2181 A Version of Nim

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

The configuration of an instance of this Nim game can be described with four nonnegative integers representing the sizes of these four piles. i.e., (p_1, p_2, p_3, p_4), where the k-th, $1 \leq k \leq 4$, number p_k representing the current size of the k-th pile.

A configuration of this Nim game is *winnable* if a player facing this configuration can allway find a way to win the game. To write a program that plays the Nim game perfectly, we need to decide whether a given instance of the Nim game is *winnable* to the current player or not facing the given configuration. 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 the number of stones on each pile is at most 9.

Input

The first line contains n, the number of Nim game configurations, which can be as large as 20. After n, there will be n lists of Nim game configurations; each line contains four integers p_1,p_2,p_3 and p_4. Note that $0 \leq p_1, p_2, p_3, p_4 \leq 9$.

Output

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

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
```