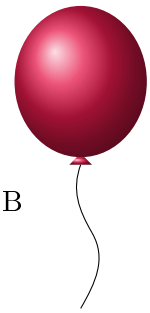
A1 Honey Heist (Subtask)



Time limit: 1.0s
Memory limit: 1024MB

Kim is playing a computer game on a grid of n by n cells. Her character is a bear, and the objective is to move the bear from its starting position to a pot of honey located elsewhere in the grid. Some cells are blocked by rocks, and it is not possible to move the bear through these rocks. There are rocks all around the perimeter of the grid.

Kim can control the bear by pressing the arrow keys ($\leftarrow$, $\uparrow$, $\rightarrow$, $\downarrow$). When she presses a key, the bear moves in that direction and cannot be stopped until it encounters a boulder. If the bear passes or stops at the honey pot, Kim wins the game.

Help Kim determine whether she can win the game, and if so, the smallest number of key presses required.

Input

The first line of input consists of two space-separated integers n and r ($0 \leq r \leq 100,000$), representing the side length of the grid and the number of rocks *inside* the grid respectively.

The second line of input consists of four space-separated integers i_b, j_b, i_h and j_h, representing the row and column of the bear's starting position and the honey pot respectively.

r lines follow, the kth of which consists of two space-separated integers i_k and j_k, representing the row and column of the kth rock.

All row and column indices are between 1 and n inclusive.

All rocks are at distinct positions. The bear's starting position and honey pot are distinct and neither of them coincide with a rock.

Subtask (50 points): $2 \leq n \leq 1,000$.

Full (50 points): $2 \leq n \leq 100,000$.

Output

If it is possible to win, print the smallest number of key presses required. Otherwise, print `0`.

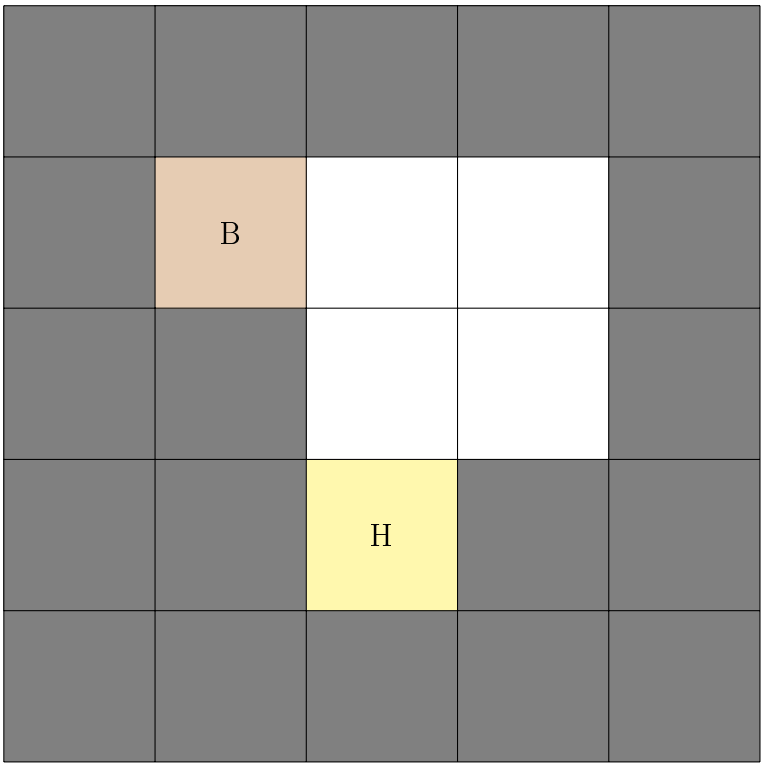
Samples

Sample input 1	Sample output 1
3 3 1 1 3 2 2 1 3 1 3 3	4

Explanation of sample 1.
The grid has three rows and three columns, and three rocks.

In the diagrams below,

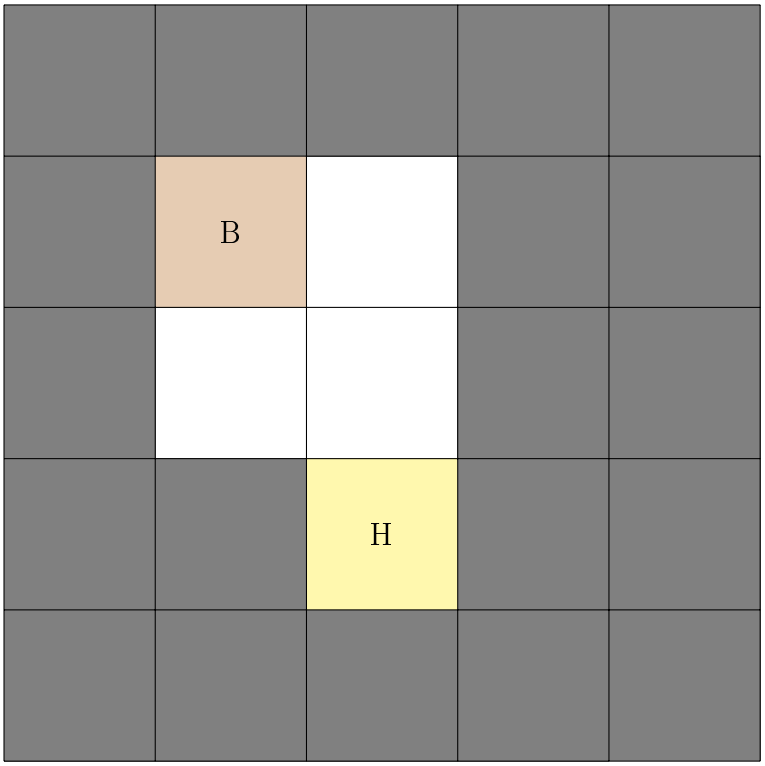
- the bear's starting position is marked in brown,
- the honey pot is marked in yellow, and
- the rocks are marked in gray.



The bear can only reach the honey pot by the following sequence of four key presses: →↓←↓.

Sample input 2	Sample output 2
3 4 1 1 3 2 3 1 1 3 2 3 3 3	2

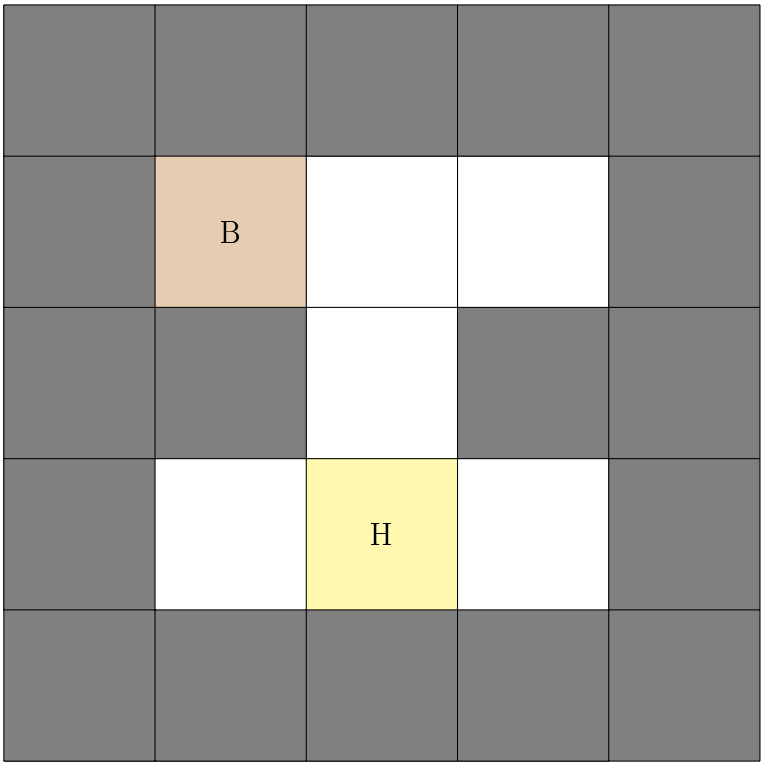
Explanation of sample 2.
The grid has three rows and three columns, and four rocks.



There are two paths to the honey pot: ↓→↓ and →↓. The second of these is preferred as it uses only two key presses.

Sample input 3	Sample output 3
3 2 1 1 3 2 2 1 2 3	0

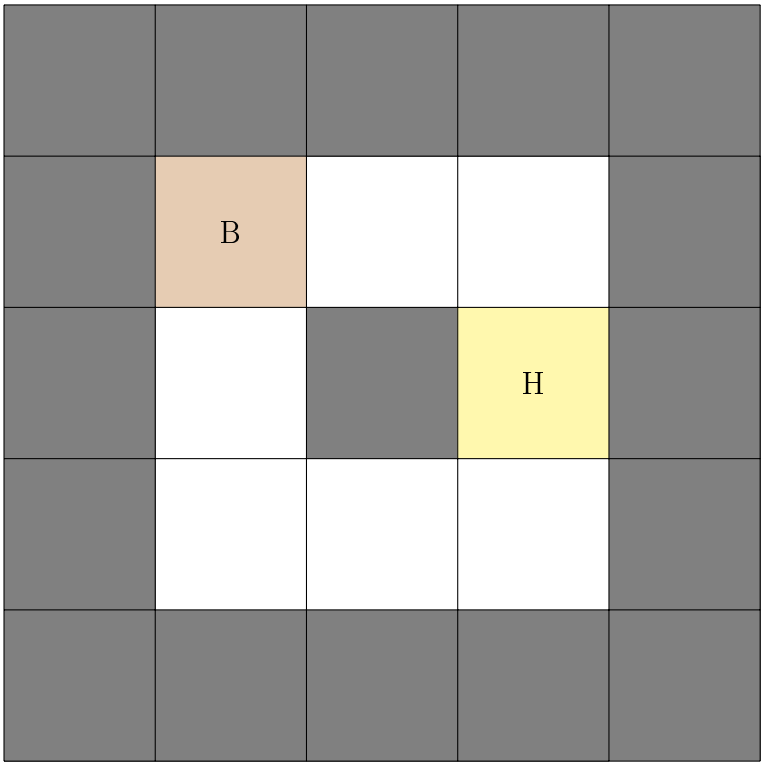
Explanation of sample 3.
The grid has three rows and three columns, and two rocks.



The bear cannot reach the honey pot, as the rocks prevent Kim from moving the bear off the top row.

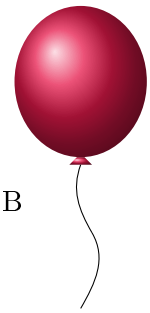
Sample input 4	Sample output 4
3 1 1 1 3 2 2 2	2

Explanation of sample 4.
The grid has three rows and three columns, and one rock.



There are two paths to the honey pot: $\downarrow\rightarrow\uparrow$ and $\rightarrow\downarrow$. The second of these is preferred as it uses only two key presses.

A2 Honey Heist (Full)



Time limit: 1.0s
Memory limit: 1024MB

Kim is playing a computer game on a grid of n by n cells. Her character is a bear, and the objective is to move the bear from its starting position to a pot of honey located elsewhere in the grid. Some cells are blocked by rocks, and it is not possible to move the bear through these rocks. There are rocks all around the perimeter of the grid.

Kim can control the bear by pressing the arrow keys ($\leftarrow$, $\uparrow$, $\rightarrow$, $\downarrow$). When she presses a key, the bear moves in that direction and cannot be stopped until it encounters a boulder. If the bear passes or stops at the honey pot, Kim wins the game.

Help Kim determine whether she can win the game, and if so, the smallest number of key presses required.

Input

The first line of input consists of two space-separated integers n and r ($0 \leq r \leq 100,000$), representing the side length of the grid and the number of rocks *inside* the grid respectively.

The second line of input consists of four space-separated integers i_b, j_b, i_h and j_h, representing the row and column of the bear's starting position and the honey pot respectively.

r lines follow, the kth of which consists of two space-separated integers i_k and j_k, representing the row and column of the kth rock.

All row and column indices are between 1 and n inclusive.

All rocks are at distinct positions. The bear's starting position and honey pot are distinct and neither of them coincide with a rock.

Subtask (50 points): $2 \leq n \leq 1,000$.

Full (50 points): $2 \leq n \leq 100,000$.

Output

If it is possible to win, print the smallest number of key presses required. Otherwise, print `0`.

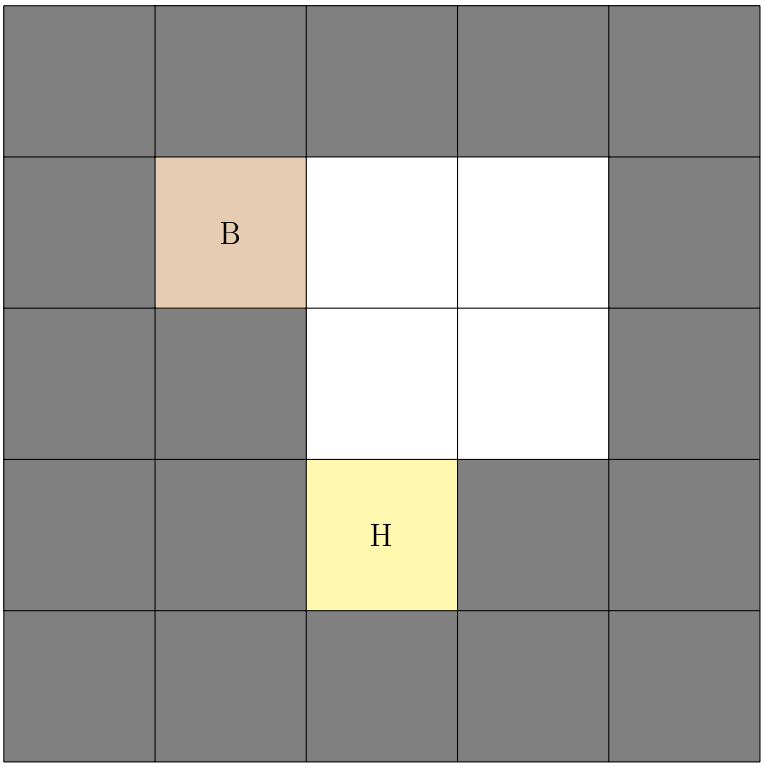
Samples

Sample input 1	Sample output 1
3 3 1 1 3 2 2 1 3 1 3 3	4

Explanation of sample 1.
The grid has three rows and three columns, and three rocks.

In the diagrams below,

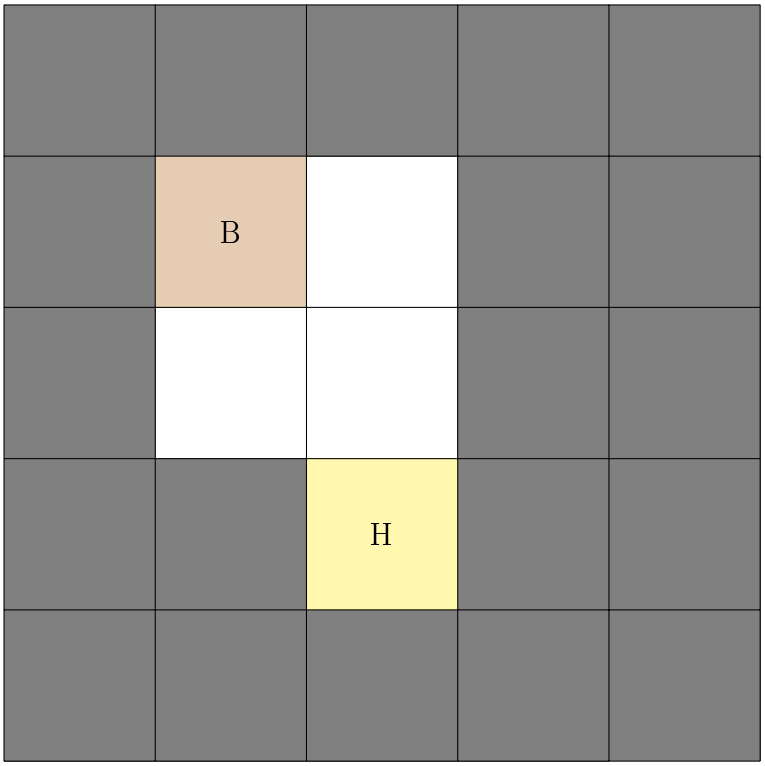
- the bear's starting position is marked in brown,
- the honey pot is marked in yellow, and
- the rocks are marked in gray.



The bear can only reach the honey pot by the following sequence of four key presses: →↓←↓.

Sample input 2	Sample output 2
3 4 1 1 3 2 3 1 1 3 2 3 3 3	2

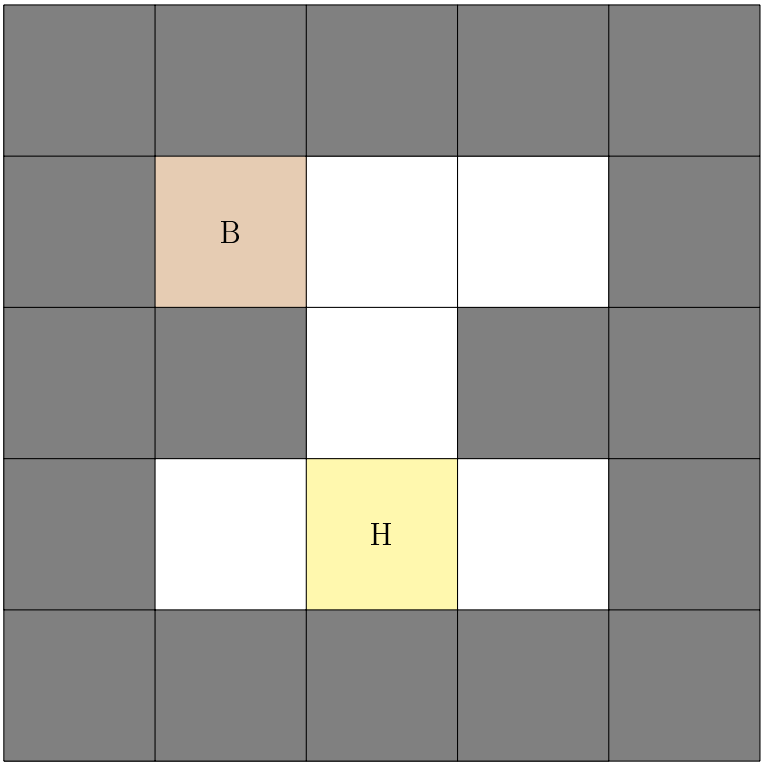
Explanation of sample 2.
The grid has three rows and three columns, and four rocks.



There are two paths to the honey pot: ↓→↓ and →↓. The second of these is preferred as it uses only two key presses.

Sample input 3	Sample output 3
3 2 1 1 3 2 2 1 2 3	0

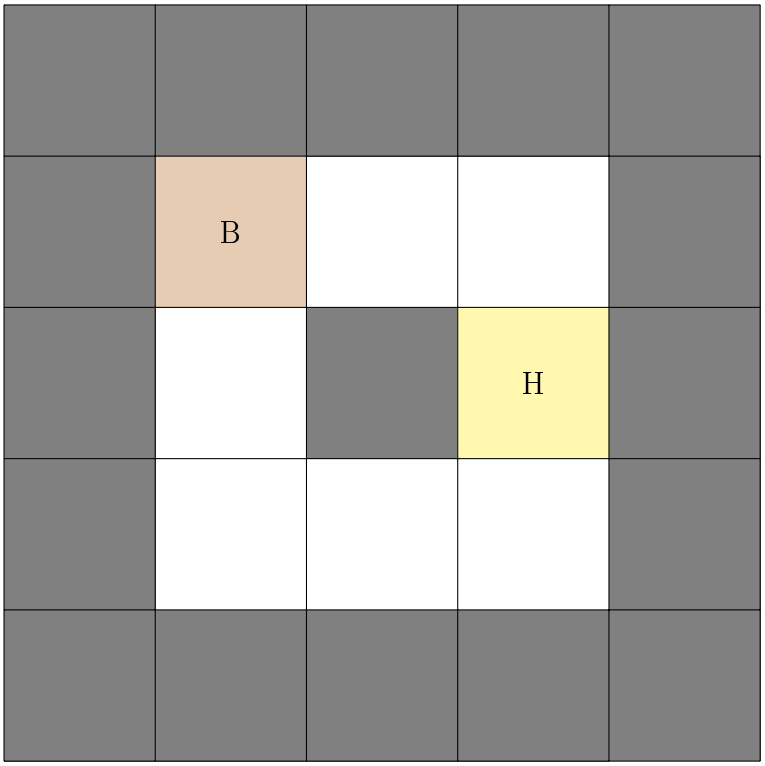
Explanation of sample 3.
The grid has three rows and three columns, and two rocks.



The bear cannot reach the honey pot, as the rocks prevent Kim from moving the bear off the top row.

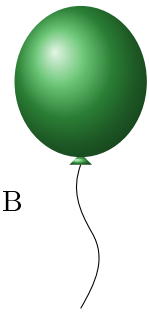
Sample input 4	Sample output 4
3 1 1 1 2 3 2 2	2

Explanation of sample 4.
The grid has three rows and three columns, and one rock.



There are two paths to the honey pot: $\downarrow\rightarrow\uparrow$ and $\rightarrow\downarrow$. The second of these is preferred as it uses only two key presses.

B1 Garden (subtask)



Time limit: 1.0s
Memory limit: 1024MB

Costa has a garden, represented by a square grid of n by n cells. Each cell is either fertile or infertile, and he wants to decorate the garden by planting one flower in each of the fertile cells.

He can buy up to n types of special flowers, which will add a variety of colours and fragrances to his garden. However, he insists that no type of special flower appears in a row or column more than once, as this would diminish how special they are. He can also buy any number of regular flowers. Costa is willing to plant any number of regular flowers in a row or column.

Costa will be charged r dollars for every regular flower he uses, and s dollars for every *type* of special flower that he uses. Help him determine the minimum cost to decorate the garden.

Input

The first line of input consists of three space-separated integers n, r and s ($0 \leq r, s \leq 10,000$).

n lines follow, each consisting of n characters. The jth character of the ith such line is either `.` or `*`, representing an infertile or fertile cell respectively.

Subtask (50 points): $1 \leq n \leq 4$.

Full (50 points): $1 \leq n \leq 50$.

Output

Print an integer, the minimum cost to plant a flower in each of the fertile cells.

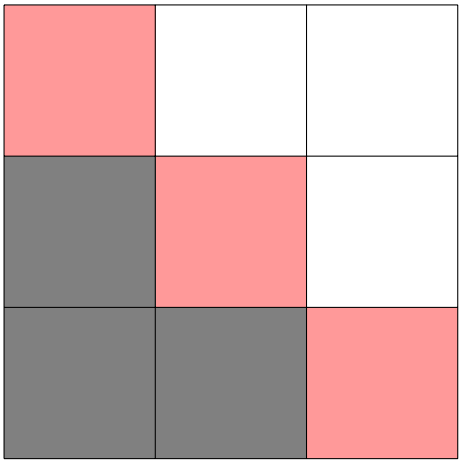
Samples

Sample input 1	Sample output 1
3 3 7 *** .** ..*	16

Explanation of sample 1.
We can place one type of special flower along the long diagonal, and three regular flowers in the top middle, top right and middle right, for a total cost of $3 \times 3 + 1 \times 7 = 16$.

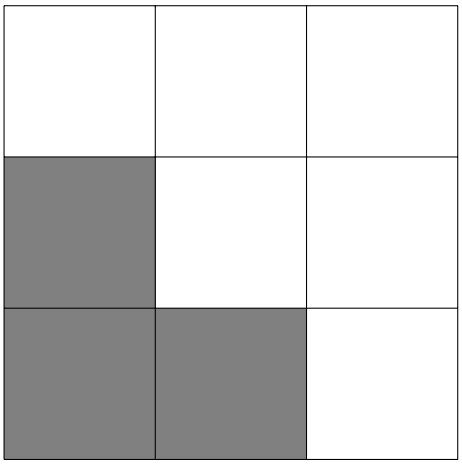
In the diagrams below:

- infertile cells are marked in gray
- cells with regular flowers are marked in white, and
- cells with special flowers are marked in other colours.



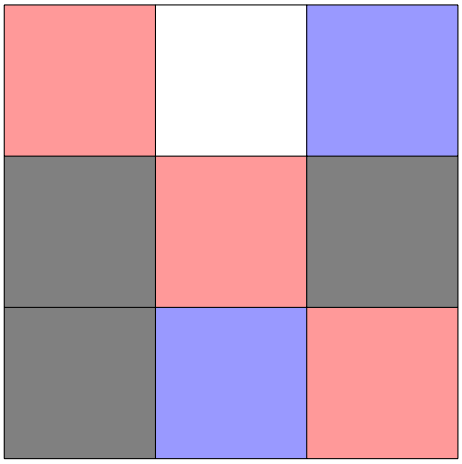
Sample input 2	Sample output 2
3 2 7 *** .** ..*	12

Explanation of sample 2.
We can place regular flowers in all six spaces, for a total cost of $6 \times 2 + 0 \times 7 = 12$.

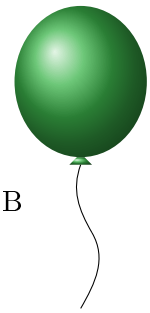


Sample input 3	Sample output 3
3 3 5 *** .*. .**	13

Explanation of sample 3.
We can use two kinds of special flowers, for a total cost of $1 \times 3 + 2 \times 5 = 13$.



B2 Garden (full)



Time limit: 1.0s
Memory limit: 1024MB

Costa has a garden, represented by a square grid of n by n cells. Each cell is either fertile or infertile, and he wants to decorate the garden by planting one flower in each of the fertile cells.

He can buy up to n types of special flowers, which will add a variety of colours and fragrances to his garden. However, he insists that no type of special flower appears in a row or column more than once, as this would diminish how special they are. He can also buy any number of regular flowers. Costa is willing to plant any number of regular flowers in a row or column.

Costa will be charged r dollars for every regular flower he uses, and s dollars for every *type* of special flower that he uses. Help him determine the minimum cost to decorate the garden.

Input

The first line of input consists of three space-separated integers n, r and s ($0 \leq r, s \leq 10,000$).

n lines follow, each consisting of n characters. The jth character of the ith such line is either `.` or `*`, representing an infertile or fertile cell respectively.

Subtask (50 points): $1 \leq n \leq 4$.

Full (50 points): $1 \leq n \leq 50$.

Output

Print an integer, the minimum cost to plant a flower in each of the fertile cells.

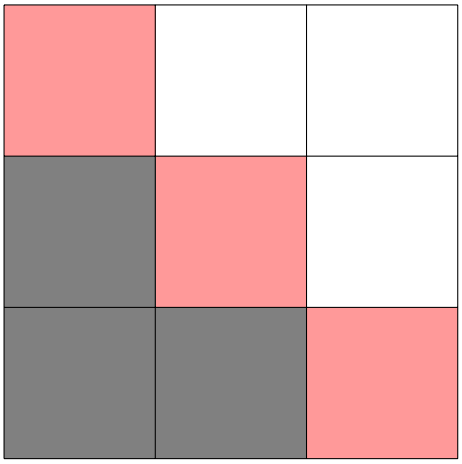
Samples

Sample input 1	Sample output 1
3 3 7 *** .** ..*	16

Explanation of sample 1.
We can place one type of special flower along the long diagonal, and three regular flowers in the top middle, top right and middle right, for a total cost of $3 \times 3 + 1 \times 7 = 16$.

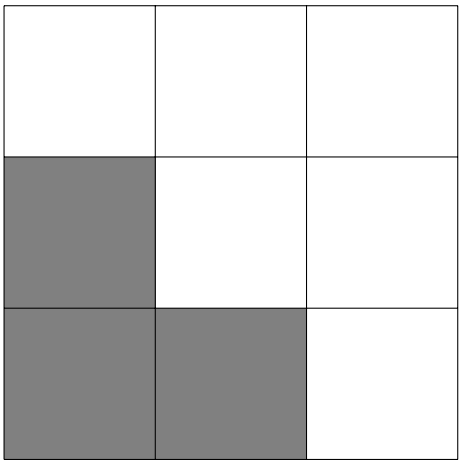
In the diagrams below:

- infertile cells are marked in gray
- cells with regular flowers are marked in white, and
- cells with special flowers are marked in other colours.



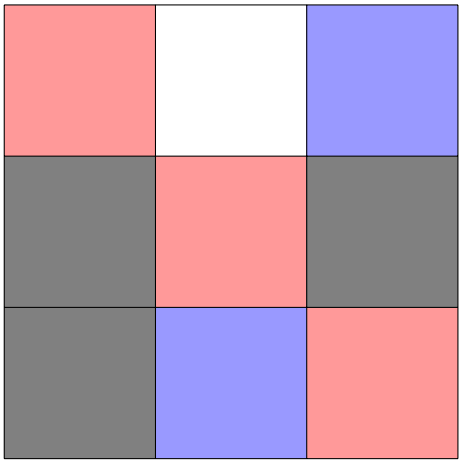
Sample input 2	Sample output 2
3 2 7 *** .** ..*	12

Explanation of sample 2.
We can place regular flowers in all six spaces, for a total cost of $6 \times 2 + 0 \times 7 = 12$.

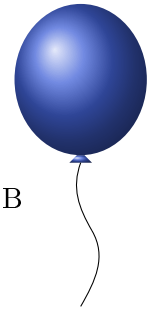


Sample input 3	Sample output 3
3 3 5 *** .*. .**	13

Explanation of sample 3.
We can use two kinds of special flowers, for a total cost of $1 \times 3 + 2 \times 5 = 13$.



C1 Medusa's Snake (Subtask)



Time limit: 3.0s
Memory limit: 1024MB

Medusa has a string of n characters, where each character is `S`, `N`, `A`, `K` or `E`.

The venom X snake is defined as the string consisting of X copies of the character `S`, followed by X of `N`, then X of `A`, then X of `K`, and finally X of `E`:

$$\underbrace{\texttt{S S}\ldots\texttt{S}}_{X}\underbrace{\texttt{N N}\ldots\texttt{N}}_{X}\underbrace{\texttt{A A}\ldots\texttt{A}}_{X}\underbrace{\texttt{K K}\ldots\texttt{K}}_{X}\underbrace{\texttt{E E}\ldots\texttt{E}}_{X}.$$

The venom level of a general string s is then the largest X such that the venom X snake is a (not necessarily contiguous) substring of s.

Medusa needs to make q edits to her string. After each of these edits, report the venom level of the string.

Input

The first line of input consists of two space-separated integers n and q.

The second line of input consists of a string of n characters, each of which is an uppercase `S`, `N`, `A`, `K` or `E`.

q lines follow, each of which contains an integer i ($1 \leq i \leq n$) and a character c ($c \in \{\texttt{S}, \texttt{N}, \texttt{A}, \texttt{K}, \texttt{E}\}$), representing a edit which changes the ith character of the string to c.

Subtask (50 points): $1 \leq n, q \leq 1,000$.

Full (50 points): $1 \leq n, q \leq 100,000$.

Output

For each of the edits to the string, print an integer on its own line, representing the updated venom level of the string. Note that you do not need to print the venom level of the original string.

Samples

Sample input 1	Sample output 1
6 4 SAAAKE 3 N 4 K 2 N 3 A	1 0 0 1

Explanation of sample 1.
The original string has six characters.

The first edit changes the string to `SANAKE`, which has the venom 1 snake `SNAKE` as a substring.

The second edit changes the string to `SANKKE`, which does not have the venom 1 snake `SNAKE` as a substring (let alone snakes of higher venom).

The third edit changes the string to `SNNKKE`, which also has venom level 0, since it does not even have any `A`s.

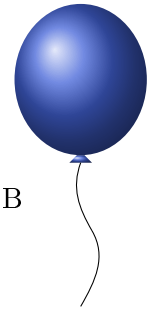
The fourth edit changes the string to `SNAKKE`, which has venom level 1.

Sample input 2	Sample output 2
20 1 NSESNNEKANKSAEKNKESE 7 E	2

Explanation of sample 2.
The only edit does not change the string, as the seventh letter is already `E`. The venom 2 snake is highlighted in red.

NSESNNEKANKSAEKNKESE

C2 Medusa's Snake (Full)



Time limit: 3.0s
Memory limit: 1024MB

Medusa has a string of n characters, where each character is `S`, `N`, `A`, `K` or `E`.

The venom X snake is defined as the string consisting of X copies of the character `S`, followed by X of `N`, then X of `A`, then X of `K`, and finally X of `E`:

$$\underbrace{\texttt{S S}\ldots\texttt{S}}_{X}\underbrace{\texttt{N N}\ldots\texttt{N}}_{X}\underbrace{\texttt{A A}\ldots\texttt{A}}_{X}\underbrace{\texttt{K K}\ldots\texttt{K}}_{X}\underbrace{\texttt{E E}\ldots\texttt{E}}_{X}.$$

The venom level of a general string s is then the largest X such that the venom X snake is a (not necessarily contiguous) substring of s.

Medusa needs to make q edits to her string. After each of these edits, report the venom level of the string.

Input

The first line of input consists of two space-separated integers n and q.

The second line of input consists of a string of n characters, each of which is an uppercase `S`, `N`, `A`, `K` or `E`.

q lines follow, each of which contains an integer i ($1 \leq i \leq n$) and a character c ($c \in \{\texttt{S}, \texttt{N}, \texttt{A}, \texttt{K}, \texttt{E}\}$), representing a edit which changes the ith character of the string to c.

Subtask (50 points): $1 \leq n, q \leq 1,000$.

Full (50 points): $1 \leq n, q \leq 100,000$.

Output

For each of the edits to the string, print an integer on its own line, representing the updated venom level of the string. Note that you do not need to print the venom level of the original string.

Samples

Sample input 1	Sample output 1
6 4 SAAAKE 3 N 4 K 2 N 3 A	1 0 0 1

Explanation of sample 1.
The original string has six characters.

The first edit changes the string to `SANAKE`, which has the venom 1 snake `SNAKE` as a substring.

The second edit changes the string to `SANKKE`, which does not have the venom 1 snake `SNAKE` as a substring (let alone snakes of higher venom).

The third edit changes the string to `SNNKKE`, which also has venom level 0, since it does not even have any `A`s.

The fourth edit changes the string to `SNAKKE`, which has venom level 1.

Sample input 2	Sample output 2
20 1 NSESNNEKANKSAEKNKESE 7 E	2

Explanation of sample 2.
The only edit does not change the string, as the seventh letter is already `E`. The venom 2 snake is highlighted in red.

NSESNNEKANKSAEKNKESE