Counting Sort 3

In the previous challenge, it was easy to print all the integers in order, since you did not have to access the original list. Once you had obtained the frequencies of all the integers, you could simply print each integer in order the correct number of times. However, if there is other data associated with an element, you will have to access the original element itself.

In the final counting sort challenge, you are required to print the data associated with each integer. This means, you will go through the original array to get the data, and then use some "helper arrays" to determine where to place everything in a sorted array.

If you know the frequencies of each element, you know how many times to place it, but which index will you start placing it from? It will be helpful to create a helper array for the "starting values" of each element.

Challenge
You will be given a list that contains both integers and strings. In this challenge you just care about the integers. For every value i from $0 to 99$, can you output L, the number of elements that are less than or equal to i?

Input Format
- n, the size of the list ar.
- n lines follow, each containing an integer x and a string s.

Output Format
Output L for all numbers from 0 to 99 (inclusive).

Constraints
$1 \le n \le 1000000$
$0 \le x \lt 100, x \in ar$
length of string ≤ 10

Sample Input

```
10
4 that
3 be
0 to
1 be
5 question
1 or
2 not
4 is
2 to
4 the
```

Sample Output

```
1 3 5 6 9 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10
```

Explanation
0 appears *1* time, so the *0*th number is 1.
0 and *1* combined appear *3* times, so the next number is 3.
This continues for the rest of the list, until no more new numbers appear.