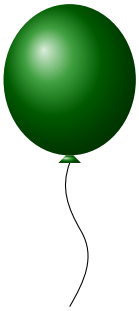
XAPC Algorithmic Programming Contest > badas

H Ash-e-emö



Time limit: 1.0s
Memory limit: 256MB

You are given a clock where a day consists of h hours and every hour consists of m minutes. The time is written in the format $a : b$ ($0 \le a < h$, $0 \le b < m$). **Note** that the leading zeros are preserved. So for $h = 240$, $m = 100$, $a = 9$, $b = 41$ the time would look like `009:41`.

We call a time `a:b` *nice* if both of the following values are perfect squares[1]:

1. The total number of minutes since the beginning of the day: $a \cdot m + b$.
2. The concatenation of a and b written as a decimal number, where m is written with leading zeros to always have the same number of digits.

More formally, let d be the number of digits in $m - 1$. Then define: $\text{concat}(a, b) = a \cdot 10^d + b$

For example, for $h = 24$, $m = 60$, the time $1 : 21$ is *nice*, because:

- $1 \cdot 60 + 21 = 81 = 9^2$
- $\text{concat}(1, 21) = 121 = 11^2$

Your task is to determine how many *nice* times there are in a day.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1000$). The description of the test cases follows.

The only line of each test case contains two integers h and m ($2 \le h, m \le 5 \cdot 10^5$).

It is guaranteed that the sum of h and the sum of m across all test cases do not exceed $5 \cdot 10^5$.

Output

For each test case, output a single integer — the number of *nice* times in the corresponding day.

[1] A perfect square is an integer that is the square of another integer.

Samples

Sample input 1	Sample output 1
7 24 60 24 91 24 99 24 100 90000 90000 10 400000 400000 10	10 11 10 49 327 640 2000