Problem ID: kerning

Kerning

In typography, *kerning* is the process of adjusting the spacing between characters (also known as *glyphs*) in a font. A well-kerned font is one where the spacing between individual characters is visually appealing. Fonts are typically represented as a two dimensional array of bit pixels (black or white) where glyphs are placed side by side and are kerned appropriately.

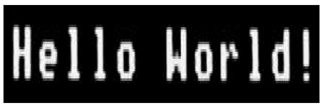

Figure 1: A well(ish)-kerned font represented as a bitmap.

```
00011111
00110000
01000000
00110000
00011111
```

Figure 2: A 5x8 bitmap. The glyph is highlighted in red.

Let's say that '1' represents a bit pixel that is part of a glyph and '0' represents a bit pixel that is part of the background. Then, we formally define a glyph as a maximal set of '1' bit pixels with the property that we can find a path from every bit pixel in the set to every other one (a path consisting of only up/down, left/right, and diagonal moves). Suppose we are given a bitmap that contains exactly two glyphs. The *level of kerning* between these two glyphs is the minimum distance between any point of the first and second glyph (where distance is measured using the square of the euclidean distance). More specifically, the distance between two points (x_0, y_0) and (x_1, y_1) is $(x_0 - x_1)^2 + (y_0 - y_1)^2$; the first coordinate is the row number (starting at the top as 0) and the second coordinate is the column number (starting from the left as 0).

Input

The input will begin with a line containing a single positive integer, t, representing the number of test cases to process. Each test case will begin with a single line containing two space-separated integers N and M ($1 \leq N, M \leq 100$) each representing number of rows and columns in the bitmap, respectively. Following will be N lines each containing M characters (which will be either '0' or '1') which collectively are the bitmap for this test case. It is guaranteed that each bitmap will contain *exactly* two glyphs.

Output

For each test case print the level of kerning between the two glyphs on its own line.

Sample Input

```
2
6 20
00000000000000000000
11111111000011000000
00011000000011000000
00011000000011100000
00011000000011111110
00000000000000000000
5 10
0000111000
0010001000
1110001000
0010001000
0000000100
```

Sample Output

```
25
5
```