TASK	GLJIVE	KOŠARKA	BRODOVI	HONI	DVONIZ	SLIKA

source code	`gljive.pas`
	`gljive.c`
	`gljive.cpp`
		`kosarka.pas`
		`kosarka.c`
		`kosarka.cpp`
			`brodovi.pas`
			`brodovi.c`
			`brodovi.cpp`
				`honi.pas`
				`honi.c`
				`honi.cpp`
					`dvoniz.pas`
					`dvoniz.c`
					`dvoniz.cpp`
						`slika.pas`
						`slika.c`
						`slika.cpp`

input	standard input (*stdin*)					
output	standard output (*stdout*)					
time limit	1 second	1 second	1 second	1 second	1 second	2 seconds
memory limit	32 MB	32 MB	32 MB	64 MB	64 MB	128 MB
point value	**30**	**50**	**70**	**100**	**120**	**130**
	500					

In front of Super Mario there are **10 mushrooms**, arranged in a row. A certain amount of points is awarded for picking each of the mushrooms. Super Mario must pick mushrooms **in order** they appear, but is not required to pick them all – his goal is to score a number of points **as close as possible to 100**.

In case there exist two such numbers which are equally close to 100 (e.g. 98 and 102), Mario will pick the **greater** one (in this case 102).

Help Super Mario and tell him how many points he will score.

INPUT

Input consists of 10 lines, each of which contains one positive integer less than or equal to 100, denoting the scores awarded for picking each mushroom, in the order that Mario can pick them in.

OUTPUT

The first and only line of output must contain the required number of points.

SAMPLE TESTS

input	input	input
10 20 30 40 50 60 70 80 90 100	1 2 3 5 8 13 21 34 55 89	40 40 40 40 40 40 40 40 40 40
output 100	**output** 87	**output** 120

Slavko has started to follow the NBA league. The game duration is exactly **48 minutes**. When a game ends, the statistics are shown. Slavko has written down whenever a team scored. He is curious about how long each team was in the lead.

INPUT

The first line of input contains one integer **N** ($1 \leq \mathbf{N} \leq 100$). The following **N** lines describe events when a team scored.

Each description consists of a team that scored, which is either **1** or **2**, and a timestamp in format **MM:SS** (minutes:seconds), the time when a team scored. Minutes and seconds are zero padded and from ranges [00, 47] and [00, 59] (inclusive). The given timestamps are unique.

OUTPUT

The first line of output must contain the duration that the first team was in the lead.

The second line of output must contain the duration that the second team was in the lead.

All durations should be in MM:SS format, with leading zeros.

SAMPLE TESTS

input	input	input
1 1 20:00	3 1 01:10 2 21:10 2 31:30	5 1 01:10 1 02:20 2 45:30 2 46:40 2 47:50
output	**output**	**output**
28:00 00:00	20:00 16:30	45:30 00:10

Mirko lives in a small town with a harbour: once in a blue moon a ship passes by. However, to this day Mirko remembers the day when **all the ships** who had ever visited the harbour showed up. He denoted this day by index **1**.

Many days have passed since, but Mirko noted each day when **at least one** ship visited the harbour, naming these days **entertaining**.

Additionally, Mirko has noticed that each ship visits the harbour periodically, at regular intervals. For instance, an interval of length 3 implies that some ship visited the harbour on days 1, 4, 7, 10 etc.

Given Mirko's list of entertaining days (including today which is considered to be an entertaining day as well), compute the **minimum** possible number of ships visiting his harbour.

Notes: All entertaining days appear on Mirko's list. It is guaranteed that the given data is consistent - in other words, a solution will always exist.

INPUT

The first line of input contains an integer **N** ($2 \leq \mathbf{N} \leq 5000$), the number of entertaining days.

The following **N** lines contain indices of entertaining days, one per line, in ascending order. The first and the last indices, representing the day from which Mirko started monitoring harbour traffic and today, respectively, will always appear on the list. The first index will always be 1, and the last one (index of today) will be less than 10^9.

OUTPUT

The first and only line of output must contain the required minimum number of ships.

SCORING

In test cases worth 70% of total points, day indices will be less than 5 000 000.

SAMPLE TESTS

input	input	input
3 1 3 4	5 1 7 10 13 19	3 1 500000000 999999999
output	**output**	**output**
2	2	1

From a pile of suggested tasks, authors of COCI must choose ones that will appear in the next round.

Difficulty of a task is described with an integer in range 1 to **N**. For some tasks, however, it's not easy to exactly determine their difficulty. Authors of COCI decided that these tasks can be considered as having either one of two consecutive difficulties. For example, some task can be considered as having difficulty of either 3 or 4.

The next round of COCI will contain exactly **N** tasks. For each difficulty, there will be exactly one task with that difficulty. Of course, no task will appear twice.

Find the number of different ways authors can choose tasks for the next round. We say that two ways are different if for some difficulty, a different task is assigned to that difficulty.

Since the expected result can be very large, output the number of ways modulo 1 000 000 007.

INPUT

The first line of input contains the integer **N** ($2 \leq \mathbf{N} \leq 100\,000$).

The second line of input contains **N** integers not greater than 10^9. **i**-th number in this line is equal to the number of tasks in a pile having difficulty exactly **i**.

The third line of input contains **N**-1 integers not greater than 10^9. **i**-th number in this line is equal to the number of tasks in a pile having difficulty either **i** or **i**+1.

OUTPUT

The first and only line of output must contain the required number of ways modulo 1 000 000 007.

SAMPLE TESTS

input	input
3 3 0 1 0 1	4 1 5 3 0 0 2 1
output	**output**
3	33

We say that a sequence of 2***K** elements is **interesting** if neither the sum of the first **K** elements, nor the sum of the last **K** elements, is greater than **S**.

A sequence **A** of length **N** is given. For every element, output the length of the longest interesting subsequence starting with that element.

INPUT

The first line contains integers **N** and **S** ($2 \leq \mathbf{N} \leq 100\,000$, $1 \leq \mathbf{S} \leq 2*10^9$).

The following **N** lines contain the sequence **A**, one integer per line. The integers are positive and their sum does not exceed $2*10^9$.

OUTPUT

Output must consist of **N** lines. **i**-th line must contain one integer, the length of the longest interesting subsequence starting with the **i**-th element. If an interesting subsequence at that position doesn't exist, output 0 (zero).

SAMPLE TESTS

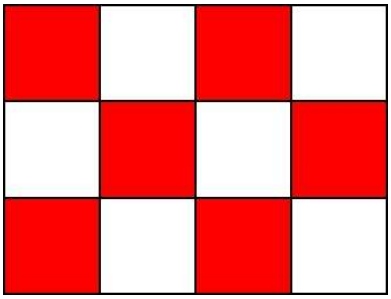
input	input	input
5 10000 1 1 1 1 1	5 9 1 1 10 1 9	8 3 1 1 1 1 1 1 1 1
output	**output**	**output**
4 4 2 2 0	2 0 0 2 0	6 6 6 4 4 2 2 0

Mirko has just installed a brand new drawing program. The program supports **K** different colours, denoted by integers from 1 to **K**. All drawing is performed on a canvas with dimensions **N**×**N**. In the beginning, all cells are white (denoted by 1).

The upper left cell of the canvas has coordinates (0, 0). The first coordinate, x, increases iterating over rows, and the second, y, increases iterating over columns.

Mirko's favourite pastime is drawing rectangular checkerboard patterns using the command **PAINT c x1 y1 x2 y2**, where **c** denotes the chosen colour, and (**x1**, **y1**) and (**x2**, **y2**) are coordinates of the upper left and lower right cells, respectively, of the rectangle being painted.

The upper left cell of the rectangle will be painted in the chosen colour, while the rest are determined by the checkerboard pattern. Cells that are not painted over by the chosen colour will **retain their previous colour**. For example, a white canvas painted over by a red checkerboard pattern will look like this:



Mirko has recently discovered two additional commands. He can save his painting at any time using the creatively named command **SAVE**, and load it again later using the command **LOAD x**, where **x** is a positive integer representing the ordinal number of the save.

Unfortunately, the program has crashed and Mirko's painting is lost forever. Luckily, Mirko has saved a log of all used commands. Can you help Mirko by reconstructing the lost painting?

INPUT

The first line of input contains three positive integers, **N** ($1 \leq \mathbf{N} \leq 1000$), **K** ($2 \leq \mathbf{K} \leq 100\,000$), and **M** ($1 \leq \mathbf{M} \leq 100\,000$, **M** is the number of commands).

Each of the following **M** lines contains one of the three described commands. Input will not contain any illegal commands.

OUTPUT

Output must consist of **N** lines, each containing **N** integers representing the colours of cells in the corresponding row of the painting.

SAMPLE TESTS

input	**input**	**input**
4 3 2 PAINT 2 0 0 3 3 PAINT 3 0 3 3 3	3 3 4 PAINT 3 0 0 1 1 SAVE PAINT 2 1 1 2 2 LOAD 1	3 4 7 PAINT 2 0 0 1 1 SAVE PAINT 3 1 1 2 2 SAVE PAINT 4 0 2 0 2 LOAD 2 PAINT 4 2 0 2 0
output	**output**	**output**
2 1 2 3 1 2 1 2 2 1 2 3 1 2 1 2	3 1 1 1 3 1 1 1 1	2 1 1 1 3 1 4 1 3