Problem A. Saintly . . . Coins

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	512 megabytes

Rikka found a futuristic coat covering her body when waking up.

"It's cold at a high place like here."

It is the sound from the bankrupt boy standing there.

"Don't show that face... 'Bankrupt' doesn't mean that for me. I do have run out of my money, but I still have a job... And nobody can pull me out of my chair in the net bar for debts. "

Rikka knows that he has managed to smile to her. She is too weak, inexperienced and naive to say anything, but at least she could do something... The clashing coins remind her.

"Do you know 'Coin Fighting'? It used to be popular on the earth..."

"I'm a lunar native."

"Sorry... But I thought it may be... Good for you now..." Rikka's voice is reducing. Maybe her words hurt the boy who must be extremely sensitive to "coins" now.

"What's its rule?"

...

"Thank you, earthling." He reached his hand out and little lovely Rikka caught it, with thanks to and for both young juveniles. "We will meet again at the crossing of our fate."

"Jyaou Shingan Wa Saikou Desu!"

There are m coin stacks. Each of them has n coins initially, but their capacities are unlimited. They are lined up, forming an $n \times m$ grid as columns. Let (i, j) be the i-th position from bottom to top in the j-th stack from left to right. A position (x, y) is *less than* (x', y') if and only if $x < x'$ or $(x = x', y < y')$.

There are two categories of coins: ordinary ones and special ones. Ordinary coins are divided into 6 types by their denominations: $\{1, 5, 10, 50, 100, 500\}$; special coins are divided into two types: $\{X, Y\}$. Hence there are 8 types of coins in total.

Each ordinary coin is either *inactive* or *active*, while special coins are **always** *inactive* even if they should be *activated*.

Ordinary coins can be merged into a higher level or removed by some rules. Each denomination has a basic merging threshold, of $\{5, 2, 5, 2, 5, 2\}$ respectively. If a *four-connected component* (explained in the section "Notes", the same below) of the same denomination contains at least one *active* coin, and its size is not less than the corresponding threshold, then it is *removable*.

While removing a *removable* component, one can get a score increment of its size. Besides, if the same denomination of the removed coins is less than 500, **one** additional *active* coin of the denomination one level higher will be generated and placed at the *least* position among the places of the removed coins. Nothing will be placed when the same denomination of the removing coins is 500. Notice that no matter how many coins are removed, no more than one coin will be placed.

A game is made up of some *rounds*. Each *round* is described in order as follows:

1. *Inactivate* all coins at the beginning of the *round*.
2. Choose a coin-type t. Then all coins of type t at the top of a stack will be *picked*; do this in each stack repeatedly until nothing changes. Let $c(t)$ be the number of *picked* coins when choosing type t. An ordinary type t can be chosen only if $c(t) \geq 1$, while a special type t must satisfy $c(t) \geq 2$ for choosing.

- If t is ordinary, a stack x should be chosen and the *picked* coins will be put at the top of the stack immediately and *activated*.
- If t is special, a non-empty stack x with the top of an ordinary coin should be chosen, which also means one cannot choose the type t in the previous step if there is no such stack x. Let y be the type of the coin at the top of the chosen stack.

  Then, in case that t is special, the *picked* coins will be removed and thus the player will get a score increment of their amount.

  Meanwhile, all coins of type y will be removed immediately and thus the player will get a score increment of their amount.

  After that, if t is the type X and the denomination of y is less than 500, *active* coins of the denomination one level higher than y will be generated and take the places of all the coins of type y just removed.

3. *Activate* each *inactive* coin whose position is *four-adjacent* to at least one position where a coin was just removed (in step 2 and step 6, including the positions where high-level coins have been placed).
4. Coins fall down. That is, if a coin is *over* an adjacent blank position, it will be moved there; do this repeatedly until nothing changes. Obviously, the final status does not depend on the order of moving.
5. *Inactivate* each coin which is not in any *removable* component.
6. Remove all *removable* components **together**, get their score increments and place additional generated coins. If nothing could be removed, then the *round* ends, or jump to step 3 otherwise.

As mentioned above, a valid *operation* is determined by (t, x), the type and the stack chosen, respectively. A valid *operation* must gain at least a score incerment of 1. If there is no valid *operation* for a *round*, then the game ends.

In each round, the boy chooses a *best* operation among all valid ones. For two different valid *operations* $o_1 = (t_1, x_1), o_2 = (t_2, x_2)$, he compares them as follows:

1. If the categories of t_1 and t_2 are different, the special one is *better* than the ordinary one.
2. In case of a tie, if the increments of scores are different, the higher one is *better* than the lower one.
3. In case that a tie still remains, if t_1 and t_2 are different, the one whose type is appeared at left in $\{X, Y, 1, 10, 100, 5, 50, 500\}$ is *better*.
4. In other cases, the one with smaller x is *better*.

Could you please play a referee for them?

Given the initial stacks, please implement the boy's *operations*. You need to calculate the total number of *rounds*. And for each *round*, the type and the stack chosen are wondered, and so is the score he gets.

Input

The first line contains two integers n, m ($1 \le n \le 2000, 1 \le m \le 20$), the initial number of coins in each stack and the number of stacks, respectively.

Each of the following n lines contains a string consisting of $\{A, B, C, D, E, F, X, Y\}$, where characters $\{A, B, C, D, E, F\}$ refers to denominations $\{1, 5, 10, 50, 100, 500\}$ respectively, and the j-th character in the i-th line describes the type of the coin at (i, j). Notice that the character matrix is upside down for input.

It is guaranteed that there are at most 20 special coins (i.e. the coins of types X and Y).

Output

Output a single integer K at the first line, the total number of rounds.

Each of the following K lines contains a character and two integers. The character describes the type (in $\{A, B, C, D, E, F, X, Y\}$, as the input) which he chooses in that *round*, and the integers are the index of the chosen stack and the score increment in that *round*, respectively.

Examples

standard input	standard output
6 3 EEF EED ACC AAC AAC AAC	3 A 1 12 D 2 7 F 2 2
6 3 EEF EED ABC AAC AAC XAX	2 X 3 22 F 2 2
6 3 EEF EED ACC AAC AAC YAY	1 Y 3 12
7 2 EE FF FA AD DC EE YY	2 Y 1 9 D 1 2

Note

Two positions $(x_1, y_1), (x_2, y_2)$ are *four-adjacent* if and only if $|x_1 - x_2| + |y_1 - y_2| = 1$.

Two positions a, b are *four-connected* if and only if there exists a sequence of positions $a, c, \ldots, b$ beginning with a and ending with b such that every two adjacent positions in the sequence are *four-adjacent*.

A set of coins S is *four-connected* if and only if positions of every two coins in it are *four-connected*.

A *four-connected component* is a *four-connected* set of coins such that there doesn't exist a *four-connected* proper superset of it.

A position (x_1, y_1) is *over* (x_2, y_2) if and only if $x_1 = x_2 + 1, y_1 = y_2$.

In step 2, when you *pick* a coin, it is moved out from the stacks and put into a buffer.