Started	Monday, 23 December 2024, 5:33 PM
Completed	Friday, 6 December 2024, 2:26 PM
Duration	17 days 3 hours

Question **1**

Correct

Marked out of 3.00

Flag question

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.

Example Input / Output 1:

Input:

5 10 15 20 25 30 35 40 45 50

Output:

5

Explanation:

The numbers meeting the criteria are 5, 15, 25, 35, 45.

Answer: (penalty regime: 0 %)

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

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

Passed all tests! ✓

5.00

Flag question

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.

Example 2:

89 -> 68

Input: 89

Output: true

Explanation:

We get 68 after rotating 89, 86 is a valid number and 86!=89.

Example 3:

11 -> 11

Input: 11

Output: false

Explanation:

We get 11 after rotating 11, 11 is a valid number but the value remains the same, thus 11 is not a confusing number.

Note:

1. 0 <= N <= 10^9
2. After the rotation we can ignore leading zeros, for example if after rotation we have 0008 then this number is considered as just 8.

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main()
{
    int a,b,c;
    scanf("%d",&a);
    while(a!=0){
    b=a%10;
    if(b==2 || b==3 || b==4|| b==3|| b==7
        c++;
    }
    a/=10;
}
   if(c>0){
      printf("false");
   }
   else{
      printf("true");
   }
   return 0;
}
```

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

Passed all tests! ✓

Sample Case 2

Sample Input For Custom Testing

Sample Input 2

3

3

Sample Output 2

5

Explanation 2

2 + 3 = 5, is the best case for maximum nutrients.

Answer: (penalty regime: 0 %)

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

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

Passed all tests! ✓

Finish review