Problem C. Flexible Spaces

Source file name:	flexible.c, flexible.cpp, flexible.java
Input:	`standard`
Output:	`standard`

Golomb Industries is designing their new office building following modern principles that allow for flexible, reconfigurable meeting spaces. Their plans include a very wide rectangular room, with a series of optional parallel partitions.

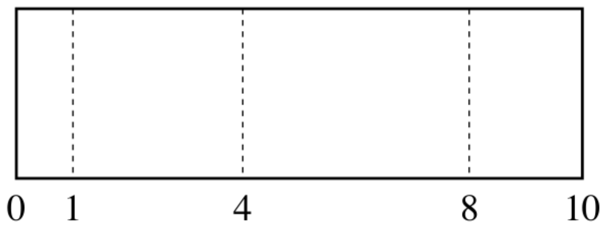

Figure C.1: An example of a configurable space

Figure C.1 illustrates such a room having a width of 10 units and three optional partitions located 1 unit, 4 units, and 8 units away from the left wall of the room. The width of the room always measures the distance between the left and right walls, and partitions always run parallel to these walls. We do not concern ourselves with the depth of the room.

For this example, if no partitions are used, a meeting can be held in the original space having width 10. If the company needs a space that is precisely 4 units wide, they can use the portion of the room between the left wall and a partition placed at location 4 (or they could use the portion between the partitions at locations 4 and 8). To provide a space having width 7 they can use the portion of the room between the partitions placed at locations 1 and 8 (omitting the partition at location 4).

Given a particular room design, your job is to determine all feasible widths for a meeting.

Input

The first line of the input contains two integers: the overall width W of the room, with $2 \leq W \leq 100$, and the number P of intermediate partitions, such that $1 \leq P < W$. Following that is a line with P integers, each designating the location L of an optional partition, such that $0 < L < W$. Each partition is at a distinct location and the partitions' locations will be listed in increasing order.

Output

Your program should output a list, from smallest to largest, of each distinct width that can be achieved for a meeting space.

Example

Input	Output

`10 3`
`1 4 8`
	`1 2 3 4 6 7 8 9 10`