Hong Kong Olympiad in Informatics 2022/23 Junior Group

Task Overview

ID	Name	Time Limit	Memory Limit	Subtasks
J231	Bubble Tea Shop	1.000 s	256 MB	9 + 14 + 17 + 23 + 18 + 19
J232	Move Pads	1.000 s	256 MB	12 + 8 + 11 + 16 + 21 + 10 + 22
J233	Mechanical Grid	1.000 s	256 MB	13 + 39 + 48
J234	Rotating Needle	1.500 s	256 MB	15 + 17 + 22 + 25 + 21

Notice:
Unless otherwise specified, inputs and outputs shall follow the format below:

- One space between a number and another number or character in the same line.
- No space between characters in the same line.
- Each string shall be placed in its own separate line.
- Outputs will be automatically fixed as follows: Trailing spaces in each line will be removed and an end-of-line character will be added to the end of the output if not present. All other format errors will not be fixed.

C++ programmers should be aware that using C++ streams (`cin` / `cout`) may lead to I/O bottlenecks and substantially lower performance.

For some problems 64-bit integers may be required. In Pascal it is `int64`. In C/C++ it is `long long` and its token for `scanf`/`printf` is `%lld`.

All tasks are divided into subtasks. You need to pass all test cases in a subtask to get points.

J231 - BUBBLE TEA SHOP

Time Limit: 1.000 s / Memory Limit: 256 MB

Alice is a bubble tea lover. One day, she decided to open a bubble tea shop herself.

10 types of bubble tea are available in her shop. The menu is as follows:

Index	Name	Price
1	Jasmine Green Tea	$15
2	Four Season Oolong	$15
3	Black Tea	$15
4	Lemon Black Tea	$18
5	Lemon Green Tea	$18
6	Orange Green Tea	$18
7	Black Tea Latte	$20
8	Green Tea Latte	$20
9	Green Tea with Fresh Fruit	$30
10	Tieguanyin Tea with Fresh Milk	$30

Like other bubble tea shops, Alice allows customers to order with custom ice and sugar levels. However, the customer is charged an additional cost when making the customization.

For adjusting the ice level, the additional costs are listed below:

Ice Level	`Normal`	`Less Ice`	`No Ice`
Additional cost	+$0	+$2	+$3

For adjusting the sugar level, the additional costs are listed below:

Sugar Level	`100%`	`50%`	`30%`	`0%`
Additional cost	+$0	+$1	+$2	+$4

Besides that, the customers could also add extra toppings to their bubble teas. The toppings have various prices.

Toppings	Price
`Pearl`, `Pudding`	$4
`Aloe`, `Agar`	$5
`Grass Jelly`, `Lychee Jelly`, `Coffee Jelly`	$6
`Red Beans`, `Crystal Jelly Ball`	$7

In each bubble tea order, the customers can add multiple toppings. However, if the same topping has been listed in the order more than once, it should only be considered once. For example, an order with toppings [`Pudding`, `Pearl`, `Pudding`, `Red Beans`] should be considered as [`Pudding`, `Pearl`, `Red Beans`]. These toppings should cost an additional fee of $\$(4+4+7) = \$15$.

Now, Alice is given an order of bubble tea. Can you figure out the total price of the order for her?

INPUT

The first line of input contains an integer B, the index of the bubble tea ordered.

The second line of input contains a string I, the chosen ice level.

The third line of input contains a string S, the chosen sugar level.

The fourth line contains a single non-negative integer, N, the number of added toppings in the order.

If the order contains no added toppings, the input ends here. Otherwise, the input has a fifth line consisting of N strings, $T_1, T_2, \ldots, T_N$, each string representing an added topping in the order and each is separated by `, ` (a comma and a space).

OUTPUT

Output a single integer, the total price of the order.

SAMPLE TESTS

	Input	Output

1	`3`
	`Normal`
	`100%`
	`0`
		`15`

The bubble tea of index `3` costs $15.

	Input	Output

2	`7`
	`Less Ice`
	`50%`
	`0`
		`23`

The bubble tea of index `7` costs $20.
The ice level is `Less Ice`, which costs an additional fee of $2.
The sugar level is `50%`, which costs an additional fee of $1.

	Input	Output

3	`9`
	`Less Ice`
	`30%`
	`3`
	`Pudding, Pearl, Agar`
		`47`

The added toppings, `Pudding`, `Pearl`, `Agar`, cost an additional fee of $(4+4+5)=$13.

	Input	Output

4	`1`
	`No Ice`
	`100%`
	`4`
	`Agar, Pearl, Agar, Red Beans`
		`34`

The added toppings in the order should be considered as `Agar`, `Pearl`, `Red Beans`. They cost an additional fee of $(5+4+7)=$16.

SUBTASKS

For all cases:
$1 \leq B \leq 10$
I is one of `Normal`, `Less Ice` or `No Ice`
S is one of `100%`, `50%`, `30%` or `0%`
$0 \leq N \leq 30$
T_i must be one of the listed toppings in the problem description

	Points	Constraints
1	9	$I =$ `Normal` $S =$ `100%` $N = 0$
2	14	$I =$ `Normal` $N = 0$
3	17	$N = 0$
4	23	Price of each added toppings in the order $\leq \$5$ $T_1, T_2, \ldots, T_N$ are pairwise distinct
5	18	Price of each added toppings in the order $\leq \$5$
6	19	No additional constraints

J232 - MOVE PADS

Time Limit: 1.000 s / Memory Limit: 256 MB

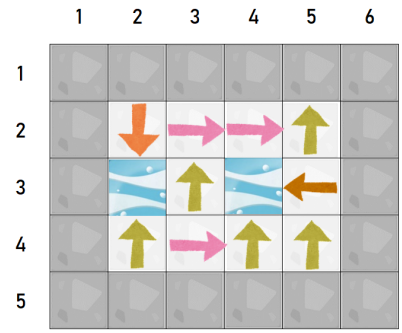
Alice has recently joined a game company as a level designer. She introduces a mechanism that can move the player around to make interesting levels. That is, Move Pads!

The level newly created by Alice is made up of a grid with $N \times M$ cells. Each cell in the grid is one of the three following types.

1. **Move Pad**: When the player is on it, it moves the player towards a pre-determined direction. [U], [D], [L], and [R] represents Move Pad that moves the player towards the four directions - up, down, left, and right respectively.
   Suppose the Move Pad is located at (x, y).
   - [U]: The player will move from (x, y) to $(x - 1, y)$.
   - [D]: The player will move from (x, y) to $(x + 1, y)$.
   - [L]: The player will move from (x, y) to $(x, y - 1)$.
   - [R]: The player will move from (x, y) to $(x, y + 1)$.
2. **Stone Floor**: When the player is on it, it stops the player. Represented by [.].
3. **Ice Floor**: When the player is moved to an Ice Floor, it preserves the movement, and continues moving the player in the original movement direction. Represented by [#].

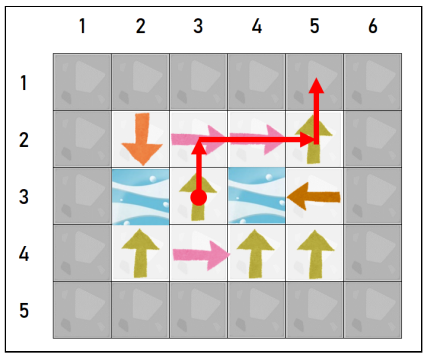
A special thing to note is, the outermost ring of the grid is always made up of Stone Floors.

An example of a valid grid is as follows.



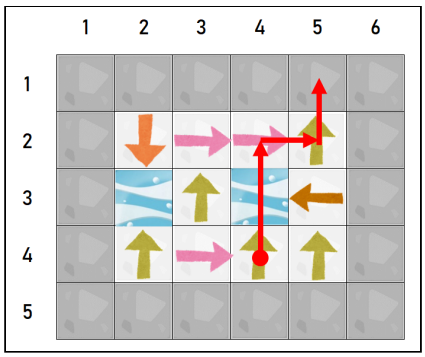
The player could **only start at a Move Pad cell**. After that, the player would either be stuck in the movement forever or end by stopping at a Stone Floor cell. The following scenarios are examples of the player starting at different Move Pad cells.

Starting at cell $(3, 3)$



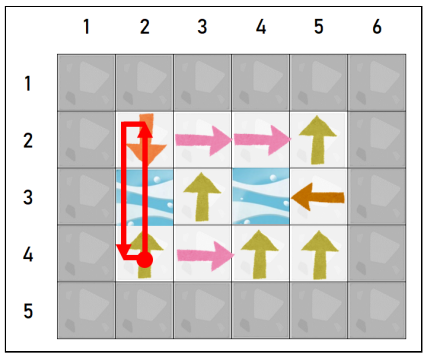
The player is moved to cell $(1, 5)$ by Move Pads.

Starting at cell $(4, 4)$



When moving from cell $(4, 4)$ to $(3, 4)$, the player's movement are preserved. The player continues to be moved up to cell $(2, 4)$.

Starting at cell $(4, 2)$



The player is stuck in movement forever.

Considering all the starting positions of the player, could you find the number of possible ending positions?

INPUT

The first line of input contains two integers N and M, the number of rows and columns in the grid.

The next N lines, each line consists of M characters.
The j^{th} character on the i^{th} line, $C_{i,j}$, represents the type of cell (i, j).

OUTPUT

Output a single integer, the number of possible ending positions.

SAMPLE TESTS

1

Input	Output
5 6 ..... .DRRU. .#U#L. .URUU.	1

This sample corresponds to the sample in the problem statement.
$(1, 5)$ is the only possible ending position.

2

Input	Output
3 7 ..... .UD##U.	3

$(1, 2)$, $(1, 6)$, and $(3, 3)$ are possible ending positions.

3

Input	Output
3 9 ..... .R##RLR#.	1

$(2, 9)$ is the only possible ending position.

4

Input	Output
5 5 ..... .U.L. ..D.. .R.U.	4

This sample satisfies the constraints of Subtask 6.
$(1, 2)$, $(2, 3)$, $(3, 4)$, and $(4, 3)$ are possible ending positions.

SUBTASKS

For all cases:
$3 \leq N, M \leq 2000$
$C_{i,j}$ is one of U, D, L, R, . and #
$C_{1,i} = C_{N,i} =$. for all $1 \leq i \leq M$
$C_{i,1} = C_{i,M} =$. for all $1 \leq i \leq N$

	Points	Constraints
1	12	$N = 3$ $C_{2,j} =$ U, D, or # for all $2 \leq j \leq M - 1$
2	8	$N = 3$ $C_{2,j} =$ L or R for all $2 \leq j \leq M - 1$
3	11	$N = 3$ $C_{2,j} =$ L, R, or # for all $2 \leq j \leq M - 1$
4	16	$3 \leq N, M \leq 40$
5	21	$C_{i,j} \neq$. for all $2 \leq i \leq N - 1, 2 \leq j \leq M - 1$
6	10	$C_{i,j} \neq$ # for all $2 \leq i \leq N - 1, 2 \leq j \leq M - 1$
7	22	No additional constraints

J233 - MECHANICAL GRID

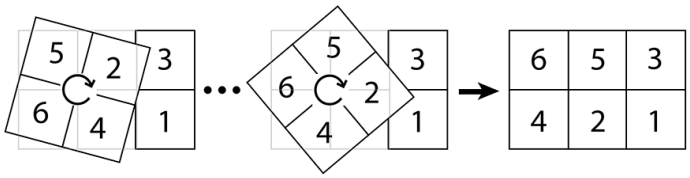
Time Limit: 1.000 s / Memory Limit: 256 MB

Bob, an engineering student, dislikes writing programs. He enjoys making mechanical models using small gears.

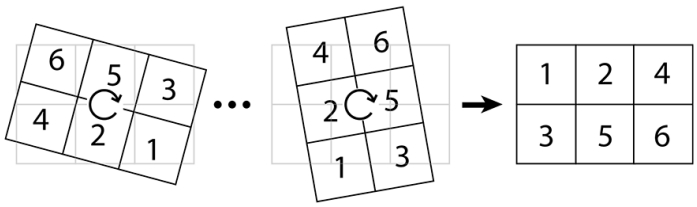
In HKOI 2022/23 Final Event, Bob proudly presents his new invention: *Mechanical Grid*. The grid consists of 2 rows and N columns, each cell of which is marked with a unique integer between 1 to $2N$ (inclusive). The cells on the first row are denoted as $(1,1),(1,2),\ldots,(1,N)$ from left to right, and the cells on the second row are denoted as $(2,1),(2,2),\ldots,(2,N)$ from left to right.

Of course, mechanical inventions involve moving objects. In the mechanical grid, all rectangular subgrids are clockwise-rotatable. That is, Bob can select a subgrid that consists of some cells forming a rectangle, and rotate that subgrid clockwise.

Here, we define a *rotation operation* as follows: select a rectangular subgrid and rotate the subgrid clockwise until the subgrid perfectly fits back onto the grid. For example, the following figure shows a rotation operation acted on the subgrid with top-left corner $(1,1)$ and bottom-right corner $(2,2)$:



The following figure shows another rotation operation acted on the subgrid with top-left corner $(1,1)$ and bottom-right corner $(2,3)$:



Alice, a computer science student who loves algorithms, is intrigued by Bob's invention. She wonders whether there is an efficient method to sort the numbers on the grid using rotation operations only. Also, she doesn't want to use too many operations.

Let $a_{i,j}$ be the number on the cell (i,j) after the rotation operations. The grid is considered sorted if and only if $a_{i,j} > a_{i,j-1}$ for all $1 \le i \le 2$, $2 \le j \le N$ and $a_{2,j} > a_{1,j}$ for all $1 \le j \le N$. Informally, the number on each cell must be greater than the numbers on the cell above it and the cell to its left (if they exist).

Can you solve Alice's problem?

INPUT

The first line of input contains a single integer N, the number of columns in the grid.

The second line of input contains N integers $A_{1,1}, A_{1,2}, \ldots, A_{1,N}$, the numbers on the cells on the first row from left to right.

The third line of input contains N integers $A_{2,1}, A_{2,2}, \ldots, A_{2,N}$, the numbers on the cells on the second row from left to right.

OUTPUT

Output an integer K on the first line, the number of rotation operations used.

K lines should follow. Each line should contain 4 integers x_1, y_1, x_2, y_2 ($1 \le x_1 \le x_2 \le 2, 1 \le y_1 \le y_2 \le N$), representing an operation acted on the subgrid with top-left corner (x_1, y_1) and bottom-right corner (x_2, y_2).

You should output the K operations in order.

SCORING

For each test case,

- You score 0% if the your sequence of operations is invalid or $K > 10^5$; otherwise
- You score 100% if $K \le \lfloor 2.5N \rfloor$; otherwise
- You score 70% if $\lfloor 2.5N \rfloor < K \le 3N$; otherwise
- You score 50% if $3N < K \le 6N$; otherwise
- You score 45% if $6N < K \le 10^5$.

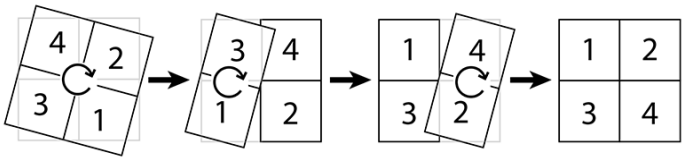
You score for each subtask is the lowest score among all test cases within that subtask.

SAMPLE TESTS

1

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

The following figure shows the rotation operations:



This solution scores 100% of the points since $K \le \lfloor 2.5N \rfloor$.

2

3 5 2 3 6 4 1	2 1 1 2 2 1 1 2 3

The two operations are illustrated by the two figures in the problem statement.
This solution scores 100% of the points since $K \le \lfloor 2.5N \rfloor$.

3

3 5 2 3 6 4 1	8 2 1 2 2 2 2 2 3 2 1 2 2 1 1 2 1 1 2 2 3 1 2 1 3 2 1 2 3 2 2 2 3

This solution scores 70% of the points since $\lfloor 2.5N \rfloor < K \le \lfloor 3N \rfloor$.

SUBTASKS

For all cases:
$1 \le N \le 200$
$1 \le A_{i,j} \le 2N$
$A_{i,j}$ are pairwise distinct

	Points	Constraints
1	13	$N = 2$
2	39	The second row consists of the numbers $N+1, N+2, \ldots, 2N$ in this order In other words, $a_{2,i} = N + i$ for all $1 \le i \le N$
3	48	No additional constraints

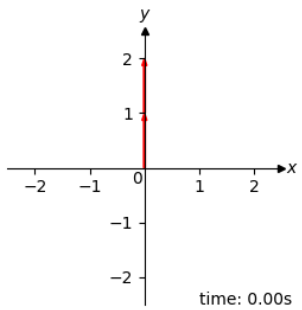
J234 - ROTATING NEEDLE

Time Limit: 1.500 s / Memory Limit: 256 MB

Mechanical grids are too complex. Here is Bob's another invention which uses simpler objects — needles.

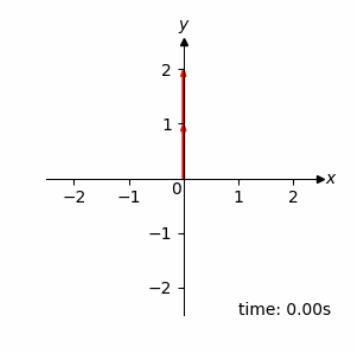
N needles are placed in a Cartesian coordinate plane. The length of each of them is 1. The needles are connected end-to-end, except for the first and the N^{th} needle. That is, the head of the i^{th} needle is connected to the tail of the $(i+1)^{th}$ needle for $1 \leq i \leq N-1$.

Initially, the needles are placed in a vertical line along the Y-axis, where the tail of the first needle is at $(0, 0)$, and the head of the N^{th} needle is at $(0, N)$. The below figure shows the initial positions of needles when $N = 2$:



Bob chooses a rotating speed for each needle. For the i^{th} needle, let S_i be the rotating speed that Bob chooses. The needle will rotate clockwise around its tail at the rate of $\frac{S_i}{10}$ degrees per second. For example, if $S_i = 225$, the needle will rotate 22.5° clockwise per second. After choosing the rotating speeds for all needles, they will start rotating at the same time. As the needles are connected end-to-end, the position of the i^{th} needle will be affected by the rotations of all previous $i-1$ needles.

By plotting the trajectory of the head of the N^{th} needle, some beautiful images may be generated when appropriate rotating speeds are chosen. Here is an example with $S_1 = 600$ and $S_2 = 1200$:



When N is large, Bob has no idea about how the final image will be. To get a better understanding of how the final image will look like, Bob wants to ask you Q questions. For the i^{th} question, he gives you a time T_i, and asks you to find the X- and Y-coordinates of the head of the N^{th} needle after T_i seconds. Could you answer all of his Q questions?

INPUT

The first line contains 2 integers N and Q, the number of needles and the number of questions.

The second line contains N integers $S_1, S_2, \ldots, S_N$, the rotating speeds of the needles.

The third line contains Q integers $T_1, T_2, \ldots, T_Q$, the times in the questions.

OUTPUT

Output Q lines, the i^{th} line should contain 2 numbers, the X- and Y-coordinates of the head of the N^{th} needle after T_i seconds.

Your output will be considered correct if the absolute or relative error between your output and the answer doesn't exceed 10^{-4}.

SAMPLE TESTS

	Input	Output
1	1 5 300 2 3 5 7 11	0.866025 0.500000 1.000000 0.000000 0.500000 -0.866025 -0.500000 -0.866025 -0.500000 0.866025
2	2 4 900 900 1 2 3 4	1.000000 -1.000000 0.000000 0.000000 -1.000000 -1.000000 0.000000 2.000000
3	2 6 600 1200 1 2 3 4 5 6	0.866025 -0.500000 0.866025 0.500000 0.000000 -2.000000 -0.866025 0.500000 -0.866025 -0.500000 0.000000 2.000000

SUBTASKS

For all cases:
$1 \leq N, Q \leq 250000$
$1 \leq S_i \leq 3600$
$1 \leq T_i \leq 10^6$

	Points	Constraints
1	15	$1 \leq N, Q \leq 1000$ $1 \leq T_i \leq 1000$ $S_i = 900, 1800, 2700,$ or 3600
2	17	$1 \leq N, Q \leq 1000$ $1 \leq T_i \leq 1000$
3	22	$S_i = 900, 1800, 2700,$ or 3600
4	25	$1 \leq Q \leq 3600$
5	21	No additional constraints

HINT

Given a length-1 needle with head located at $(0, 1)$ and tail located at $(0, 0)$ initially. If it rotates around its tail clockwise by D $(0 \leq D < 360)$ degrees, the x- and y-coordinates of its head after the rotation can be calculated by the following C++ code snippet:

```
const double PI = acos(-1);
double x = sin(PI * D / 180);
double y = cos(PI * D / 180);
```

For example, when $D = 210$, you should be able to use the above code snippet to get the same result as shown in the below figure. For the decimal representation, you can refer to line 4 of sample output 1.