**Problem 5 – Salt and Pepper**

Chef Oliverov participates in a famous cooking show. There he cooks multiple dishes at a time, so it happens that he forgets whether he has added his favorite spices salt or pepper to the dishes. He takes the saltshaker and tastes some of the dishes he doubts they are salted. If they aren’t, he adds salt. The same he does with the pepper.

On the first line you will receive a **64 bit number** representing all the dishes Oliverov is cooking. On the next lines, you will start to receive info about the spice and the position of the dishes that have to be checked, **until you receive the command “end”**. The information will be in the following format: **[spice step],** where **spice** will be either **“salt”** or **“pepper”,** and **step** is an integer representing the position of each dish that has to be checked.

For example, if you receive “**salt 2”**, Oliverov has to check whether **every second dish** has been salted, in other words he has to check if every second bit is 1, **counting from right to left, starting from 0.** If the dish is **1(one)**, it means that it **has not been salted**, so Oliverov has to add salt and **set it to** **0 (zero).** If the command is **“pepper 3”**,Oliverov has to check **every third dish** if it has been salted, counting from **right to left, starting from 0.** If the dish is **0 (zero)**, it means that it lacks pepper, so he has to set it to **1(one).** Cooking is over when the **command “end”** is received, then the resulting number has to be printed on the console.

Spices will always be separate from the step number with a single space. You can use string.Split() to easily get each value from the command.

### Input

The input data should be read from the console.

* On the first input line you will receive a 64 bit number representing Oliverov’s dishes.
* On the next **n** lines until the command “end” you will receive commands containing the spice “salt” or “pepper” and the step at which the bit checks must be performed. Bits are counted from right to left starting from 0.

The input data will always be valid and in the format described. There is no need to check it explicitly.

### Output

The output should consist of only one line – the number representing the dishes after adding all the spices.

### Constraints

* The number representing the dishes will be a valid integer in the range [0 – 18,446,744,073,709,551,615].
* The number representing the step will be between [0 - 63].
* The number of commands will be in the range [0 - 10].
* Allowed working time for your program: 0.1 seconds. Allowed memory: 16 MB.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 25785  salt 3  end | 25776 | 25785 ->  0000000000000000000000000000000000000000000000000110010010111001  Only the bits at zero and third position need to be set to 0 (salted). So the result is:  25776 ->  0000000000000000000000000000000000000000000000000110010010110000 |
| **Input** | **Output** | **Comments** |
| 6063311135418901680  salt 2  pepper 4  end | 1238825364823814577 | 6063311135418901680 ->  0101010000100101001101010101101001010101010100000110010010110000  After the first command every second bit with value 1 is set to 0 and the result is:  9042426576511136 ->  0000000000100000001000000000101000000000000000000010000010100000  After the second command every forth bit with value 0 is set to 1  and the result is:  1238825364823814577 ->  0001000100110001001100010001101100010001000100010011000110110001 |