Colorful Garden

Objective

Give practice with backtracking

Story

You are ready for the tournament. The only thing left to do is to arrange the garden as a backdrop to the contest. You hired a research firm to determine the optimal garden for maximal aesthetics. The only problem is they did not send the image for the layout, they used a compress manner to represent the data.

The garden itself is a rectangular grid of plants. Each plant will have either red or black flowers. The firm sent you a description of each row and each column. In their row and column description they sent the size of the groups of black flowered plants. A group of flowers is a consecutive run of plants with the same flower color. For example if a row contains the flowers, R, R, R, B, B, R, B, R, then the firm's description for the row would be 2 1. NOTE ORDER MATTERS. The order goes from left to right in the rows and from top to bottom in the columns.

It's almost time for the opening ceremonies, and you don't have the time to play games. Write a program to determine the firm's arrangement.

Problem

Given a list of row descriptions and column descriptions determine a possible arrangement of flowers that adheres to the constraints.

Input

Input will begin with a line containing 2 integers, ***r*** and ***c*** (1 ≤ ***r***, ***c*** ≤ 15), representing the number of rows and columns for the flower arrangement.

The following ***r*** lines will each contain a row description. The row description starts with a non-negative integer, $\boldsymbol{g_b}$, representing the number of groups of black flowers in the corresponding row. Following this integer will be $\boldsymbol{g_b}$ positive integers representing the group sizes for the black flowers in the corresponding row.

Following the row descriptions will be ***c*** lines each containing a column description. The column description starts with a non-negative integer, $\boldsymbol{g_b}$, representing the number of groups of black flowers in the corresponding column. Following this integer will be $\boldsymbol{g_b}$ positive integers representing the group sizes for the black flowers in the corresponding column.

It is guaranteed that at least one arrangement will work for the given input.

Output

Output should contain ***r*** lines. Each line will contain c character. The i-th character of the j-th line represents the color of a plant ("r" for red and "B" for black).

Sample Input	**Sample Output**

5 5
1 5
2 2 1
2 1 1
2 3 1
1 5
2 1 2
1 5
2 2 2
3 1 1 1
2 2 2
	BBBBB
	rBBrB
	rBrBr
	BBBrB
	BBBBB

3 6
2 1 4
3 1 2 1
2 4 1
1 3
1 1
1 3
1 3
1 1
1 3
	BrBBBB
	BrBBrB
	BBBBrB

Explanation

Case 1

The output should look like the following

```
B B B B B
r B B r B
r B r B r
B B B r B
B B B B B
```

In the first row (BBBBB) there is 1 group of 5 black flowering plants.
In the second row (rBBrB) there are 2 groups of 2 and 1 black flowering plants.
In the third row (rBrBr) there are 2 groups of 1 and 1 black flowering plants.
In the fourth row (BBBrB) there are 2 groups of 3 and 1 black flowering plants.
In the last row (BBBBB) there is 1 group of 5 black flowering plants.

In the first column (BrrBB) there are 2 groups of 1 and 2 black flowering plants.
In the second column (BBBBB) there is 1 group of 5 black flowering plants.
In the third column (BBrBB) there are 2 groups of 2 and 2 black flowering plants.
In the fourth column (BrBrB) there are 3 groups of 1, 1, and 1 black flowering plants.
In the last column (BBrBB) there are 2 groups of 2 and 2 black flowering plants.

Case 2

```
B r B B B B
B r B B r B
B B B B r B
```

Hints

Like Sudoku: You should think of this problem as something like that of sudoku. However, instead of having 9 choices per spot, you only have 2 (red or black). Unlike in sudoku in this problem the whole board should start empty.

Validation: You can run the validation on the last set node. You will have to ensure that no group in the relevant row/column has too many black flowers. Additionally, you have to ensure that, if a row or column is completed, then all the groupings have been fulfilled in said row or column.

Grading Criteria

- Read/Write from/to **standard** input/output (e.g. scanf/printf and no FILE *)
    - 10 points
- Good comments, whitespace, and variable names
    - 15 points
- Create a way to validate some spot in the grid by checking it’s row and column
    - 10 points
- Use a recursive method to try all “reasonable” possibilities
    - 5 points
- Store the grid and print it when the solution is found or ensure that the grid is not modified when returning
    - 10 points
- Programs will be tested on <u>10 cases</u>
    - 5 points <u>each</u>

No points will be awarded to programs that do not compile using “gcc -std=gnu11 -lm”.

Sometime a requested technique will be given, and solutions without the requested technique will have their maximum points total reduced. For this problem use backtracking. ***<u>Without this programs will earn at most 50 points!</u>***

Any case that causes a program to return a non-zero return code will be treated as wrong. Additionally, any case that takes longer than the maximum allowed time (the max of {5 times my solutions time, 10 seconds}) will also be treated as wrong.

<u>No partial credit will be awarded for an incorrect case.</u>