

# Hard Image Recognition

It is a well established fact that every civilization goes through one key stage of technological development, which is best illustrated by the rapid transition from typewriters to GUI based word-processing computer software. What is more remarkable, during this transition period taste and fashion is turned upside down, and good old esthetic sensibility is replaced by the urge to show off the whole repertoire of fonts available in the typists brand new word-processor to the wide, wide world. Even if some of the fonts look like the handwriting of a 1-year-old chimpanzee.

All civilizations which have already gone through this stage of transition look back on it with a smile. All, with one notable exception -- that of Byteland. When the multiple font craze in Byteland reached its heights, it engulfed everyone (including the usually conservative passport department officials), and now everyone is paying the price. Automating the process of passport scanning is turning out a real nightmare, since all 6-digit passport numbers are written in different fonts. Not surprisingly, you've been asked to help out.

You should write a program for recognizing images of 6-digit numbers. Naturally, in some cases, the image may be blurred or deformed, but a human can always recognize the number written (possibly after overcoming the initial surprise). The picture is black & white, and consists of a white background with a black raster image on it (character "." denotes the color white and character "X" - the color black).

## Input

$t$  – number of test cases [ $t \leq 250$ ], then  $t$  tests follow.

Each test case starts with 2 integers,  $H$  and  $W$ , denoting height and width of picture [ $10 \leq H, W \leq 200$ ]. Then  $H$  rows with  $W$  characters ('.' or 'X') in each follow.

## Output

For each test case you should output the numerical value of the recognized 6-digit number in separate lines.

## Score

The score is equal to the number of images successfully identified.

## Example

Input :

2

13 40

```
.....
.....
.....XX...XX...XX.....X..XX...XX.....
.....X..X..X..X..X..X...XX..X..X..X.....
.....X..X..X..X..X..X...XX..X..X..X.....
.....X..X..X..X..X..X...XX..X..X..X.....
.....X..X..X..X..X..X...XX..XX.....X.....
.....X..X..X..X..X..X...X..X..X..X.....
```

```

.....X..X.X.X.X.X.X.X.X.X.....
.....X..X.X.X.X.X.X.X.X.X.....
.....X..X.X.X.X.X.XXXXXXX.X.X.....
.....XX..XX..XX.....X..XX..XXXX.....
.....
12 60
.....XX
.....XXXXX...X..XXXXXX.XXXXX...X..XXXX.....
XX.....X.....XX...X..X.X...X..XX.X...X.....
.....X.....X.....X..X.X...X..X.X.X...X.....
.....XXXX.X.....X..X...X..X.X...X.....
.....X..X.X.XXX.....X..XXXX..X..X...XXX.....
.....X.XX..X...X...X.X.X.X...X.....
.....X.X...X...X..X..X.XXXXXXX...X.....
.....X.X...X..X...X...X...X.X.X...X.....
.....X..X..X.X...X...X...X...X.X.X...X.....
.....XXX.....XX...X.....XXXXX...XXXXXX.XXX.....XX
XX.....

```

**Output :**

000482  
567833

**Score :**

score = 1 (the first answer is correct,  
the second one has one incorrect digit)

000482	567843
--------	--------

**Picture examples :**

381729 254839 349010 491881, 457840  
~~189765~~ 371981 271891 **177188** 388910