Problem C: Bad-hash students

Time limit: 5 seconds

A *hash table* is a standard data structure in computer science that allows the storage of different items in a relatively small array. There are different implementations of hash tables and different ways to deal with overflows and collisions. One of which is called *hashing with open addressing and quadratic probe.*

This is how it works. For some fixed integer constant α suppose our hash table T has length n and consists of the cells $T[0]$, $T[1]$, $T[2]$, ..., $T[n-1]$. In order to save an object, we first apply an initial hash function to compute an initial key k_1, which is simply an index between 0 and $n-1$. We try to store the object at position k_1 in T. If $T[k_1]$ is already used, we try another position at index $k_2 = k_1 + \alpha \cdot 1^2 \mod n$. If $T[k_2]$ is also already used, we continue to probe the hash table at positions

$$k_i = k_1 + \alpha \cdot (i-1)^2 \mod n \text{ for } i = 3, 4, 5, \ldots,$$

and so on until we find the first empty position in the array T. If we are unable to find an empty cell of T after n probes, we give up and raise an exception. This way of storing objects is supposed to work well when the hash function that computes the initial key k_1 uniformly distributes its values among the integer interval $[\![0, n-1]\!]$ and when the number m of objects to be stored is a little smaller than n.

The sophomore students of Professor Amaretto's class have been asked to implement hash tables. Alas, not one student got it right — a horrendous situation that would never have happened in Professor Mayor's freshman class! It took a long time for Professor Amaretto to understand what his students had coded. Their first mistake was that their complicated choice of initial hash function was buggy and in fact always returns the same value k_1. Their second mistake was that they all coded the formula

$$k_i = k_1 + \alpha \cdot {k_{i-1}}^2 \mod n$$

instead of the correct formula.

Still, Professor Amaretto is curious to know how usable are these broken hash tables, implemented incorrectly by his bad-hash students.

Input

The input file consists of multiple test cases. The first line of the input file consists of a single integer c indicating the number of test cases. Each test case follows, and consists of, a single line with 3 integers n, k_1, α separated by single spaces, where n ($2 \leq n \leq 200000000$) is the size of the hash table, k_1 ($0 \leq k_1 \leq n-1$) is the constant return value of the wrongly coded initial hash function, and α ($1 \leq \alpha \leq n-1$ and $\alpha \leq 100$) is the coefficient used for the buggy quadratic probing.

Output

For each test case in the input, your program should produce one line consisting of one integer that indicates the number of cells of the hash table that can be assigned.

Sample Input

```
2
3 0 1
7 3 3
```

Sample Output

```
1
4
```