D. Haar Features

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

The first algorithm for detecting a face on the image working in realtime was developed by Paul Viola and Michael Jones in 2001. A part of the algorithm is a procedure that computes *Haar features*. As part of this task, we consider a simplified model of this concept.

Let's consider a rectangular image that is represented with a table of size $n \times m$. The table elements are integers that specify the brightness of each pixel in the image.

A *feature* also is a rectangular table of size $n \times m$. Each cell of a *feature* is painted black or white.

To calculate the value of the given feature at the given image, you must perform the following steps. First the table of the feature is put over the table of the image (without rotations or reflections), thus each pixel is entirely covered with either black or white cell. The *value* of a feature in the image is the value of W - B, where W is the total brightness of the pixels in the image, covered with white feature cells, and B is the total brightness of the pixels covered with black feature cells.

Some examples of the most popular Haar features are given below.

Your task is to determine the number of operations that are required to calculate the feature by using the so-called *prefix rectangles*.

A *prefix rectangle* is any rectangle on the image, the upper left corner of which coincides with the upper left corner of the image.

You have a variable $value$, whose value is initially zero. In one *operation* you can count the sum of pixel values at any prefix rectangle, multiply it by any integer and add to variable $value$.

You are given a feature. It is necessary to calculate the minimum number of *operations* required to calculate the values of this attribute at an arbitrary image. For a better understanding of the statement, read the explanation of the first sample.

Input

The first line contains two space-separated integers n and m $(1 \le n, m \le 100)$ — the number of rows and columns in the feature.

Next n lines contain the description of the feature. Each line consists of m characters, the j-th character of the i-th line equals to "W", if this element of the feature is white and "B" if it is black.

Output

Print a single number — the minimum number of operations that you need to make to calculate the value of the feature.

Examples

input

```
6 8
BBBBBBBB
BBBBBBBB
BBBBBBBB
WWWWWWWW
WWWWWWWW
WWWWWWWW
```

output

```
2
```

input

```
3 3
WBW
BWW
WWW
```

output

```
4
```

input

```
3 6
WWBBWW
WWBBWW
WWBBWW
```

output

```
3
```

input

```
4 4
BBBB
BBBB
BBBB
BBBW
```

output

```
4
```

Note

The first sample corresponds to feature B, the one shown in the picture. The value of this feature in an image of size 6×8 equals to the difference of the total brightness of the pixels in the lower and upper half of the image. To calculate its value, perform the following two *operations*:

1. add the sum of pixels in the prefix rectangle with the lower right corner in the 6-th row and 8-th column with coefficient 1 to the variable $value$ (the rectangle is indicated by a red frame);
2. add the number of pixels in the prefix rectangle with the lower right corner in the 3-rd row and 8-th column with coefficient -2 and variable $value$.

Thus, all the pixels in the lower three rows of the image will be included with factor 1, and all pixels in the upper three rows of the image will be included with factor $1 - 2 = -1$, as required.