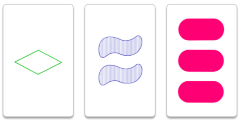
Task Set



In the popular card game *SET*, the player's goal is to identify a certain triplet of cards with some special properties, called a *set*. Each card shows some figures, which differ in number, shape, transparency and color.

Marin and Josip have recently bought a deck of these cards and now they can't stop playing. They've become so skilled at noticing *sets* that it soon became boring that the cards are determined by only four properties. Thus, they have decided to have fun with a generalized version of the game.

At their disposal is a deck of n **different** cards. Each card is represented by a sequence of k characters, each being one of `1`, `2` or `3`. The order of the cards in the deck does not matter.

An unordered triplet of cards is called a *set* if for each of the k positions, the three characters corresponding to the three cards are either the same or pairwise different. For example, three cards represented by `1123`, `1322` and `1221` make a *set* because all of the characters in the first and third positions are the same (`1` and `2` respectively), and the characters in the second and fourth positions are different (`1`, `2` and `3` in some order).

While looking at these n cards on the table, they started to wonder: how many unordered triplets of these n cards make a set. Write a program which will answer their question.

Input

The first line contains the integers n and k - the number of cards in the deck and the number of properties of a single card, respectively.

Each of the following n lines contains a sequence of k characters representing a card. Each character is one of `1`, `2` or `3`. Different lines contain different sequences of characters.

Output

In the only line, print the number of unordered triplets which form a *set*.

Scoring

In every subtask, it holds that $1 \le k \le 12$ i $1 \le n \le 3^k$.

Subtask	Points	Constraints
1	10	$1 \le k \le 5$
2	30	$1 \le k \le 7$
3	70	$1 \le k \le 12$

Examples

input

```
3 4
1123
1322
1221
```

output

```
1
```

input

```
2 2
11
22
```

output

```
0
```

input

```
5 3
111
222
333
123
132
```

output

```
2
```

Clarification of the third example:
The two *sets* are `111`, `222`, `333` and `111`, `123` i `132`.