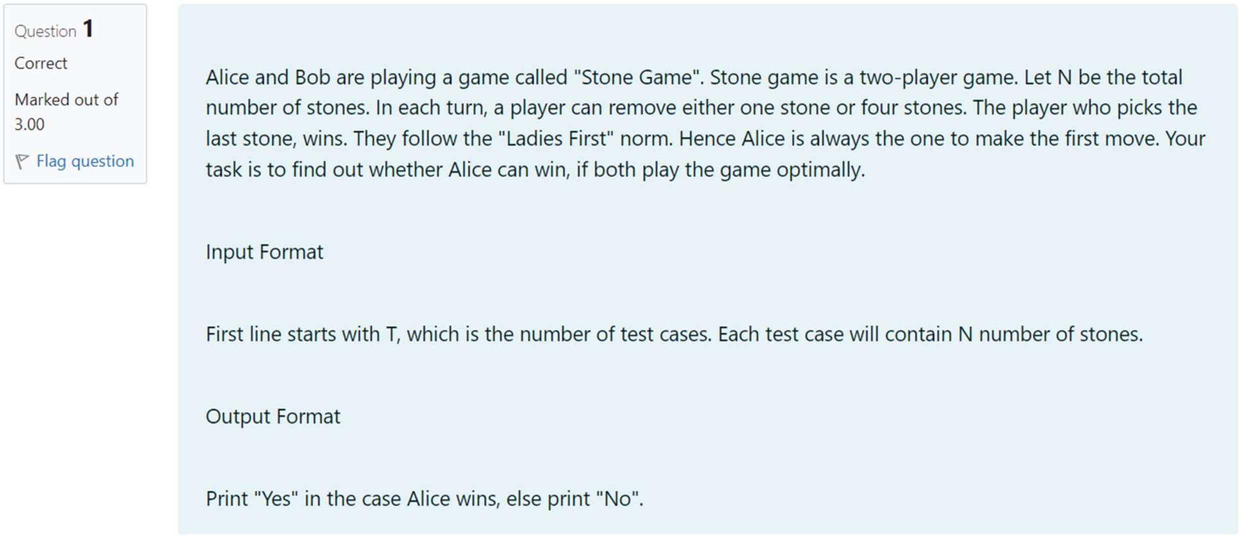
RITTVIK S 2024-CSE 2116240701616

Week-04-Decision Making and Looping - while, do...while and for

Week-04-01-Practice Session-Coding



Question 1

Correct

Marked out of 3.00

Alice and Bob are playing a game called "Stone Game". Stone game is a two-player game. Let N be the total number of stones. In each turn, a player can remove either one stone or four stones. The player who picks the last stone, wins. They follow the "Ladies First" norm. Hence Alice is always the one to make the first move. Your task is to find out whether Alice can win, if both play the game optimally.

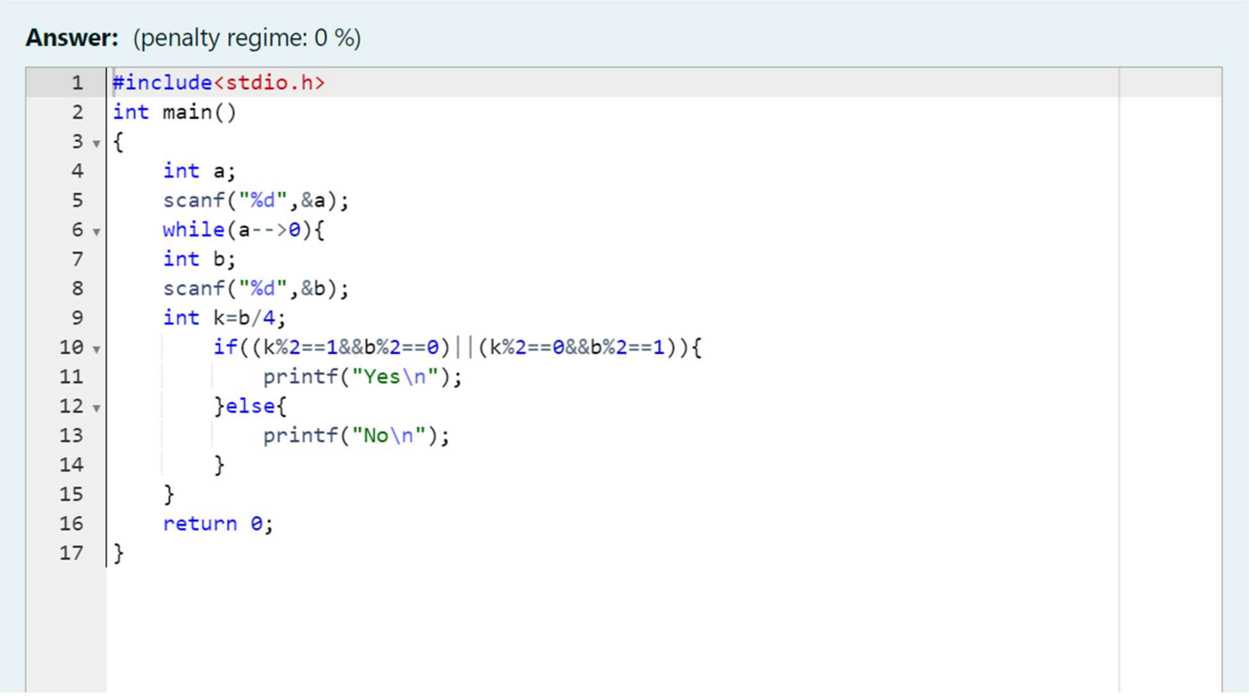
Input Format

First line starts with T, which is the number of test cases. Each test case will contain N number of stones.

Output Format

Print "Yes" in the case Alice wins, else print "No".

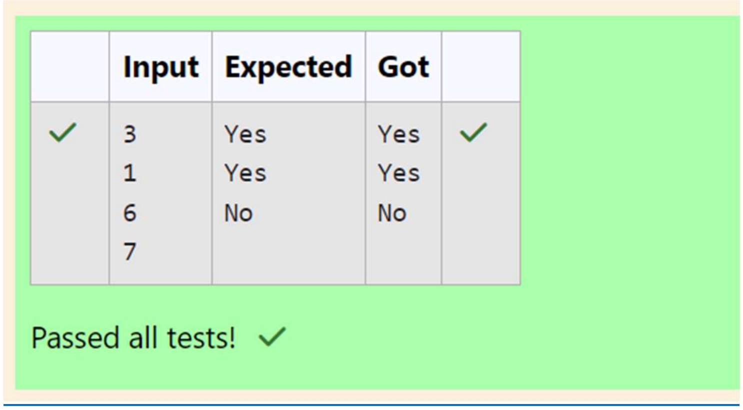
Source code



Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int a;
    scanf("%d",&a);
    while(a-->0){
    int b;
    scanf("%d",&b);
    int k=b/4;
        if((k%2==1&&b%2==0)||(k%2==0&&b%2==1)){
            printf("Yes\n");
        }else{
            printf("No\n");
        }
    }
    return 0;
}
```

Output

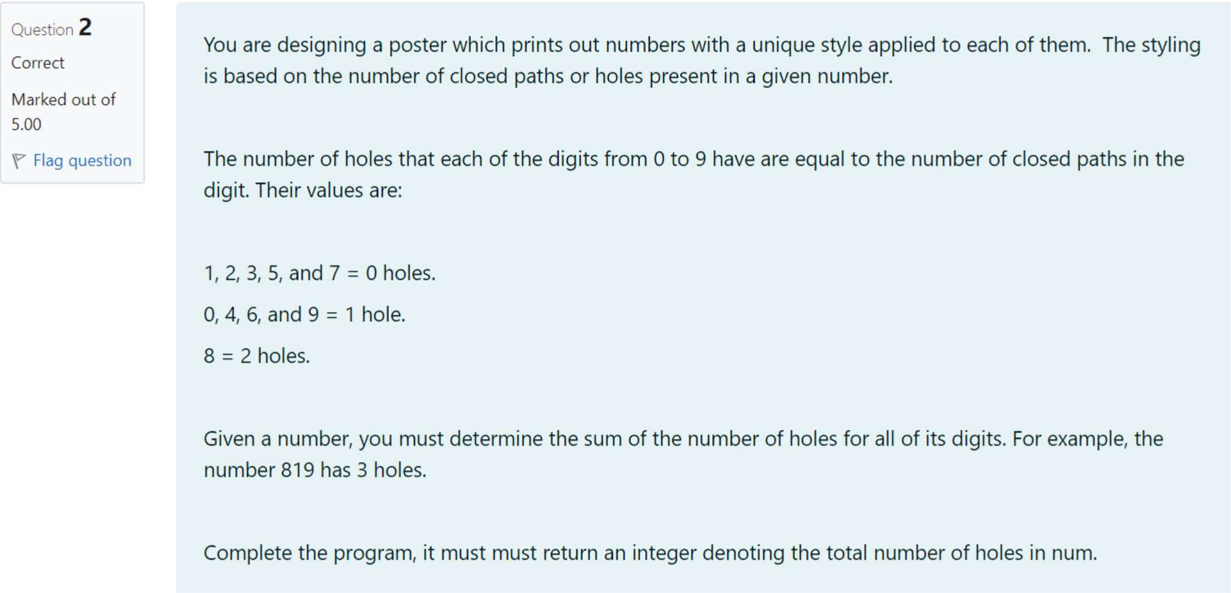


	Input	Expected	Got	
✓	3 1 6 7	Yes Yes No	Yes Yes No	✓

Passed all tests! ✓

Result

The above program is executed successfully and provides the above output.



Question **2**

Correct

Marked out of 5.00

Flag question

You are designing a poster which prints out numbers with a unique style applied to each of them. The styling is based on the number of closed paths or holes present in a given number.

The number of holes that each of the digits from 0 to 9 have are equal to the number of closed paths in the digit. Their values are:

1, 2, 3, 5, and 7 = 0 holes.

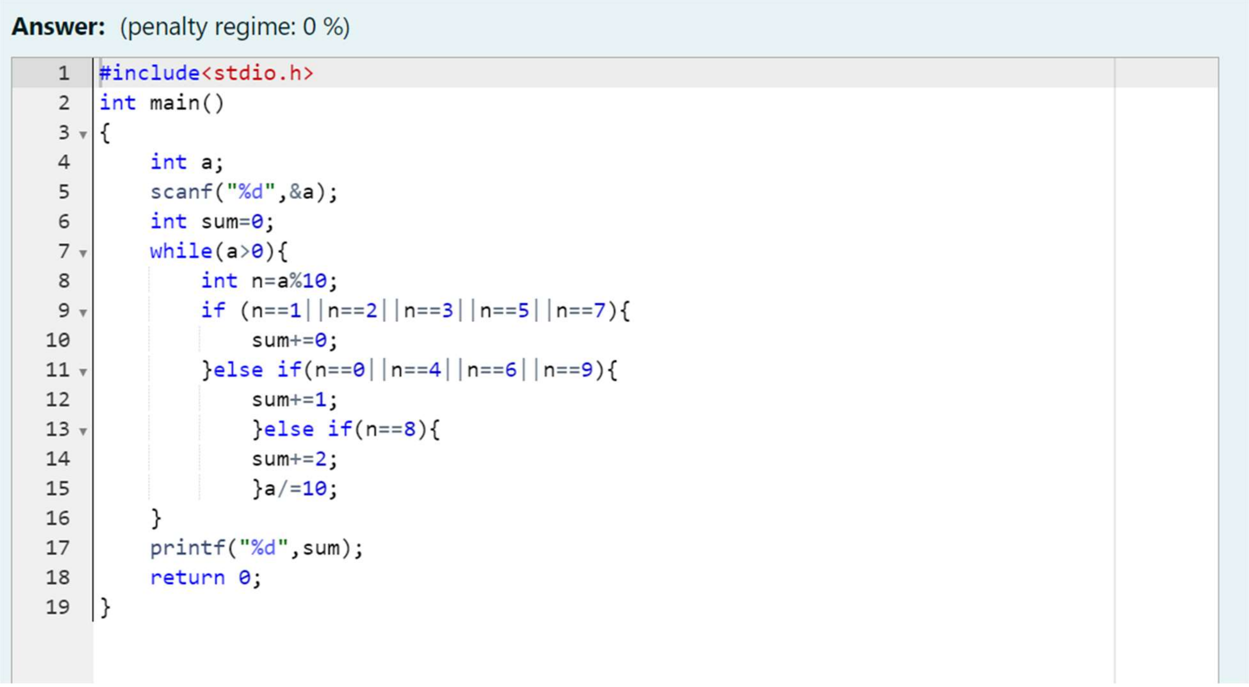
0, 4, 6, and 9 = 1 hole.

8 = 2 holes.

Given a number, you must determine the sum of the number of holes for all of its digits. For example, the number 819 has 3 holes.

Complete the program, it must must return an integer denoting the total number of holes in num.

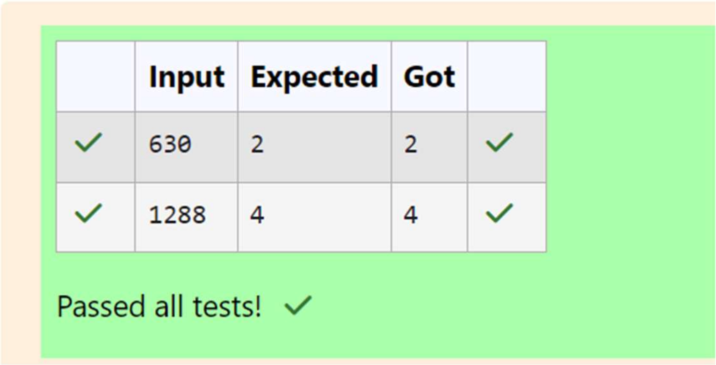
Source code



Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int a;
    scanf("%d",&a);
    int sum=0;
    while(a>0){
        int n=a%10;
        if (n==1||n==2||n==3||n==5||n==7){
            sum+=0;
        }else if(n==0||n==4||n==6||n==9){
            sum+=1;
            }else if(n==8){
            sum+=2;
            }a/=10;
    }
    printf("%d",sum);
    return 0;
}
```

Output

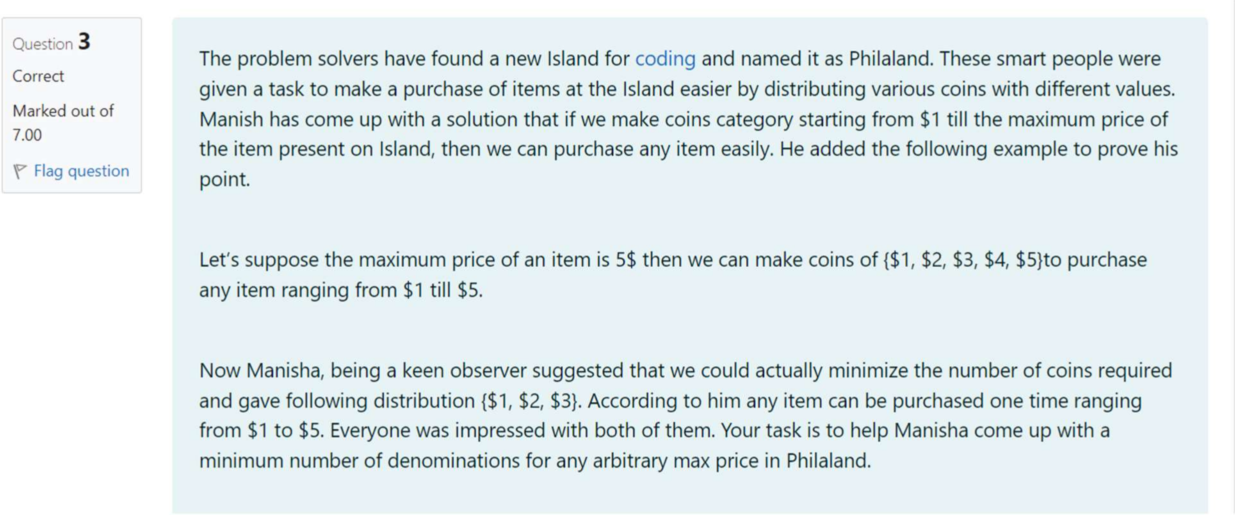


	Input	Expected	Got	
✓	630	2	2	✓
✓	1288	4	4	✓

Passed all tests! ✓

Result

The above program is executed successfully and provides the above output.



Question **3**

Correct

Marked out of 7.00

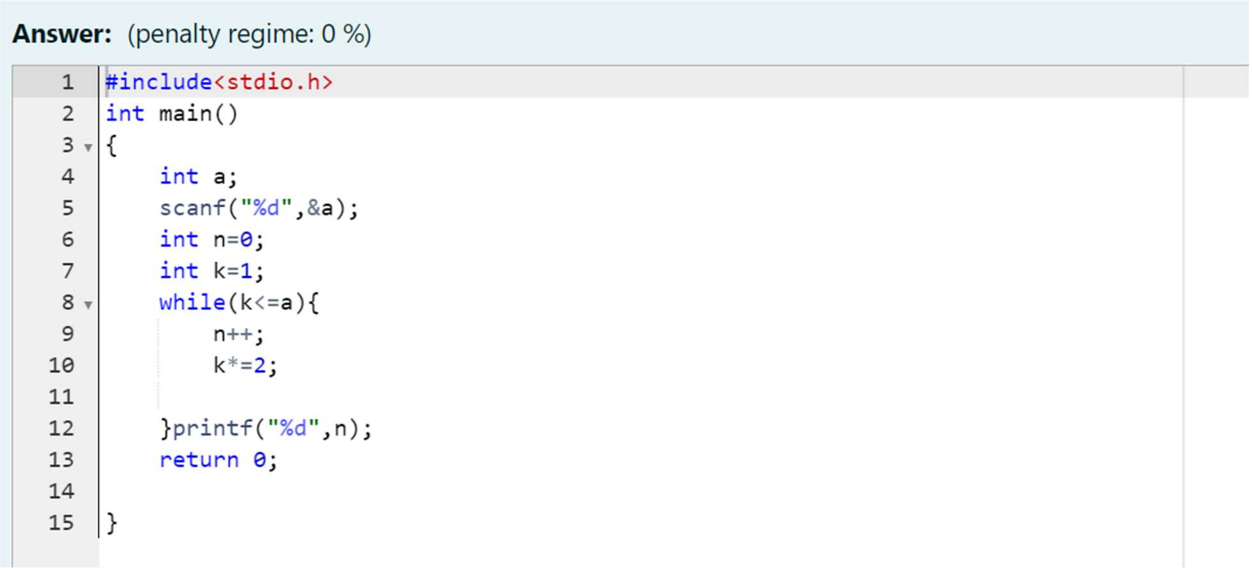
Flag question

The problem solvers have found a new Island for coding and named it as Philaland. These smart people were given a task to make a purchase of items at the Island easier by distributing various coins with different values. Manish has come up with a solution that if we make coins category starting from $1 till the maximum price of the item present on Island, then we can purchase any item easily. He added the following example to prove his point.

Let's suppose the maximum price of an item is 5$ then we can make coins of {$1, $2, $3, $4, $5}to purchase any item ranging from $1 till $5.

Now Manisha, being a keen observer suggested that we could actually minimize the number of coins required and gave following distribution {$1, $2, $3}. According to him any item can be purchased one time ranging from $1 to $5. Everyone was impressed with both of them. Your task is to help Manisha come up with a minimum number of denominations for any arbitrary max price in Philaland.

Source code



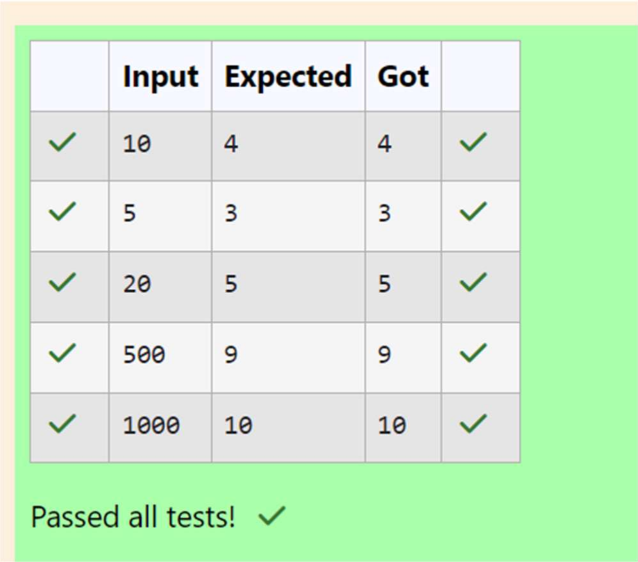
Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main()
{
    int a;
    scanf("%d",&a);
    int n=0;
    int k=1;
    while(k<=a){
        n++;
        k*=2;

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

}
```

Output



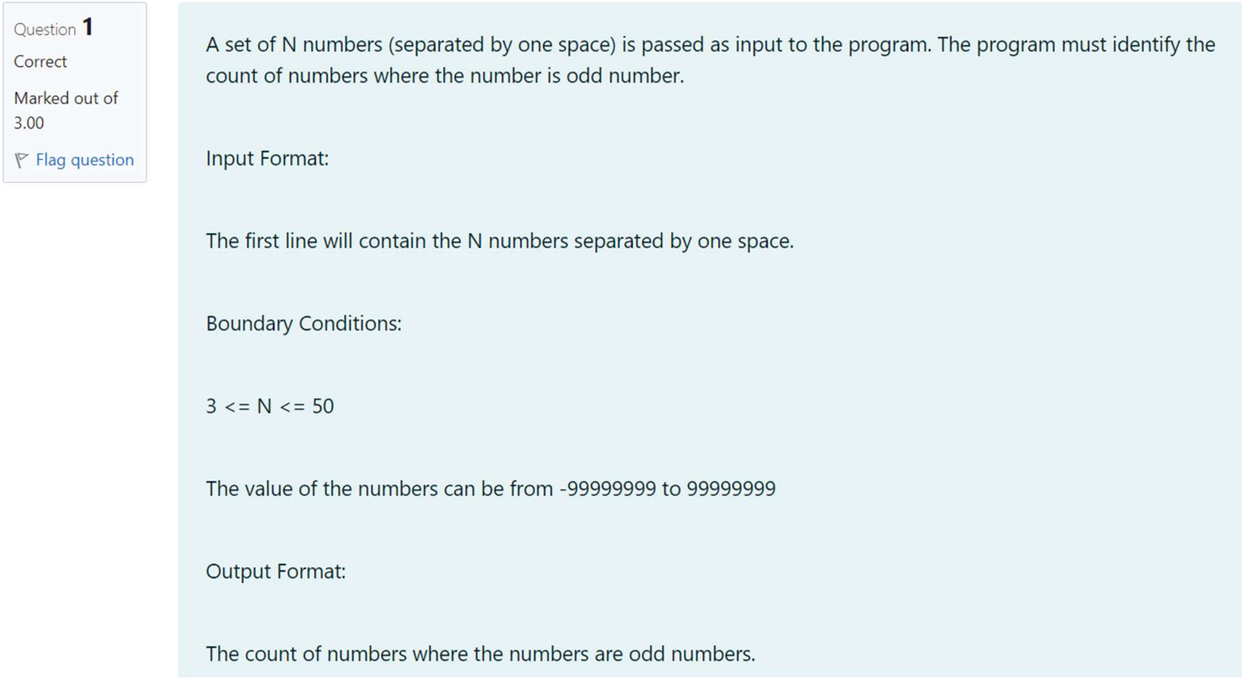
	Input	Expected	Got	
✓	10	4	4	✓
✓	5	3	3	✓
✓	20	5	5	✓
✓	500	9	9	✓
✓	1000	10	10	✓

Passed all tests! ✓

Result

The above program is executed successfully and provides the above output.

Week-04-02-Practice Session-Coding



A set of N numbers (separated by one space) is passed as input to the program. The program must identify the count of numbers where the number is odd number.

Input Format:

The first line will contain the N numbers separated by one space.

Boundary Conditions:

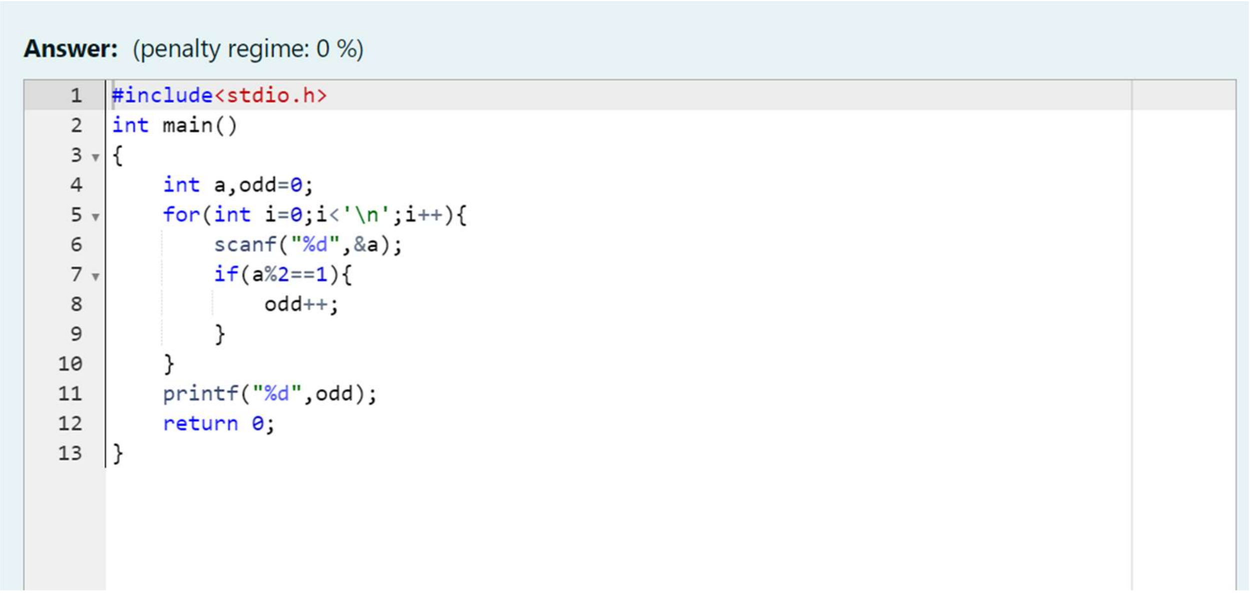
3 <= N <= 50

The value of the numbers can be from -99999999 to 99999999

Output Format:

The count of numbers where the numbers are odd numbers.

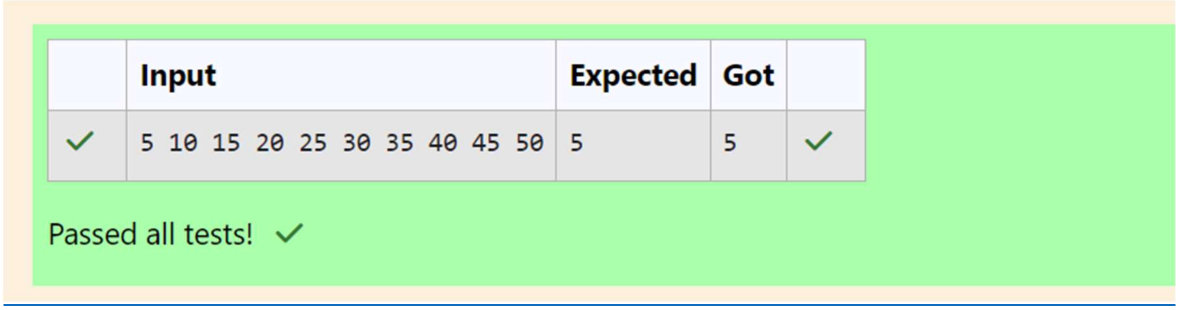
Source code



Answer: (penalty regime: 0 %)

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

Output

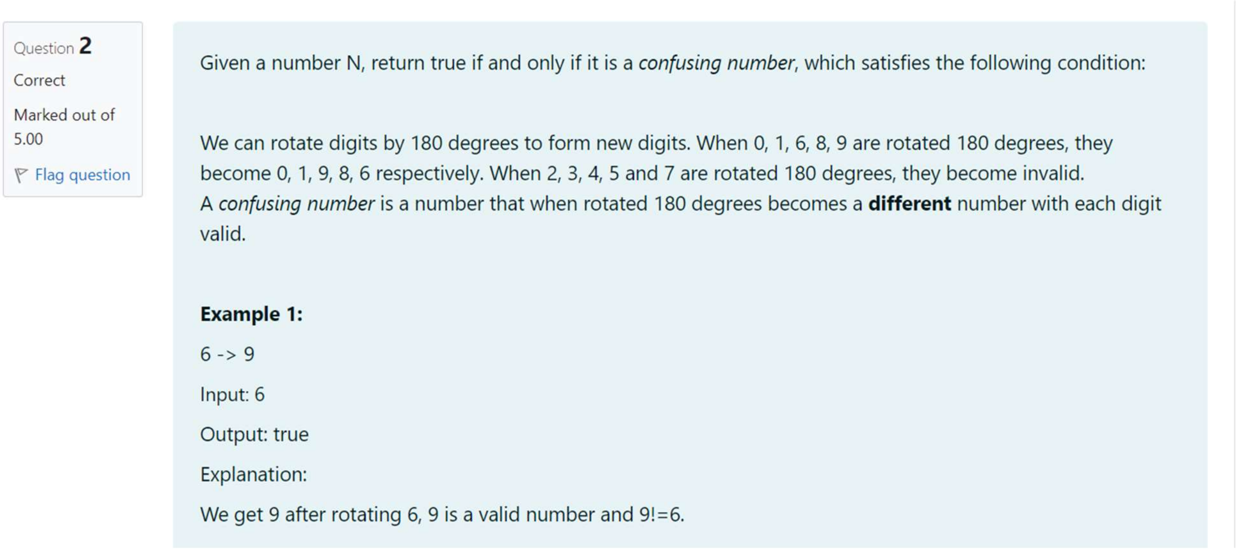


	Input	Expected	Got	
✓	`5 10 15 20 25 30 35 40 45 50`	5	5	✓

Passed all tests! ✓

Result

The above program is executed successfully and provides the above output.



Question **2**

Correct

Marked out of 5.00

Flag question

Given a number N, return true if and only if it is a *confusing number*, which satisfies the following condition:

We can rotate digits by 180 degrees to form new digits. When 0, 1, 6, 8, 9 are rotated 180 degrees, they become 0, 1, 9, 8, 6 respectively. When 2, 3, 4, 5 and 7 are rotated 180 degrees, they become invalid.
A *confusing number* is a number that when rotated 180 degrees becomes a **different** number with each digit valid.

Example 1:

6 -> 9

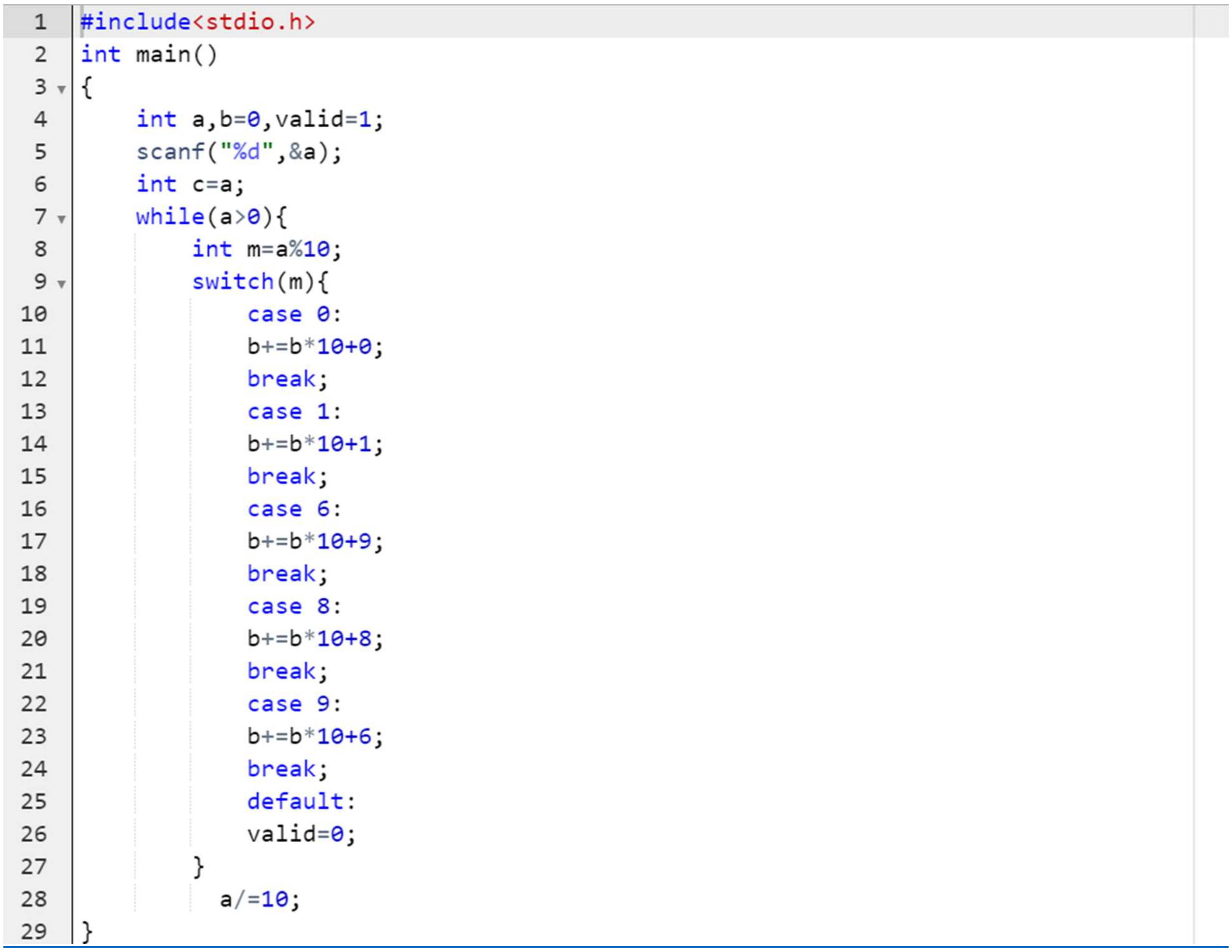
Input: 6

Output: true

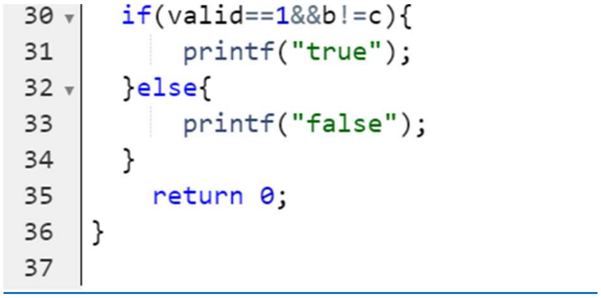
Explanation:

We get 9 after rotating 6, 9 is a valid number and 9!=6.

Source code



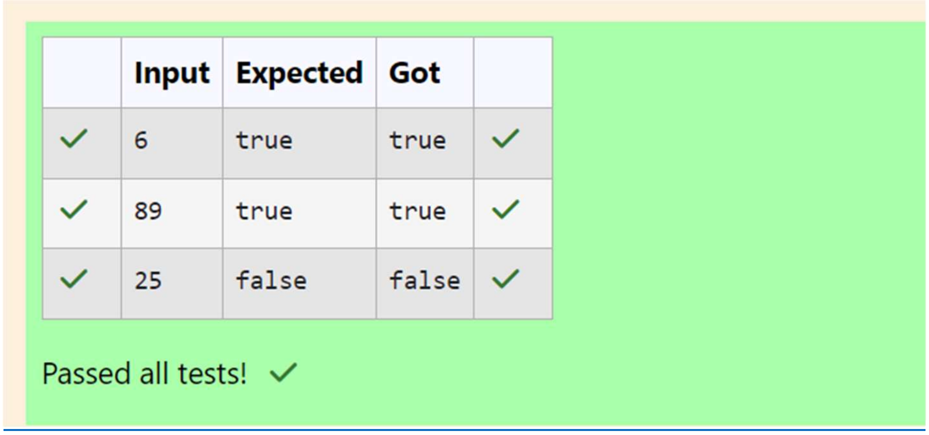
```c
#include<stdio.h>
int main()
{
    int a,b=0,valid=1;
    scanf("%d",&a);
    int c=a;
    while(a>0){
        int m=a%10;
        switch(m){
            case 0:
            b+=b*10+0;
            break;
            case 1:
            b+=b*10+1;
            break;
            case 6:
            b+=b*10+9;
            break;
            case 8:
            b+=b*10+8;
            break;
            case 9:
            b+=b*10+6;
            break;
            default:
            valid=0;
        }
          a/=10;
}
```



```c
  if(valid==1&&b!=c){
      printf("true");
  }else{
      printf("false");
  }
    return 0;
}

```

Output

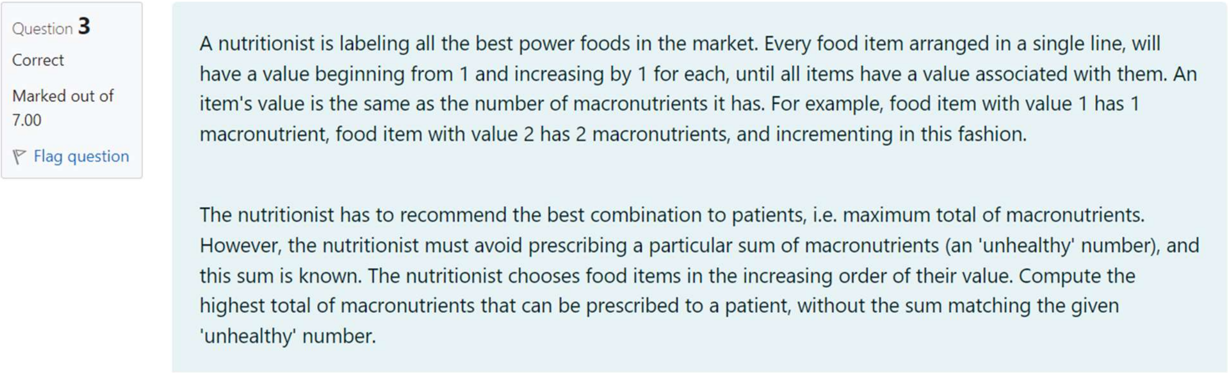


	Input	Expected	Got	
✓	6	`true`	`true`	✓
✓	89	`true`	`true`	✓
✓	25	`false`	`false`	✓

Passed all tests! ✓

Result

The above program is executed successfully and provides the above output.



Question 3

Correct

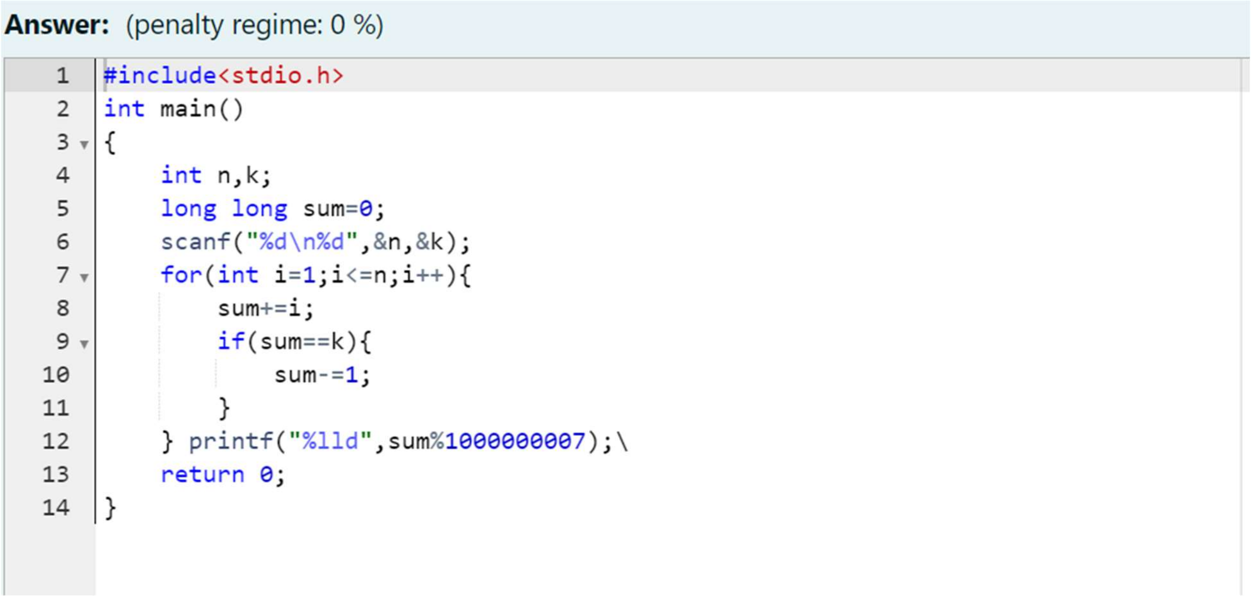
Marked out of 7.00

Flag question

A nutritionist is labeling all the best power foods in the market. Every food item arranged in a single line, will have a value beginning from 1 and increasing by 1 for each, until all items have a value associated with them. An item's value is the same as the number of macronutrients it has. For example, food item with value 1 has 1 macronutrient, food item with value 2 has 2 macronutrients, and incrementing in this fashion.

The nutritionist has to recommend the best combination to patients, i.e. maximum total of macronutrients. However, the nutritionist must avoid prescribing a particular sum of macronutrients (an 'unhealthy' number), and this sum is known. The nutritionist chooses food items in the increasing order of their value. Compute the highest total of macronutrients that can be prescribed to a patient, without the sum matching the given 'unhealthy' number.

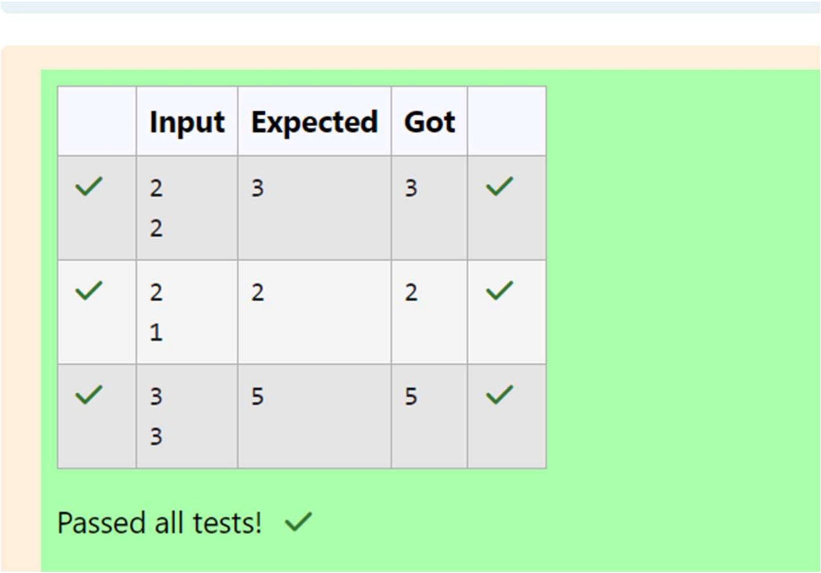
Source code



Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main()
{
    int n,k;
    long long sum=0;
    scanf("%d\n%d",&n,&k);
    for(int i=1;i<=n;i++){
        sum+=i;
        if(sum==k){
            sum-=1;
        }
    } printf("%lld",sum%1000000007);\
    return 0;
}
```

Output



	Input	Expected	Got	
✓	2 2	3	3	✓
✓	2 1	2	2	✓
✓	3 3	5	5	✓

Passed all tests! ✓

Result

The above program is executed successfully and provides the above output.