Problem L — A Version of Nim Game

Here we introduce a special version of the combinatoric game called *Nim*. This progressively finite take-away game involving 4 piles of stones. In this game, the players take turns removing at least one but at most 3 stones from one of the piles. The player who takes the last stone from the table losses.

The configuration of an instance of the Nim game can be described with four nonnegative integers representing the sizes of these four piles. That is, the configuration in a Nim game requires a vector of nonnegative integers (p_1, p_2, p_3, p_4), the kth number p_k representing the current size of the kth pile.

To write a program that plays the Nim game perfectly, we need to decide whether a given instance of the Nim game is winnable. For example, the configuration $(0, 0, 0, 2)$ is winnable by removing one stone from the last pile; however, you can verify that neither $(0, 0, 0, 5)$ nor $(2, 2, 0, 0)$ is winnable. Further, to make the problem easier, we assume that the number of stones on each pile is at most 9 stones.

Input File

Several instances of Nim game configurations. The inputs are a list of integers. Within each set, the first integer (in a single line) represents the number of Nim game configurations, n, which can be as large as 20. After n, there will be n lists of Nim game configurations; each line contains four integers $\langle p_1, p_2, p_3, p_4 \rangle$. Note that integers $0 \leq p_1, p_2, p_3, p_4 \leq 9$.

Output File

For each configuration appeared in the input, decide whether it is winnable. In particular, output a single number '`1`' if the instance is winnable; otherwise, output a single number '`0`' if it is not winnable.

Sample Input

```
4
0 0 0 5
0 0 0 6
0 0 2 2
1 2 3 4
```

Sample Output

```
0
1
0
0
```