

GE23131-Programming Using C-2024

Finish review

Question 1

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.

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

REC-CIS

Output Format

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

Constraints

$$1 \leq T \leq 1000$$

$$1 \leq N \leq 10000$$

Sample Input and Output

Input

3

1

6

7

REC-CIS

Output

Yes

Yes

No

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     int T;
5     scanf("%d",&T);
6     while(T--)
7     {
8         int N;
9         scanf("%d",&N);
10        if(N%8==0 || N%8==2 || N%8==5 || N%8==7)
11        {
12            printf("No\n");
13        }
14        else
15        {
```

REC-CIS

```
15 {  
16     printf("Yes\n");  
17 }  
18 }  
19 return 0;  
20 }
```

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

Passed all tests! ✓

Question 2

Correct

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.

REC-CIS

Question ↗

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.

Constraints

$1 \leq \text{num} \leq 109$

REC-CIS

Input Format For Custom Testing

There is one line of text containing a single integer num, the value to process.

Sample Input

630

Sample Output

2

Explanation

Add the holes count for each digit, 6, 3 and 0. Return $1 + 0 + 1 = 2$.

Sample Case 1

REC-CIS

Sample Input 2:

5

Sample Output 2:

3

Explanation:

For test case 1, $N=10$.

According to Manish $\{\$1, \$2, \$3, \dots \$10\}$ must be distributed.

But as per Manisha only $\{\$1, \$2, \$3, \$4\}$ coins are enough to purchase any item ranging from $\$1$ to $\$10$. Hence minimum is 4. Likewise denominations could also be $\{\$1, \$2, \$3, \$5\}$. Hence answer is still 4.

Managing Input

REC-CIS

For test case 2, $N=5$.

According to Manish $\{\$1, \$2, \$3, \$4, \$5\}$ must be distributed.

But as per Manisha only $\{\$1, \$2, \$3\}$ coins are enough to purchase any item ranging from $\$1$ to $\$5$. Hence minimum is 3. Likewise, denominations could also be $\{\$1, \$2, \$4\}$. Hence answer is still 3.

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     int n;
5     scanf("%d",&n);
6     int coins=0,max=0,deno=1;
7     while(max<n){
8         coins ++;
9         max+=deno;
10        deno=max+1;
11    }
12    printf("%d",coins);
13 }
```


REC-CIS

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

Passed all tests! ✓

Finish review

GE23131-Programming Using C-2024

Finish review

Question 1

Marked out of 3.00

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:

$3 \leq N \leq 50$

REC-CIS

 $3 \leq N \leq 50$

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

REC-CIS

Explanation:

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

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     int i,n,odd=0;
5     for(i=0;i<10;i++)
6     {
7         scanf("%d",&n);
8         if(n%2==1)
9         {
10             odd++;
11         }
12     }
13     printf("%d",odd);
14     return 0;
15 }
```

REC-CIS

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

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

REC-CIS

Output: true

Explanation:

We get 9 after rotating 6, 9 is a valid number and $9 \neq 6$.**Example 2:**

89 -> 68

Input: 89

Output: true

Explanation:

We get 68 after rotating 89, 86 is a valid number and $86 \neq 89$.**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.

REC-CIS

 $3 \leq N \leq 50$

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

REC-CIS

```
19     printf("false");
20 }
21 }
```

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

Passed all tests! ✓

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.

REC-CIS

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.

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 ($10^9 + 7$).

REC-CIS

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

REC-CIS

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

REC-CIS

2

Explanation 1

1. Cannot use item 1 because $k = 1$ and $sum \equiv k$ 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

REC-CIS

5

Explanation 2

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

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     long n,k,sum=0;
5     scanf("%ld %ld",&n,&k);
6     for(int i=1;i<=n;i++)
7     {
8         sum+=i;
9         if(sum==k)
10        {
11            sum-=1;
12        }
13    }
14    printf("%ld",sum%1000000007);
15 }
```

REC-CIS

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

Passed all tests! ✓

Finish review