Problem L

LCG Manipulation

Time Limit: 2 seconds

A linear congruential generator (LCG) is a process that yields an infinite stream of pseudo-random numbers. This process is used by many video games in order to quickly generate a sequence of "good enough" random numbers for various non-deterministic processes.

For example, consider the Pokémon games. Whenever you catch a new Pokémon, each of its six stats receives an integer bonus "uniformly randomly" selected from $+0$ to $+31$. The intent is to model how each individual Pokémon has its own strengths and weaknesses that differentiate it from others of the same species. These bonus stats are decided by the output of an LCG.

Competitive players need their team to be at peak strength. However, the chances of landing a perfect $+31$ on *all* six stats is $1/2^{30}$, or a little less than one in one *billion*. But what if I told you that, *somehow*, a knowledgeable player can "miraculously" get perfect-statted Pokémon in only a handful of attempts.

In fact, in many games, it is possible to rig seemingly-random in-game events in order to always produce the optimal outcome. How!? Well, let's delve a little bit into how LCGs work...

An LCG generates an infinite sequence $x_0, x_1, x_2, x_3, \ldots$ which is determined by four values: the non-negative integers a, b, s, and p. Let $x_0 = s$ (which is why s is often called the "seed"). For each $n \geq 1$, define the next term in the sequence as,

$$x_n = (ax_{n-1} + b) \bmod p$$

In general, the modulus can be any number, but for this problem, **we use the letter p because we are only considering the cases where the modulus is prime**.

The LCG may *seem* random, but it's actually quite predictable! If you can uncover the values of a, b, s, and p, and if you know that the optimal outcome occurs when the LCG generates a value of v, then you can use a computer program to know exactly how long you would need to wait until your game is in the desired "random" state.

I think you can guess the problem now. You must perform an *Implausible Chance's Probability Calculation.* Given the values of a, b, s, p, and some value v, find the minimum non-negative integer n such that $x_n = v$; if no such n exists, then you should report so as well. Also, there will be T independent test cases per file.

Input Format

Input begins with a line containing the positive integer T, the number of test cases.

Then, T lines follow. Each line contains the five space-separated integers a, b, s, p, and v.

Constraints

- $1 \leq T \leq 40$
- $2^3 - 1 \leq p \leq 2^{31} - 1$
- p is prime

- $0 < a, b < p$
- $0 \leq s, v < p$

Output Format

For each test case, output a single line:

- If an n exists such that $x_n = v$, then output the minimum such n
- Otherwise, output the word `IMPOSSIBLE`

Sample Input 1

```
8
2 3 1 7 5
2 3 1 7 6
2 3 1 7 1
2 3 1 7 0
7 1 2 31 24
7 1 2 31 25
7 1 2 31 26
7 1 1 31 26
```

Sample Output 1

```
1
2
0
IMPOSSIBLE
6
4
IMPOSSIBLE
2
```

Sample Input 2

```
2
314159265 1234567890 2022 2147483647 2023
1234567890 314159265 2022 2147483647 2023
```

Sample Output 2

```
1058977885
IMPOSSIBLE
```

Explanation

Let's suppose that $(a, b, s, p) = (7, 1, 2, 31)$.

Then, we would have $x_0 = 2$ and $x_n = (7x_{n-1} + 1) \bmod 31$.

- $x_1 = (7 \times 2 + 1) \bmod 31 = 15$.
- $x_2 = (7 \times 15 + 1) \bmod 31 = 13$.
- $x_3 = (7 \times 13 + 1) \bmod 31 = 30$.
- $x_4 = (7 \times 30 + 1) \bmod 31 = 25$.
- $x_5 = (7 \times 25 + 1) \bmod 31 = 21$.
- $x_6 = (7 \times 21 + 1) \bmod 31 = 24$.
- $\vdots$