Q1, Day of the week

Write a program that calculates the day of the week for any particular date in the past or future.

Input:

The first line of input contains a single integer T denoting the number of test cases. Then T test cases follow. Each test case consist of one line. The first line of each test case consists of an integer d,m and y, d is day, m is month and y is the year.

Output:

Print day of given date.

Constraints:

1 ≤ T ≤ 100
1990 ≤ Y ≤ 2100

Example:

Input:

2

28 12 1995

30 8 2010

Output

Thursday

Monday

Q2. ANAGRAMS

Sayak is taking a cryptography class and finding *anagrams* to be very useful. We consider two strings to be anagrams of each other if the first string's letters can be rearranged to form the second string. In other words, both strings must contain the same exact letters in the same exact frequency For example, `bacdc` and `dcbac` are anagrams, but `bacdc` and `dcbad` are not.

Sayak decides on an encryption scheme involving two large strings where encryption is dependent on the minimum number of character deletions required to make the two strings anagrams. Can you help him find this number?

Given two strings, and , that may or may not be of the same length, determine the minimum number of character deletions required to make and anagrams. Any characters can be deleted from either of the strings.

Input Format

The first line contains a single string, .
The second line contains a single string, .

Constraints

-
- It is guaranteed that and consist of lowercase English letters.

Output Format

Print a single integer denoting the number of characters which must be deleted to make the two strings anagrams of each other.

Sample Input

```
cde
abc
```

Sample Output

```
4
```

Explanation

We delete the following characters from our two strings to turn them into anagrams of each other:

1. Remove `d` and `e` from `cde` to get `c`.
2. Remove `a` and `b` from `abc` to get `c`.

We had to delete characters to make both strings anagrams, so we print on a new line.

Q3. CANDY CRUSH

Sayak and Sumit love to eat candies. Sumit is currently at Sayak's mansion to watch a superhero movie when they decide to play a game. Sayak has N number of candies in a jar. The rules are simple. In each turn, a player can eat either one or four candies. Sumit is allowed to start first every

time. The player who can eat the last candy, wins. Your mission, should you choose to accept it is to find out whether Sumit 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 candies.

Output Format:

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

Constraints:

1<=T<=1000 1<=N<=10000

Sample Input:

```
3
1
5
6
```

Sample Output:

```
Yes
No
Yes
```