Problem B. 三进制

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

这是一道交互题。

对于 n 位三进制数，小 A 设计了一个非常简单的加密算法：对每一位 $i = 0, 1, \cdots, n-1$ 构造双射 $f_i : \{0,1,2\} \to \{0,1,2\}$。对于一个 n 位三进制数 A，假设其加密前从高位到低位依次是：

$$a_{n-1}, a_{n-2}, \cdots, a_0$$

那么其加密后从高位到低位就是：

$$f_{n-1}(a_{n-1}), f_{n-2}(a_{n-2}), \cdots, f_0(a_0)$$

为了方便表示每一位的加密，我们可以将双射 $f_i = \{0 \to x, 1 \to y, 2 \to z\}$（当然有 $x, y, z \in \{0,1,2\}$ 且 x, y, z 互不相同）简单地表示为 $f_i = xyz$。例如 $f_0 = \{0 \to 1, 1 \to 2, 2 \to 0\}$ 就可以表示为 $f_0 = 120$。

比如小 A 对 3 位三进制数构造双射 $f_0 = 021, f_1 = 120, f_2 = 210$，那么 120 加密后就是 100，201 加密后就是 012。

小 A 当然不希望别人知道它构造的双射，但是加密算法终究是要使用的，所以小 A 会告诉你 n 的大小，并允许你询问 2 次加密状态下加法的结果，而你的任务是依靠这 2 次询问破解小 A 构造的 n 个双射 $f_0, f_1, \cdots, f_{n-1}$。

Input

第一行一个正整数 T $(1 \le T \le 10^4)$，表示数据组数。

每组数据一行一个正整数 n $(1 \le n \le 10^5)$，表示三进制位数。

保证单个测试点内每组数据中 n 的和不超过 10^6。

Interaction Protocol

你可以进行不超过 2 次询问，之后回答小 A 的加密映射。

每次询问，你应该以格式 `? a b` 给出加法的两个参数 a, b，其中 a, b 都是 n 位三进制数并要求你以字符串格式给出。在刷新输出流之后，你需要以字符串格式读入一行一个 $n+1$ 位三进制数 c，其中 c 是 a, b 分别解密后相加之后再加密的结果，其中最高位作为溢出标志不加密。形式化的：

$$c = f(f^{-1}(a) + f^{-1}(b))$$

其中 f 是加密双射，f^{-1} 是 f 的逆映射。

注意，所有三进制字符串的输入都是从高位到低位！

注意，c 的最高位即第 n 位作为溢出标志不加密，或者你可以认为 $f_n = 012$。

如果要回答小 A 的加密映射，你应该以格式 `! f0 f1 ... f(n-1)` 给出 n 个加密映射，其中 $f_0, f_1, \cdots, f_{n-1}$ 都应该是字符串 `012`, `021`, `102`, `120`, `201`, `210` 中的一个。在刷新输出流之后，交互器会立即判定你的答案是否正确，然后进行下一组数据交互或结束程序，而不会有多余输出。

注意，小 A 的加密映射在询问前已经完全确定，不会随着询问更改。换句话说，交互器不是自适应的。

为了刷新输出流，你可以：

- 在 `C/C++` 中使用 `fflush(stdout)`（如果使用 `printf`）或者 `cout.flush()`（如果使用 `cout`）。
- `Java` 中使用 `System.out.flush()`。

- 在 Python 中使用 `sys.stdout.flush()`。

Example

standard input	standard output
2 3 0210 1102 1 01	? 011 102 ? 010 202 ! 021 120 102 ? 0 1 ! 012

Note

对于第一组数据的第一个询问，011 和 102 解密后分别是 102 和 021，相加后得到 0200，再加密后是 0210。

对于第一组数据的第二个询问，010 和 202 解密后分别是 100 和 221，相加后得到 1021，再加密后是 1102。

对于第二组数据的第一个询问，0 和 1 解密后分别是 0 和 1，相加后得到 01，再加密后是 01。