Task 4: Tiles (`tiles`)

Having recently moved into a new house, Eustace the Sheep has decided to renovate his lavatory as he simply cannot stand the sight of its drab interior. At the moment, the toilet floor consists of a 3 by N grid of black and white squares in some initial pattern.

Eustace notices that he has a very large number of identical rectangular 1 by 2 tiles at his disposal. To preserve the aesthetic appeal of his washroom, each tile can be rotated but must be placed parallel to the walls of the toilet. Furthermore, the glue that he uses to secure the tiles in place cannot be applied to the black squares, meaning that tiles can only be placed on white squares.

Alas, the successful renovation of Eustace's bathroom is contingent on the availability of his contractor, who has unilaterally postponed their plans. In the midst of daydreaming, Eustace gazes wistfully at a section of his bathroom floor that spans columns a to b, wondering how many different patterns he can make by placing down some or none of the tiles within that region. Two patterns are considered different if two squares that share a tile in one pattern do not share a tile in the other.

Just as he has finished calculating the total number of possible patterns, he realises that some squares have been discoloured due to a variety of factors such as mould, mildew and the like. In particular, sometimes a single square in row x and column y may have its colour flipped from black to white and vice versa.

Help Eustace answer his questions by determining the number of possible patterns of tiles amid the ever-changing colour scheme of his bathroom floor!

As the answer may be large, output the remainder when the answer is divided by 1 000 000 007.

Input

Your program must read from standard input.

The first line of the input contains 2 integers N and Q denoting the length of Eustace's bathroom floor and the total number of queries and updates.

3 lines will follow, representing the initial pattern of squares. Each line contains a string of length N consisting solely of dots '.' and crosses 'x'. Here, a dot denotes a white square while a cross denotes a black square.

Q lines then follow, with each line taking on one of the following forms:

- 1 x y which indicates an update where the colour of the square at row x and column y has been flipped.
- 2 a b which represents a query for the number of patterns that can be formed if tiles are confined to columns a to b. Not placing any tiles also forms a pattern.

Output

Your program must print to standard output.

For each query, output on a new line the remainder when the number of possible patterns is divided by 1 000 000 007.

Implementation Note

As the input lengths for subtasks 2 and 4 may be very large, you are recommended to use C++ with fast input routines to solve this problem. The scientific committee does not have a solution written in Java or Python that can fully solve this problem.

C++ and Java source files containing fast input/output templates have been provided in the attachment. You are strongly recommended to use these templates.

If you are implementing your solution in Java, please name your file `Tiles.java` and place your main function inside `class Tiles`.

Subtasks

The maximum execution time on each instance is 4.0s, and the maximum memory usage on each instance is 1GiB. For all testcases, the input will satisfy the following bounds:

- $1 \leq N, Q \leq 30000$
- $1 \leq x \leq 3$
- $1 \leq y \leq N$
- $1 \leq a \leq b \leq N$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	**Points**	**Additional Constraints**
1	17	$1 \leq N, Q \leq 8$
2	23	There will never be any black squares.
3	26	$1 \leq N, Q \leq 7000$
4	34	-

Sample Testcase 1

This testcase is valid for subtasks 1, 3, and 4 only.

Input	Output
`4 5`	`11`
`.x.x`	`3`
`xx..`	`3`
`...x`	`1`
`2 1 4`	
`2 3 3`	
`1 2 3`	
`2 1 4`	
`2 3 3`	

Sample Testcase 1 Explanation

Using −− to denote a tile, the 11 patterns for the first query are:

```
.x.x    .x.x    .x.x    .x.x
xx..    xx..    xx--    xx|.
...x    .--x    --.x    ..|x

.x.x    .x.x    .x|x    .x|x
xx..    xx--    xx|.    xx|.
--.x    .--x    .--x    ...x

.x.x    .x.x    .x|x
xx--    xx|.    xx|.
...x    --|x    --.x
```

For the second query, tiles are restricted to column 3. These are the 3 patterns:

```
.    .    |
.    |    |
.    |    .
```

After the first update, the bathroom floor will look like this:

```
.x.x
xxx.
...x
```

There are only 3 possible patterns now for the third query:

```
.x.x    .x.x    .x.x
xxx.    xxx.    xxx.
...x    --.x    .--x
```

For the last query, no tile can be placed in column 3 alone. There is only one pattern, the current one.

Sample Testcase 2

This testcase is valid for subtasks 1, 3, and 4 only.

Input	Output
`2 1`	`7`
`..`	
`..`	
`xx`	
`2 1 2`	

Sample Testcase 2 Explanation

Using $--$ to denote a tile, the 7 patterns are:

```
..    ..    .|    --    |.    --    ||
..    --    .|    ..    |.    --    ||
```

Sample Testcase 3

This testcase is valid for subtasks 2, 3, and 4 only.

Input	Output
`14 2`	`47177097`
`..............`	`254767228`
`..............`	
`..............`	
`2 2 11`	
`2 1 14`	