


Programming Assignment Problems

 pingala.iiit.ac.in/courses/cs0-101-m24/assessments/I2c/writeup

Lab 2

Problem 1: Lucky Year

A year is considered **lucky** if all the digits in its year number are distinct. Given a current year, your task is to find the smallest **lucky** year that is strictly greater than the given year and the number of years you need to wait to experience the **lucky** year.

Input:

An integer representing the current year.

Output:

A single line containing two integers representing the next lucky year that is greater than the given year and the number of years you need to wait to experience this lucky year.

Constraints:

$1000 \leq \text{year} \leq 9000$

Good luck!

Example:

Input:

1234

Output:

1235 1

In the above example, the year 1235 is strictly greater than 1234 and also the smallest year with all distinct digits.

Input:

5555

Output:

5601 46

Problem 2: Talib and Bombs

Talib is an avid collector of bombs, and he takes great care to keep them safe. He originally had n bombs, each uniquely numbered from 1 to n . However, after a month, Talib discovers that one of his bombs is missing. Now, he only has $n-1$ bombs, each with a unique number still visible on it.

Your task is to help Talib quickly determine which bomb is missing. Given the list of the remaining bomb numbers, identify the number of the missing bomb.

If you succeed, the bomb is yours!

Input Format:

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

First line containing an integer n .

Second line containing $n-1$ space separated integers.

Output Format:

A single integer representing the number of the missing bomb.

Constraints:

- $2 \leq n \leq 10^5$
- Sum of n over all test cases does not exceed 10^5

Good luck!

Note:

Please use `bash driver.sh` to test your code locally.

Example:

Input:

```
2
10
1 2 6 9 7 4 8 5 10
5
1 5 4 3
```

Output:

```
3
2
```

In the first test case, 3 is missing from the given range 1 to 10.

In the second test case, 2 is missing from the given range 1 to 5.

Problem 3: The Quest for the Prime Gems

In the ancient city of Numerica, legends tell of a secret treasure hidden deep within the Prime Caves. This treasure is known as the "Prime Gems," mystical stones that possess the power to unlock unimaginable knowledge and wisdom. These gems, however, can only be found by those who possess a sharp mind and a deep understanding of numbers.

Your task is to help the explorers find out how many Prime Gems (prime numbers) are hidden between two specific numbers, `l` and `r`.

Input Format:

Each test case contains two integers `l` and `r`, representing the range of numbers in which the explorers need to find the Prime Gems.

Output Format:

For each test case, print a single integer representing the number of Prime Gems between `l` and `r` (inclusive).

Constraints:

$1 \leq l \leq r \leq 10^6$

Example:

Input:

10 20

Output:

4

Input:

15 30

Output:

4

Input:

1 10

Output:

4

Explanation:

Let's break down the examples:

- For the range 10 to 20, the prime numbers are 11, 13, 17, and 19. So, there are **4 Prime Gems**.
- For the range 15 to 30, the prime numbers are 17, 19, 23, and 29. So, there are **4 Prime Gems**.
- For the range 1 to 10, the prime numbers are 2, 3, 5, and 7. So, there are **4 Prime Gems**.

In each case, the task is to identify how many prime numbers exist within the specified range l to r . These prime numbers represent the "Prime Gems" that the explorers are seeking in the Prime Caves.

Are you ready to embark on this quest and discover the hidden Prime Gems? The explorers are counting on your mathematical prowess to guide them through the Prime Caves and uncover the treasures within!

Submission Guidelines

Do not rename any files given in the handout. Only write the code in the specified C files in the respective directories.