Ores

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

After a mining session, Steve and Alex gather a pile of n iron, m gold and k diamonds. Steve wants to make anvils which require iron whilst Alex wants to make golden apples which require gold. However both of them want diamonds.

So, they decide to play a game taking turns taking ores from the pile one at a time, beginning with Steve. Steve can only take iron or diamonds, and Alex can only take gold or diamonds. Once an ore is taken from the pile it cannot be put back. The first player who is unable to take an ore loses. Assuming both players play optimally, determine who will win the game.

Input

The only line of input consists of three space-separated integers n, m, and k $(0 \leq n, m, k \leq 1,000,000,000)$, representing the number of iron, gold, and diamond ores respectively.

Output

Output a single line containing either `Steve` or `Alex`, whoever will win the game assuming both players play optimally.

Examples

standard input	standard output
`3 2 0`	`Steve`
`2 3 3`	`Alex`

Note

In sample case 1, there are no diamonds, so Steve always takes 1 iron on his turn and Alex always takes 1 gold on her turn. So, Steve takes 1 iron, Alex takes 1 gold, Steve takes another iron, Alex takes another gold, Steve takes the last iron, but then Alex is unable to take anything, so Steve wins.

In sample case 2, Alex can guarantee a victory regardless of Steve's choices. One possible sequence of moves is:

Turn	Steve	Alex
1	diamond	gold
2	diamond	diamond
3	iron	gold
4	iron	gold
5	nothing	

Eki Stamps

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Ruth has recently moved to Lineland. There is only one train line in Lineland, running in both directions between the West and East of the city. This train line is so famous that every station has produced unique stamps, provided to all passengers visiting that station (i.e. boarding and/or disembarking a train there). Some stations even have several different stamps, in which case a passenger can choose which stamp they want upon each visit to the station.

It has become a common hobby among train enthusiasts to collect all n different stamps. Ruth wants to take part in this tradition, starting at any one of the stations, and taking a sequence of $n-1$ train trips, each yielding another one of the remaining stamps. However she is worried that these train trips might eventually become boring. Therefore she insists that every train trip in the sequence cover at least as much distance as each of the previous trips.

Help Ruth determine whether she can collect all the stamps, and if so, what order she can do it in.

Input

The first line of input consists of a single integer n ($1 \leq n \leq 100,000$), the number of stamps.

The second line of input consists of n space-separated integers $a_1, \ldots, a_n$ ($-100,000 \leq a_i \leq 100,000$), the ith of which represents the location of the station where stamp i is available (with positive for East, and negative for West). The a_i are not necessarily distinct.

Output

Output a single line consisting of n space-separated integers $b_1, \ldots, b_n$ ($1 \leq b_i \leq n$, $b_i \neq b_j$), representing an order in which Ruth can collect the stamps (i.e. stamp b_1, then stamp b_2, ...).

If there are multiple valid orders, you may output any of them.

If there are no valid orders, output the single integer 0 instead.

Examples

standard input	standard output
4 8 4 1 2	3 4 2 1
6 2 -5 1 5 3 2	1 6 3 5 4 2

Note

In sample case 1, Ruth can collect all stamps if she starts by collecting stamp 3. Then, Ruth can collect stamp 4 (distance 1), then collect stamp 2 from there (distance $2 \geq 1$), and finally collect stamp 1 from there (distance $4 \geq 2$), completing her collection.

In sample case 2, Ruth can again collect all stamps. She starts by collecting stamp 1, then collects stamp 6 (distance 0), then collects stamp 3 from there (distance $1 \geq 0$), then collects stamp 5 from there (distance $2 \geq 1$), then collects stamp 4 from there (distance $2 \geq 2$), and finally collects stamp 2 (distance $10 \geq 2$), completing her collection.

Snakes and Ladders

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Nigel has no friends, so he has to play board games by himself. Lately, he has taken up Snakes and Ladders.

In this game, the board is a square of even side length n, divided into n^2 squares. The squares are numbered:

- left to right from 1 to n in the bottom row,
- right to left from $n+1$ to $2n$ in the second row from the bottom,
- left to right from $2n+1$ to $3n$ in the third row from the bottom,
- $\dots$,
- right to left from n^2-n+1 to n^2 in the top row.

The player's token begins on square 1. On each turn, the player rolls a six-sided die and moves their token forward by the number of squares shown on the die. If the token reaches or passes square n^2, it remains at square n^2 regardless of any subsequent die rolls.

As the name suggests, the game board also features snakes and ladders.

- A snake joins two grid squares: the higher-numbered is the head and the lower-numbered is the tail. If a die roll would take the token to the head of a snake, it is immediately moved to the tail of that snake instead.
- A ladder joins two grid squares: the higher-numbered is the top and the lower-numbered is the base. If a die roll would take the token to the base of a ladder, it is immediately moved to the top of that ladder instead.

Given the outcomes of Nigel's rolls, report the final position of his token.

Input

The first line of input consists of two space-separated integers n and m ($2 \le n \le 100$, n is even, $0 \le m \le 10,000$) representing the side length of the game board and the number of die rolls respectively.

n lines follow, representing the rows of the game board, from top to bottom. Each of these lines contains a string of n characters, representing the squares in that row. Each of these characters is either:

- a lowercase English letter, representing one endpoint of a snake,
- an uppercase English letter, representing one endpoint of a ladder, or
- the full stop character '`.`', representing an ordinary square.

Each lowercase letter and each uppercase letter is guaranteed to appear either zero or two times in the grid. Each lowercase letter and each uppercase letter appears at most once in any row of the grid. The top left and bottom left squares are guaranteed to be full stops.

One line follows, containing m space-separated integers $r_1, r_2, \dots, r_m$ ($1 \le r_i \le 6$), representing the outcomes of the die rolls.

Output

Output a single integer, the number of the square occupied by the token after all the die rolls.

Examples

standard input	standard output
4 5 ..s. ..L. L... ...s 3 4 3 5 6	15
6 10 ...A.b .b...d ..B... c...A. ..cd.. .....B 2 3 4 4 1 3 6 1 4 5	36

Note

In the first sample case, the squares are numbered as follows:

16	15	14	13
9	10	11	12
8	7	6	5
1	2	3	4

The token takes the path $1 \to 4 \to 11 \to 4 \to 9 \to 15$. In particular:

- the first roll takes the token to the tail of a snake (`'s'`), which has no further effect because snakes only transport the token from head to tail,
- the second roll takes the token to the base of a ladder (`'L'`), which immediately moves it to the top of the same ladder,
- the third roll takes the token to the head of a snake (`'s'`), which immediately moves it to the tail of the same snake,
- the fourth roll takes the token past the base of a ladder (imparting no effect) to an ordinary square, and
- the fifth roll takes the token past the head of a snake (imparting no effect) to an ordinary square.

In the second sample case, the token takes the path $1 \to 3 \to 22 \to 26 \to 9 \to 10 \to 10 \to 16 \to 33 \to 36 \to 36$. In particular:

- the third roll takes the token to the tail of a snake (`'b'`), which has no effect,
- the sixth roll takes the token from the tail of a snake (`'c'`) to the head of that same snake, which then moves it immediately back to the tail, and
- the ninth roll starts from square 33 and adds 4, which would take the token past the end of the board (square 36), so instead the token is moved to square 36 and remains there until the end of the game.

Find the Word

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Shanise is a puzzle writer. In her newest puzzle, she presents a grid of letters and asks the puzzler to find a specified word (which can be any string of letters, not necessarily a recognised English word). The word must be identified in consecutive squares of the grid, along a row, column or diagonal, making an angle of either 45° or 90° with a boundary of the grid.

Help Shanise determine the number of locations in which the word can be found in her grid.

Input

The first line of input consists of a single integer n ($1 \leq n \leq 100$), representing the side length of the grid, followed by a string s ($1 \leq |s| \leq n$) of uppercase English letters, the word to be found.

n lines follow, representing the rows of the grid, from top to bottom. Each of these lines contains a string of n uppercase English letters, representing the letters in that row.

Output

Output a single integer, the number of locations in which the word can be found in the grid.

Examples

standard input	standard output
4 BUG BAGO UUBB BOBG GEGO	1
3 XYZ XYZ YYZ ZZZ	3

Note

In the first sample case, there is only one instance of the word `BUG`.

B	A	G	O
U	U	B	B
B	O	B	G
G	E	G	O

In the second sample case, the word `XYZ` can be found in the uppermost row, in the leftmost column, and on the long diagonal.

Duels

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 256 megabytes

Jonathan and Theodore are brothers who love to play Pokémon. They each have an even number of monsters n, which are to be paired off for n simultaneous duels. Each monster has an integer-valued Combat Power (CP):

- a monster with more CP will always beat a monster with less CP in a duel, and
- a duel between two monsters with equal CP will result in a draw.

Jonathan and Theodore will record one point for each duel their monsters win, half a point for each draw, and no points for each loss.

However, there is one problem. Whichever of Jonathan and Theodore scores fewer points will cry incessantly. Their parents would be very relieved if the brothers were to conveniently each earn the same number of points.

Determine whether it is possible to organise the n duels so that Jonathan and Theodore get the same number of points.

Input

The first line of input consists of an even integer n ($2 \le n \le 5,000$).

The second line of input consists of n space-separated integers $j_1, \ldots, j_n$ ($1 \le j_i \le 100,000$), representing the combat power of Jonathan's monsters.

The third and final line of input consists of n space-separated integers $t_1, \ldots, t_n$ ($1 \le t_i \le 100,000$), representing the combat power of Theodore's monsters.

Output

If the brothers' scores can be made equal, display `Yes`. Otherwise, display `No`. Case will be ignored.

Examples

standard input	standard output
2 1 2 4 3	No
4 8 1 4 10 10 2 9 1	Yes

Note

In sample case 1, it is impossible for the brothers' score to be made equal no matter how they duel their monsters: Theodore will always score two points and Jonathan zero.

In sample case 2, the brothers can duel their monsters as follows:

	Jonathan Monster Index	Theodore Monster Index	Outcome
Duel 1	2	4	Draw ($1 = 1$)
Duel 2	3	2	Jonathan Win ($4 > 2$)
Duel 3	4	1	Draw ($10 = 10$)
Duel 4	1	3	Theodore Win ($8 < 9$)

At the end of the duels, each brother has one win and two draws, for a total of $1 + \frac{1}{2} + \frac{1}{2} = 2$ points each.