Long before Gutenberg invented letterpress printing, books have been transcribed by monks. Cloisters wanted to be able to check that a book was transcribed by them (and not by a different cloister). Although watermarked paper would have been an option, the cloister preferred to use a system of hard-to-fake serial numbers for identifying their transcriptions.

Each serial number consists of 10 single numbers $a_1, a_2, \ldots, a_{10}$. Valid serial numbers satisfy $a_1 + a_2 + \ldots + a_9 \equiv a_{10} \pmod N$ with $0 \leq a_{10} < N$. The N is specic to and only known by the cloister that has transcribed this book and is therefore able to check its origin.

You are confronted with a pile of books that presumably have been transcribed by a single cloister. You are asked to write a computer program to determine that cloister, i.e. to calculate the biggest possible N that makes the serial numbers of these books valid. Obviously, no cloister has chosen $N = 1$. So if your calculations yield $N = 1$, there must be something wrong.

Input

Input starts with an integer t on a single line, the number of test cases ($1 \leq t \leq 100$). Each test case starts with an integer c on a single line, the number of serial numbers you have to consider ($2 \leq c \leq 1000$). Each of the following c lines holds 10 integer numbers $a_1, a_2, \ldots, a_{10}$ ($0 \leq a_i < 2^{28}$) separated by single spaces.

Output

For each test case, output a single line containing the largest possible N, so that each given serial number for that test case is valid. If you cannot nd a $N > 1$ satisfying the condition for all serial numbers or if the numbers are valid independent of the choice of N, output '`impossible`' (without the quotes) on a single line.

Sample Input

```
4
2
1 1 1 1 1 1 1 1 1 9
2 4 6 8 10 12 14 16 18 90
3
1 1 1 1 1 1 1 1 1 1
5 4 7 2 6 4 2 1 3 2
1 2 3 4 5 6 7 8 9 5
2
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 0
2
2 2 2 2 2 2 2 2 2 0
1 1 1 1 1 1 1 1 1 1
```

Sample Output

```
impossible
8
impossible
2
```