Status	Finished
Started	Monday, 23 December 2024, 5:33 PM
Completed	Friday, 6 December 2024, 9:55 AM
Duration	17 days 7 hours

Question **1**

Correct

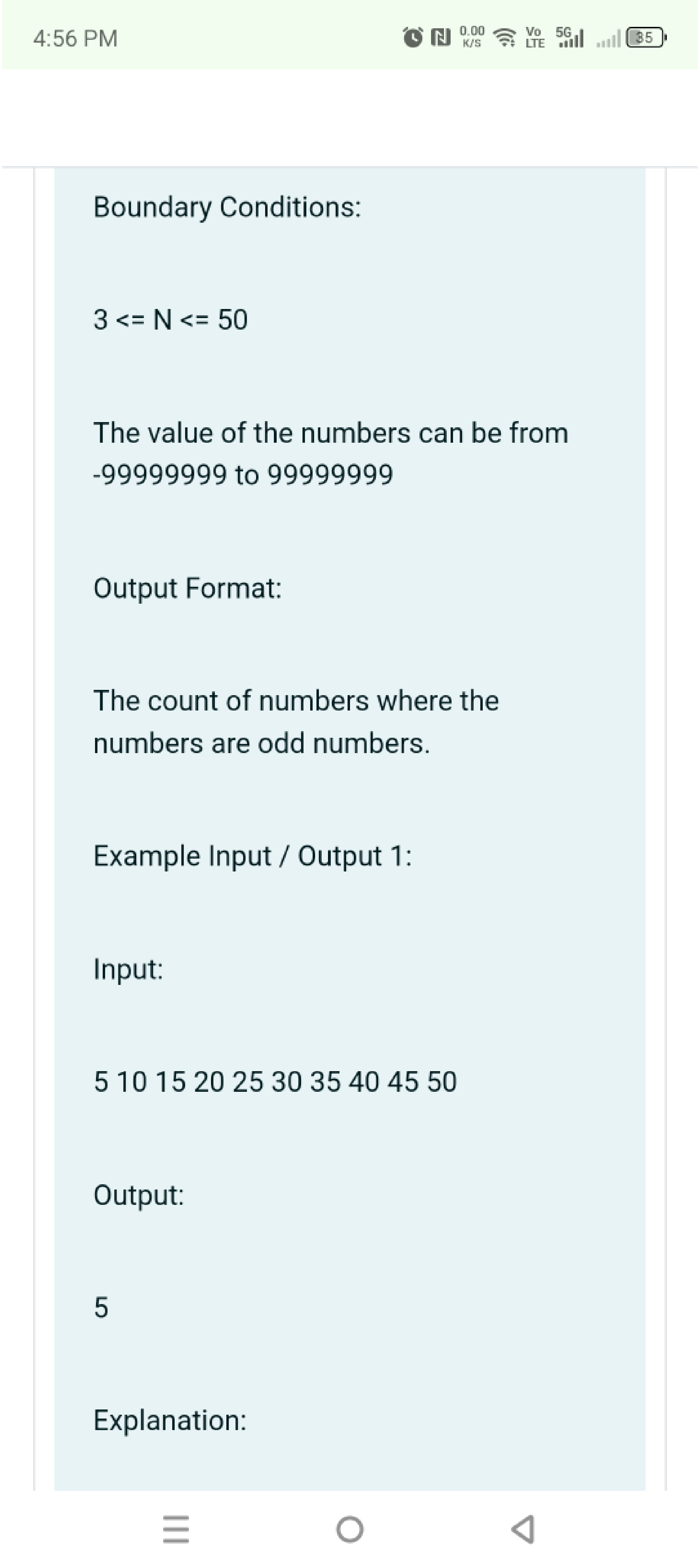
Marked out of 3.00

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:

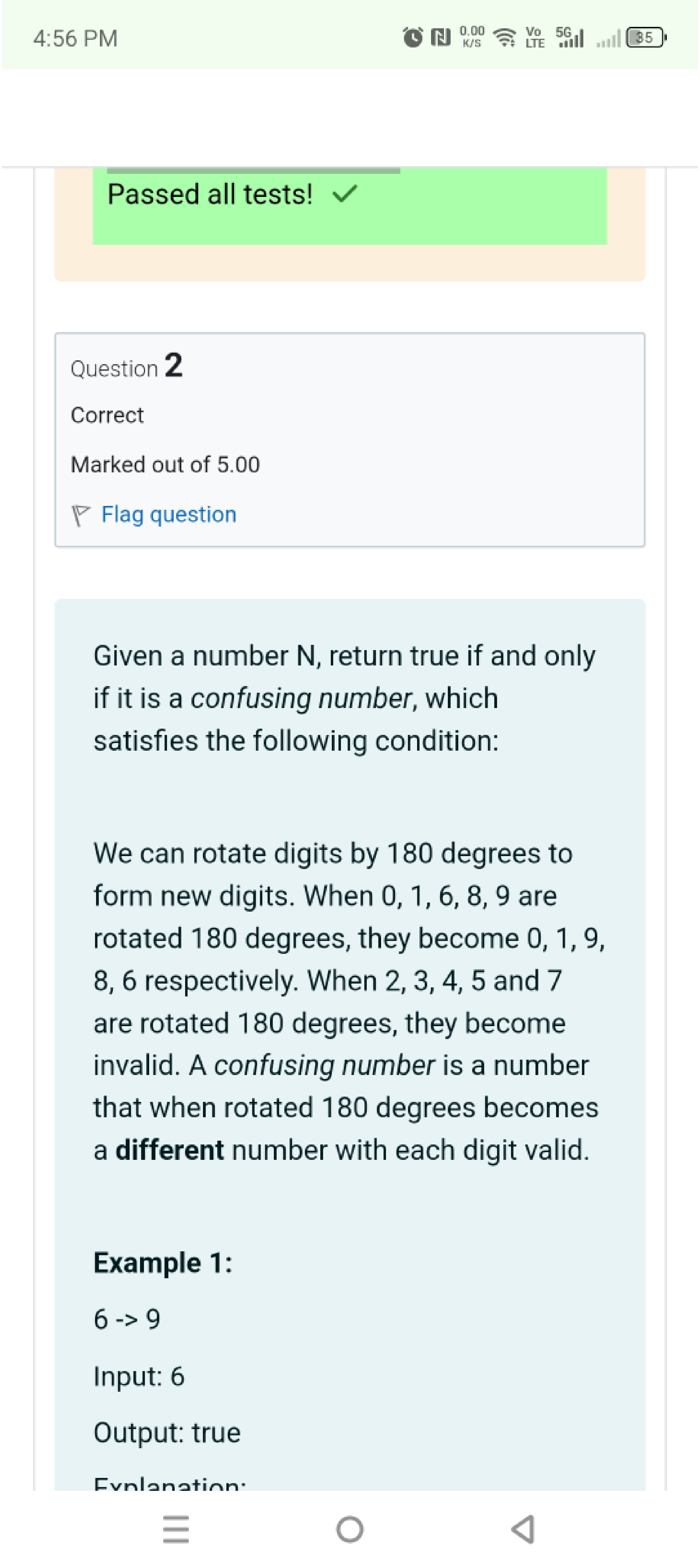
4:56 PM

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)=
    {
        if(n%2!=0)
        {
            x++;
        }
    }
    printf("%d",x);
    return 0;
}
```

	Input
✓	5 10 15 20 25 30 35 40 45 5



4:56 PM

Passed all tests! ✓

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.

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

4:56 PM

rotation we have 0008 then this number is considered as just 8.

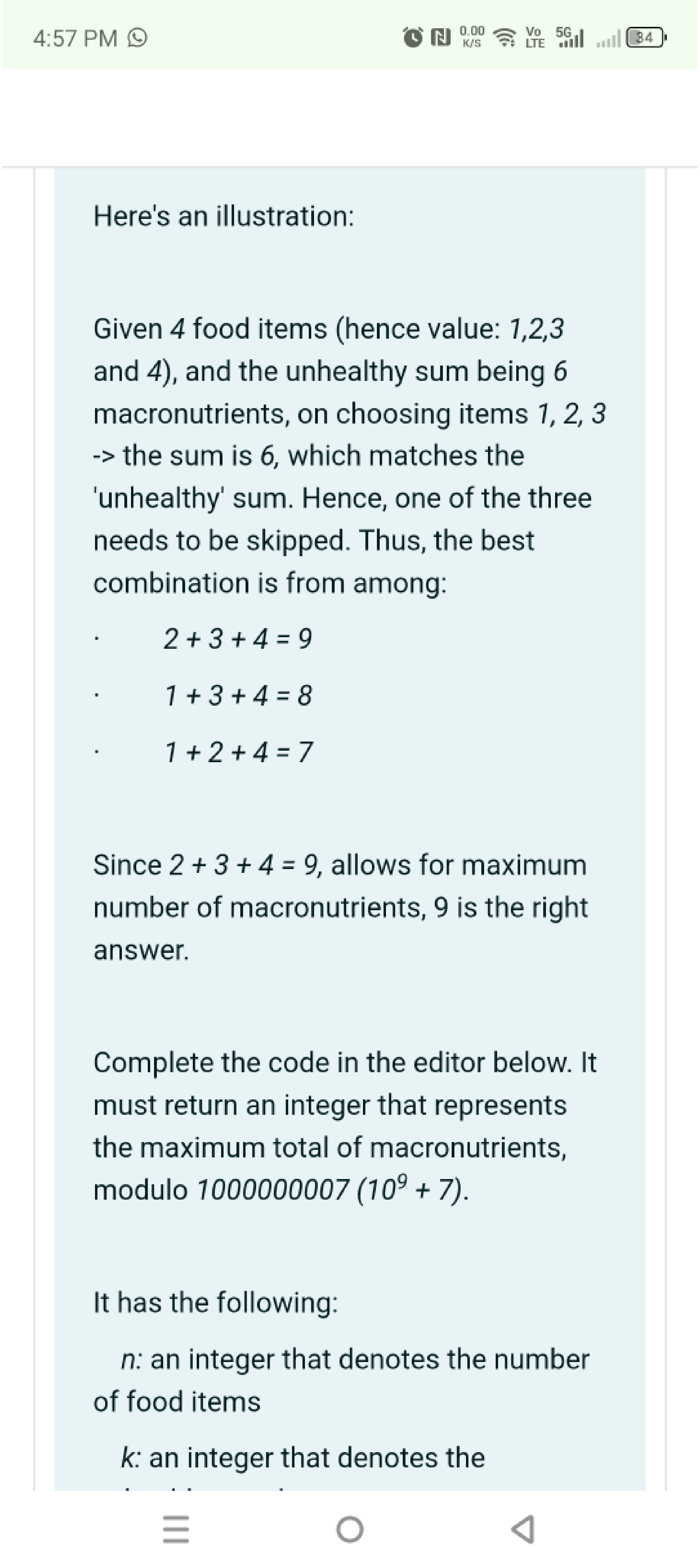
Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int n,x,y=1;
    scanf("%d",&n);
    while(n!=0&&y==1)
    {
        x=n%10;n=n/10;
        if(x==2||x==3||x=
        {
            y++;
        }
    }
    if(y==1)
    {
        printf("true");
    }
    else
    {
        printf("false");
    }
}
```

	Input	Expected	Got	
✓	6	true	true	✓
✓	89	true	true	✓

Program

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

int main()
{
	int N;
	scanf("%d", &N); // Read the number of students

	// To store the current student's name and the top scorer's name
	char studentName[101], topScorer[101];
	int maths, physics, chemistry, totalMarks, highestMarks = -1;

	// Process each student's details
	for (int i = 0; i < N; i++)
	{
		// Read name and marks
		scanf(" %[^:]:%d:%d:%d", studentName, &maths, &physics,
			&chemistry);
		// Calculate the total marks
		totalMarks = maths + physics + chemistry;

		// Check if the current student has the highest marks
		if (totalMarks > highestMarks)
		{
			highestMarks = totalMarks;
			// Update the top scorer's name
			strcpy(topScorer, studentName);
		}
	}

	// Print the name of the top scorer
	printf("%s", topScorer);

	return 0;
}
```



Here's an illustration:

Given *4* food items (hence value: *1,2,3* and *4*), and the unhealthy sum being *6* macronutrients, on choosing items *1, 2, 3* -> the sum is *6*, which matches the 'unhealthy' sum. Hence, one of the three needs to be skipped. Thus, the best combination is from among:

- *2 + 3 + 4 = 9*
- *1 + 3 + 4 = 8*
- *1 + 2 + 4 = 7*

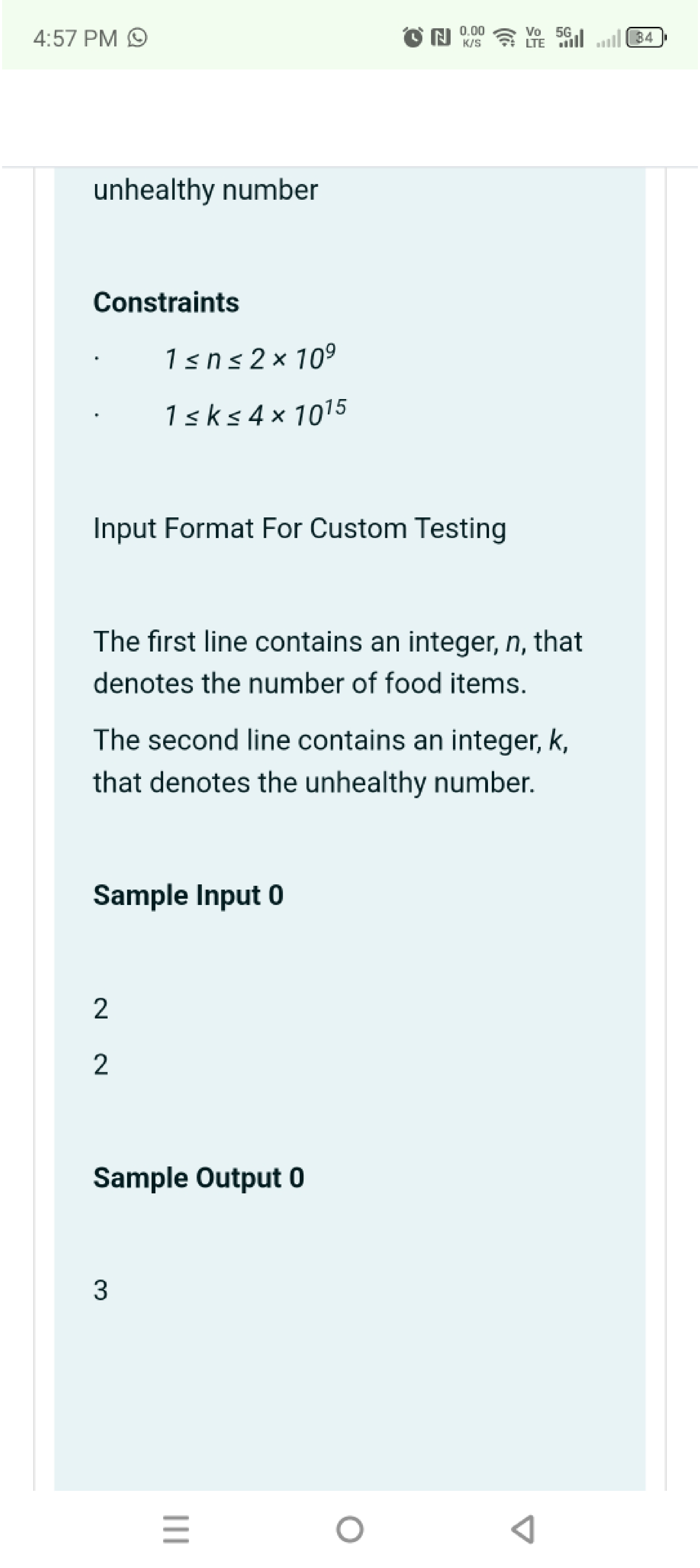
Since *2 + 3 + 4 = 9*, allows for maximum number of macronutrients, 9 is the right answer.

Complete the code in the editor below. It must return an integer that represents the maximum total of macronutrients, modulo *1000000007* ($10^9 + 7$).

It has the following:

n: an integer that denotes the number of food items

k: an integer that denotes the



unhealthy number

Constraints

- $1 \leq n \leq 2 \times 10^9$
- $1 \leq k \leq 4 \times 10^{15}$

Input Format For Custom Testing

The first line contains an integer, *n*, that denotes the number of food items.

The second line contains an integer, *k*, that denotes the unhealthy number.

Sample Input 0

```
2
2
```

Sample Output 0

```
3
```

Explanation 0

The following sequence of *n = 2* food items:

1. Item 1 has 1 macronutrients.
2. *1 + 2 = 3*; observe that this is the max total, and having avoided having exactly *k = 2* macronutrients.

Sample Input 1

```
2
1
```

Sample Output 1

```
2
```

Explanation 1

1. Cannot use item *1* because *k = 1* and *sum ≡ k* has to be avoided at any time.
2. Hence, max total is achieved by *sum = 0 + 2 = 2*.

Sample Case 2

Sample Input For Custom Testing

Sample Input 2

3

3

Sample Output 2

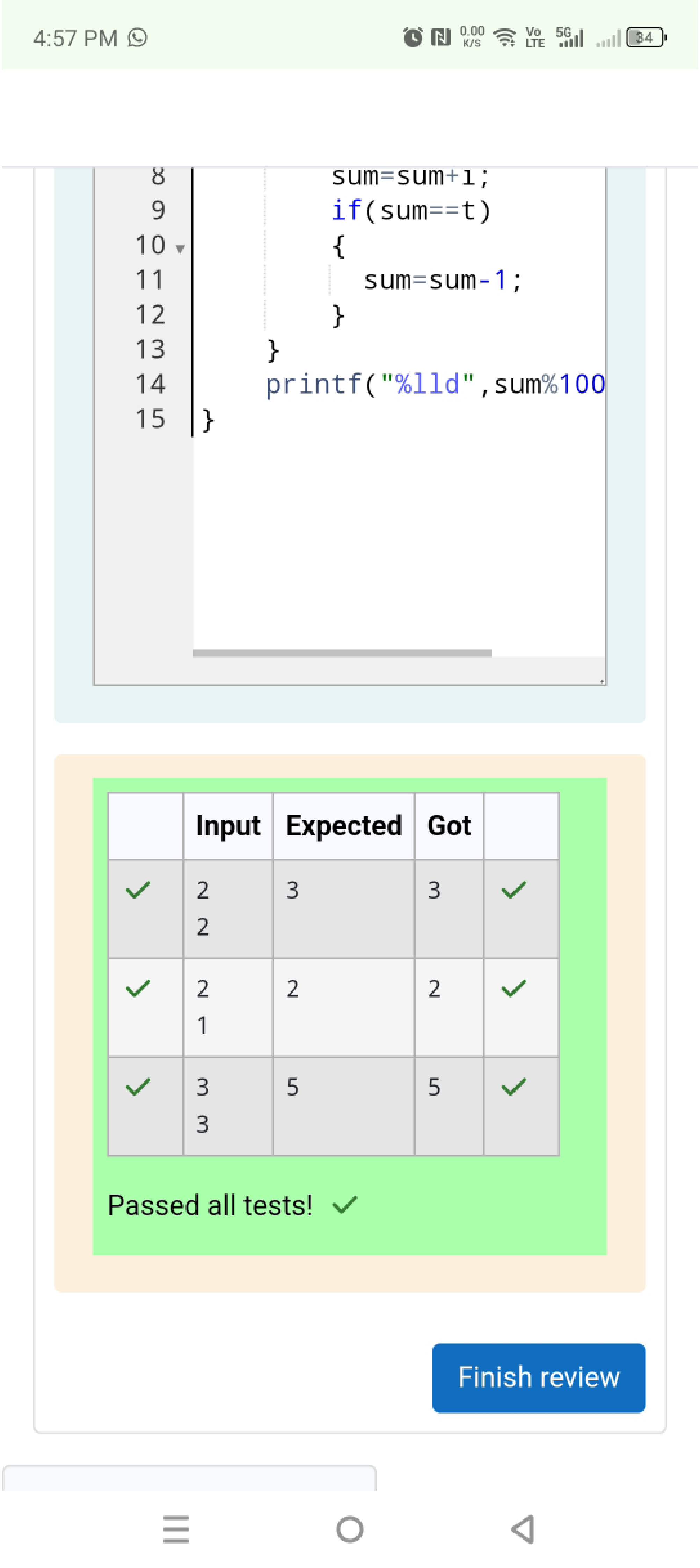
5

Explanation 2

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

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main()
{
    long long int n,t,i,s
    scanf("%lld %lld",&n
    for(i=1;i<=n;i++)
    {
```



4:57 PM

```
        sum=sum+1;
        if(sum==t)
        {
          sum=sum-1;
        }
    }
    printf("%lld",sum%100
}
```

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

Passed all tests! ✓

Finish review