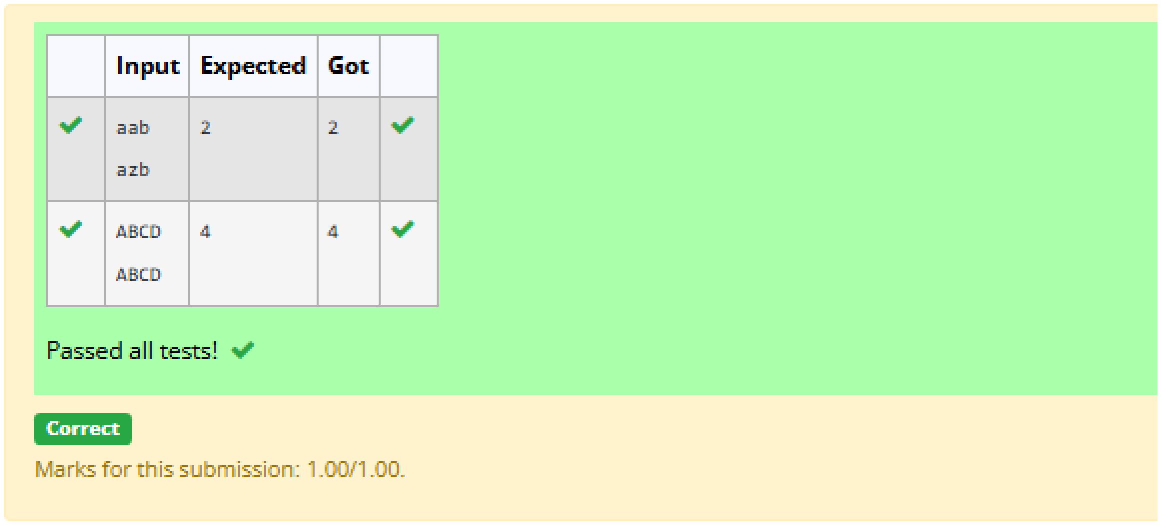
Solveing it using Dynamic Programming

**For example:**

| Input | Result |
|---|---|
| aab<br>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[100], s2[100];
    scanf("%s %s", s1, 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++)
            dp[i][j] = 0;
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= m; j++) {
            if (s1[i - 1] == s2[j - 1])
                dp[i][j] = 1 + dp[i - 1][j - 1];
            else
                dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]);
        }
    }

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

    return 0;
```



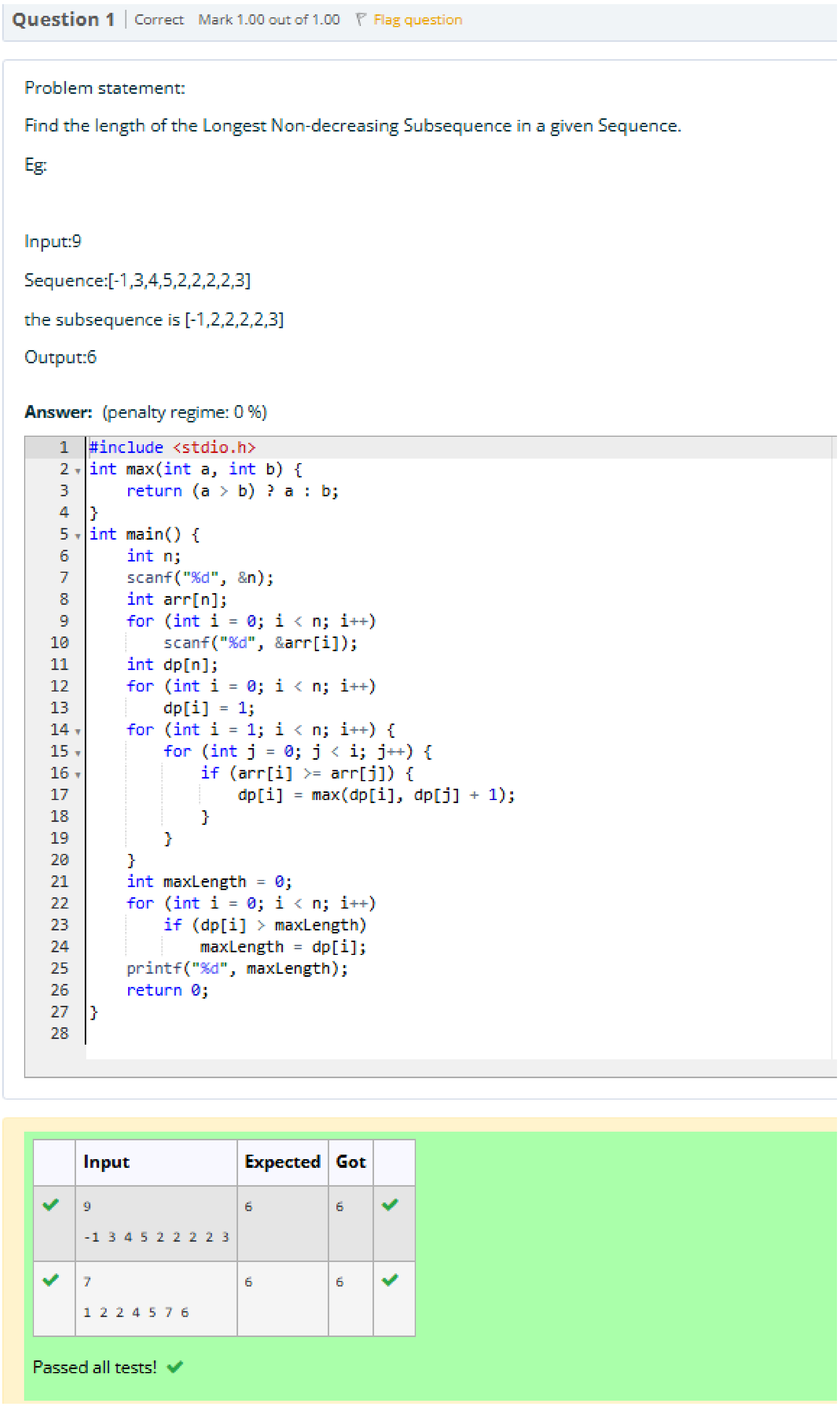
| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | aab<br>azb | 2 | 2 | ✓ |
| ✓ | ABCD<br>ABCD | 4 | 4 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

**4-DP-Longest non-decreasing Subsequence**


**Question 1** Correct Mark 1.00 out of 1.00 Flag question

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 max(int a, int b) {
    return (a > b) ? a : b;
}
int main() {
    int n;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[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 (arr[i] >= arr[j]) {
                dp[i] = max(dp[i], dp[j] + 1);
            }
        }
    }
    int maxLength = 0;
    for (int i = 0; i < n; i++)
        if (dp[i] > maxLength)
            maxLength = dp[i];
    printf("%d", maxLength);
    return 0;
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 9<br>-1 3 4 5 2 2 2 2 3 | 6 | 6 | ✓ |
| ✓ | 7<br>1 2 2 4 5 7 6 | 6 | 6 | ✓ |

Passed all tests! ✓