F2 Eating Brownies (Hard)

Time limit: 2s

The only difference between this problem and "E1: Eating Brownies (Easy)" is the constraint of $\boldsymbol{K}$. In this problem, $1 \leq \boldsymbol{K} \leq 2$. For scoring purposes, this problem and "E1: Eating Brownies (Easy)" are considered two different problems. Therefore, to obtain points from both problems, you need to submit your code in both problems.

Description

You have brownies cut into $\boldsymbol{M}-1$ horizontal cuts and $\boldsymbol{N}-1$ vertical cuts, thus having $\boldsymbol{M}$ rows (numbered 1 to $\boldsymbol{M}$) and $\boldsymbol{N}$ columns (numbered 1 to $\boldsymbol{N}$) of brownie pieces. The piece in row i and column j is denoted as piece (i, j).

You have eaten some of the pieces. The table $\boldsymbol{B}$ represents which pieces you have eaten. The value of $\boldsymbol{B}_{i,j}$ is 0 if you have eaten piece (i, j), and 1 if you have not eaten piece (i, j). There are at least $\boldsymbol{K}+1$ pieces that you have not eaten.

At the end of the day, a mouse will eat your brownies. The mouse will eat the largest rectangular set of pieces that have not been eaten. In other words, the mouse will eat pieces from row r_1 to r_2 and from row c_1 to c_2 that maximises the value of $(r_2 - r_1 + 1) \times (c_2 - c_1 + 1)$ with the condition: for each $r_1 \leq i \leq r_2, c_1 \leq j \leq c_2$, $\boldsymbol{B}_{i,j} = 1$.

You must eat exactly $\boldsymbol{K}$ pieces today. If you choose to eat pieces $(x_1, y_1), \cdots, (x_{\boldsymbol{K}}, y_{\boldsymbol{K}})$, then the set of pieces chosen by the mouse cannot contain these pieces. In other words, the mouse cannot eat the pieces from row r_1 to r_2 and from column c_1 to c_2 if there exists p satisfying $r_1 \leq x_p \leq r_2$ and $c_1 \leq y_p \leq c_2$.

Determine which pieces you should eat to minimise the number of pieces eaten by the mouse at the end of the day.

Input

The first line contains three integers $\boldsymbol{M}$, $\boldsymbol{N}$, and $\boldsymbol{K}$ ($1 \leq \boldsymbol{M}, \boldsymbol{N} \leq 1\,000; 1 \leq \boldsymbol{K} \leq 2$) separated by spaces. The next $\boldsymbol{M}$ lines each contain $\boldsymbol{N}$ characters '`0`' or '`1`'. The j-th character in the $i+1$-th line is the value of $\boldsymbol{B}_{i,j}$. There are at least $\boldsymbol{K}+1$ pieces that you have not eaten.

Output

The first line contains the minimum number of pieces eaten by the mouse at the end of the day. The next $\boldsymbol{K}$ lines each contain two integers separated by a space. The $b+1$-th line contains integers x_b and y_b. The pieces that you eat must be pairwise distinct and have not been eaten before. You can print the pieces in any order. If there is more than one solution, you may output any solution.

Sample Input 1	Sample Output 1
3 4 1 1001 1111 0011	3 2 4

Sample Input 2	Sample Output 2
3 4 2 1001 1111 0011	2 3 4 2 2

Sample Input 3	Sample Output 3
3 3 2 101 010 101	1 1 3 3 1

Explanation of samples

In sample input 1, by eating piece $(2, 4)$, the largest set of pieces that can be eaten by the mouse contains 3 pieces: pieces $(2, 1)$, $(2, 2)$, and $(2, 3)$. If you eat piece $(2, 3)$, then the largest set of pieces that can be eaten by the mouse also contains 3 pieces: pieces $(1, 4)$, $(2, 4)$, and $(3, 4)$. Therefore, the following output is also allowed.

```
3
2 3
```

If you eat piece $(2, 2)$, then the largest set of pieces that can be eaten by the mouse contains 4 pieces: pieces $(2, 3)$, $(2, 4)$, $(3, 3)$, and $(3, 4)$. Therefore, the following output is **not** allowed.

```
3
2 2
```

It can be shown that you cannot eat 1 piece such that the largest set of pieces that can be eaten by the mouse contains less than 3 pieces.