Super Hashing

Time Limit – 2 seconds

Hash functions are primarily used to generate fixed-length output data that acts as a shortened reference to the original data. This is useful when the original data is too cumbersome to use in its entirety. One practical use is a data structure called a hash table where the data is stored associatively. Searching linearly for a person's name in a list becomes cumbersome as the length of the list increases, but the hashed value can be used to store a reference to the original data and retrieve constant time (barring collisions).

It involves mapping an element into a numerical value using some mathematical function. In this problem we will consider a simple hash system. It involves assigning numerical value to the alphabets and summing these values of the characters.

For example, the string **"acm"** is mapped to 1 + 3 + 13 = 17. Unfortunately, this method does not give one-to-one mapping. The string "adl" also maps to 17 (1 + 4 + 12). This is called collision.

In this problem you will have to find the number of strings of length **L,** which maps to an integer **S,** using the above hash function. You have to consider strings that have only lowercase letters and uppercase letters in strictly ascending order.

***NB:** all lowercase letters are smaller than all uppercase letters: Example: 'z' < 'A'

Suppose **L** = 3 and **S** = 10, there are 4 such strings: **(abg , acf , ade , bce)**

agb also produces 10 but the letters are not strictly in ascending order.

bh also produces 10 but it has 2 letters.

Input :

The first line of input contains a integer T (<= 100) which denotes the number of test case. Each case consists of 2 integers **L and S**. **(0 < L , S < 100 000)**.

Output:

For each case, print the case number followed by the expected answer.

Sample Input	Sample Output
3 2 10 27 100 5 19	Case 1: 4 Case 2: 0 Case 3: 5