Problem 1

You have an initially empty bag, where you can store words, and from where you can delete words. Words can be repeated. Deleting a word means removing one of its appearances. Deleting a nonexistent word does not have any effect. At any moment you can be asked for the (lexicographically) greatest word of the bag, and how many times it is repeated. You can also be asked for the same about the smallest word.

The input consists of several lines. Each line contains either 'store w', where w is a word, or 'delete w', where w is a word, or 'maximum?', or 'minimum?'. Every word is made up of only one or more lowercase letters.

For each query, print the greatest word (or the smallest) contained in the bag at that moment. If, at the moment of answering any query, the bag is empty, tell so. Follow the format of the example.

Input	Output
minimum?	indefinite minimum
store hi	minimum: hi, 1 time(s)
minimum?	maximum: hi, 2 time(s)
delete bye	minimum: hi, 2 time(s)
store hi	minimum: bye, 1 time(s)
maximum?	indefinite maximum
minimum?	indefinite maximum
store bye	
minimum?	
delete bye	
delete hi	
delete hi	
maximum?	
delete hi	
maximum?	

Problem 2

Let us define sequences similar to those of Collatz with two parameters x and y. Given a number n, the algorithm to get the next number is:

- if n is even, we move to n/2 + x;
- otherwise, we move to 3n + y.

The standard Collatz sequence corresponds to x = 0 and y = 1.

Given x, y and a starting number n, compute the length of the cycle reached by applying the above algorithm. For example, if x = 1, y = 5 and n = 8, then the defined sequence is 8, 5, 20, 11, 38, 20, 11, 38, so the cycle has length 3.
Since numbers can become very large, and we have no mathematical guarantee that we will reach a cycle, we will stop if at some point the sequence reaches a number greater than 108.

The input consists of several cases, each with three natural numbers x, y and n. Assume that both x and y do not exceed 1000, that y is odd (for the sequence to have some interest), and that the initial n is not larger than 10^8. For every case, print the length of the cycle, or the first number that strictly exceeds 10^8.

Observation: take into account that the sequences usually reach fast a short cycle.

```
// INCLUDE ANY LIBRARY YOU MAY NEED

int max_cycle(int n, int x, int y) {
    // WRITE YOUR CODE HERE
}

int main() {
    int n, x, y;
    while (cin >> n >> x >> y) {
        cout << max_cycle(n, x, y) << endl;
    }
}
```

Problem 3

We want to build an inverted index from a sequence of words. An inverted index maps each of the words in the sequence with its positions. For example, given the following sequence of words:

house cat dog cat cat pet

its corresponding inverted index is:

- cat: 1 3 4
- dog: 2
- house: 0
- pet: 5

Write a program that reads a sequence of words and prints its corresponding inverted index in lexicografical ordered as shown in the previous example.

Problem 4

A football club does not usually have enough tickets for all its supporters, so some method is needed to distribute them: When a supporter asks for a ticket, if there is any available at that moment, the supporter immediately gets one. Otherwise, the supporters code is recorded. When a ticket is available, if there is no recorded code at that moment, the ticket is stored. Otherwise, the ticket is given to the first supporter who has asked for the ticket. Write a program to distribute tickets among supporters according to the method just described above.

The input consists of several events: T for an available ticket, and S for a supporter asking for a ticket. In the latter case, it follows the supporter's code (a non-empty string with lowercase letters). The same supporter can ask for more than one ticket.

Print the codes of the supporters who get tickets, in the order in which this happens. Print two lines with final information: count how many tickets are left and count the number of supporters who unsuccessfully asked for tickets.

Problem 4 continued on next page...

Input	Output
S abcdef	abcdef
S zyxwvu	zyxwvu
S rr	rr
S aaaaaaaa	aaaaaaaa
S aabbccdd	aabbccdd
S aabbccdc	aabbccdc
S aabbcd	
T	0 ticket(s) left
T	1 supporter(s) with no ticket
T	
T	
T	
T	
E	

BONUS: what would change if tickets, when available, are given to the supporter with the lexicographycally greatest code? And if it is given to the lexicographycally smallest code?

Problem 5

We have a queue of N robots of which only X are to be selected. Each robot has some power associated with it. There are X iterations on the queue. In each iteration, X robots are dequeued (if queue has less than X entries, all of them will be dequeued) and the one with maximum power is selected and remaining robots are enqueued back to the queue (in the order they were dequeued) but their power is decreased by one unit. If there are multiple robots with maximum power in those dequeued robots, the one which comes first in the queue is selected. If at any moment power of any robot becomes 0, it can't be decremented any further so it remains the same.

Write a program that, in each iteration, tells us the position of each selected robot in the current queue.

The first line of the input consists of two integers N and X, denoting the number of robots in the queue and the number of robots that have to be selected, respectively. Then, it follows a sequence $p_1, \ldots, p_N$ of N naturals where p_i denotes the power of robot at position i.

For each of the X iterations, output the position of the selected robot in that iteration. Position refers to the index at which the robot is present in the current queue (1 based

Problem 5 continued on next page...

indexing).

Input 1	Output 1
6 5 1 2 2 3 4 5	5 1 4 1 1

Input 2	Output 2
3 4 1 2 4	3 2 1 0

Problem 6

Write a program to keep a collection of integer numbers, perhaps with repetitions, while performing the following operations:

- S x : Stores a copy of the given number x.
- A : Asks for the greatest number.
- R : Removes the greatest number (one of them, if it is repeated).
- I x : Increases the greatest number (one of them, if it is repeated) in x units.
- D x : Decreases the greatest number (one of them, if it is repeated) in x units.

The input consists of several operations. For every query, the current greatest number. Print also an error line for each operation (except storing) when the collection is empty.

Problem 6 continued on next page...

Input	Output
A	error!
S 10 A	10
S -7 A	10
S 20 A	20
I 3 A	23
D 15 A	10
R A	8
S 30 A	30
D 37 R A	-7
R A	-7
R I 1	error!
A	error!
R	error!
D 3	error!
S 4 A	4