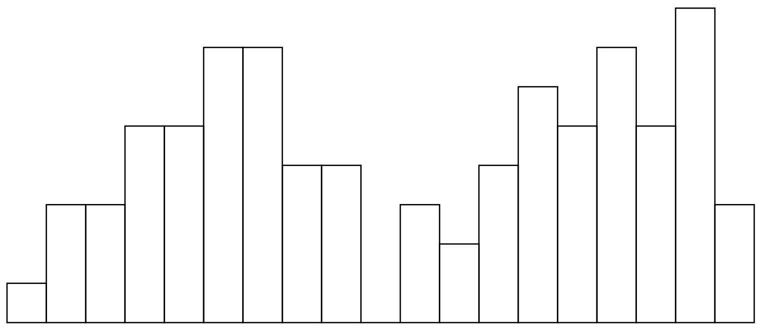
Assignment 1: Digital signals

This assignment is about digital signals: functions from some collection $\{t_0, \ldots t_{n-1}\}$ of time points to some collection of values. Here we take natural numbers $\{0, \ldots, n-1\}$ as time points, and also as the values. An example:

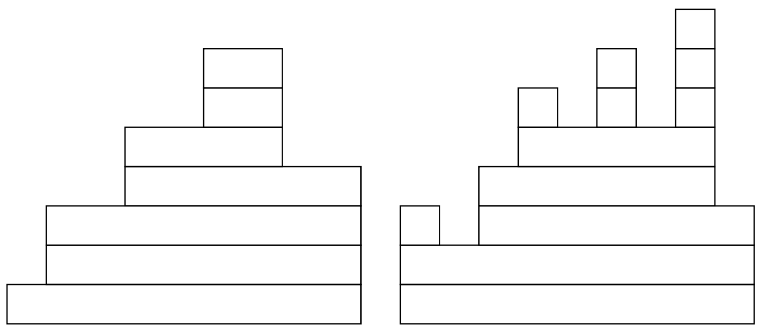
```
{1,3,3,5,5,7,7,4,4,0,3,2,4,6,5,7,5,8,3}
```

A straightforward graphical representation of a signal is



Every value of the signal corresponds with a vertical rectangle in the graph.

An alternative graphical representation consists of *layers*: horizontal rectangles. Each layer represents a maximal interval of time points where the value of the signal is at least the height of the level.



A layer can be represented as `[t1,t2)@h` where `[t1,t2)` is the halfopen interval `{t1,...,t2 - 1}` and `h` is the height of the level. So for our example the *level representation* is

```
{ [5,7)@7,   [5,7)@6,   [3,7)@5,   [3,9)@4,   [1,9)@3,   [1,9)@2,
  [0,9)@1,   [10,11)@3, [13,14)@6, [15,16)@7, [15,16)@6, [17,18)@8,
  [17,18)@7, [17,18)@6, [13,18)@5, [12,18)@4, [12,19)@3, [10,19)@2,
  [10,19)@1 }
```

The purpose of the assignment is to design a program that transforms a signal into a level representation.

Input

The input starts with a number n (with $0 \leq n \leq 1000$) indicating the number of signals. It is followed by n times the following:

a line containing the length l of the signal,
followed on the next line by a sequence of l nonnegative integers.

Output

The output gives the level representation of the input signal. The levels are separated by a space, and ordered primarily by ascending endpoints of the interval, and secondarily by descending height. This is the order used in the example above.

Example

input	corresponding **output**

`3`
`9`
`1 3 3 5 5 7 7 4 4`
`12`
`3 3 3 1 1 2 2 2 4 4 4 2`
`8`
`0 1 0 1 0 1 0 1`
	`[5,7)@7 [5,7)@6 [3,7)@5 [3,9)@4 [1,9)@3 [1,9)@2 [0,9)@1`
	`[0,3)@3 [0,3)@2 [8,11)@4 [8,11)@3 [5,12)@2 [0,12)@1`
	`[1,2)@1 [3,4)@1 [5,6)@1 [7,8)@1`

Hint

How to solve this problem efficiently? The natural idea is to read the values of the signal one by one. The question is: what do you know about the layers when you have read a new value?

- Suppose the first value you read is 2. Then you know: there are two layers that start a position 0, viz. one at height 1, one at height 2.
- Suppose the second value you read is 5. Then you know: there are three more layers, they start at position 1, and they have height 3, 4 and 5.
- Now suppose that the third value you read is 1. You know: the layers at height 5, 4, 3 and 2 have their end point at position 2. So you can output:

  ```
  [1,2)@5 [1,2)@4 [1,2)@3 [0,2)@2
  ```

  Observe that the layer at height 1 is still open: you do not know yet where it will end.
- And so on.
- When you have reached the end of the signal, you know the end point of the layers that are still open.

Now consider the linear data structures presented in Chapter 1. One of them may be very helpful in storing the information about the layers that are still open.

Extra

Design a function that, given a level representation of a signal, computes the original signal.