Roulette

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

This is an interactive problem. Please be sure to use the stream flushing operation after each query's output in order not to leave part of your output in some buffer. For example, you've got to use the `fflush(stdout)` in C/C++, `sys.stdout.flush()` in Python(do not forget `import sys`) and `System.out.flush()` in Java.

Roulette is a kind of gambling. For simplification, we assume its rules and steps as follows:

1. The whole gambling process composes of many turns.
2. In each turn:
   - You can choose an x and pay x yuan as the wager. x could be any positive integer in range $[1, 10^{700})$.
   - The maker will turn the roulette which has only two results indicated by two colors: black and white.
   - If the result corresponds to the white color, you will get $2x$ yuan from the maker, which means you gain x yuan in this turn. Otherwise, the maker will devour the x yuan you have paid, which means you lose x yuan in this turn.
3. After each turn, you can choose to continue gambling or just stop it.
4. The probability of two results are equal. That means both of them are 0.5.

Now a profiteer set a roulette game and lure people taking part in it. In order to defraud more money, he can set the probability of the roulette's result. More precisely, he can set the probability you win to p ($0 < p < 1$, accordingly, the probability you lose is $1 - p$) in advance and keep that probability until the end of game.

Unfortunately, little A is rich enough but failed to bear the endure and has lost y yuan due to gambling. Now he asks you, a smart ACMer, to help him win back his money. You don't want to use any money of yourself for gambling. So, he will lend you 10^{700} yuan–a large sum of money, hoping you to take part in that game to win back his money (that means, initially you have 10^{700} yuan, you should have at least $10^{700} + y$ yuan sometime and at any time you should have a positive amount of money all the time). You also don't want to waste time, so you hope to gamble for no more than 5000 turns.

Input

The first line includes three integers: $y(1 \le y \le 1145141919810), q(1 \le q \le 100), seed(1 \le seed \le 10^{18})$, indicates that little A lost y yuan and the profiteer set the probability p(you win) to $\frac{q}{100}$, $seed$ is used by the interactor to produce your win-lose sequence. **Your program needs to do nothing on it but only read it.** Your program will receive "0","1" or "2"(without quotes) as the result of the current turn where "0" means that you lose in this turn, "1" means you win in this turn but still haven't win back enough money and "2" means you win in this turn and have win back enough money(under this condition, to prevent unnecessary errors, please let your program terminate at once to get "Accepted". Otherwise, we have nothing to do but wish you good luck). Your program should read these results and make decisions accordingly.

Output

Several lines, each of which has the format x(without quotes, x is an positive integer in range $[1, 10^{700})$ **without leading zeros**), means that you continue the game and pay x yuan as the wager.

Interaction Protocol

The interactor will read your output. Then it will use *seed* to randomly produce a real number p' in range $[0, 1)$ and compare it with p. If $p' < p$, it will send "0", otherwise it will send "1" or "2" (See the input section). **You may assume that your win-lose sequence is unique and determined by *seed*.**

Under these (including but not limited to) conditions you may get result "Wrong answer", "Time limit exceed" or other incorrect results. Please check your code carefully before submitting.

1. Wrong output format.
2. Having used excessive amount of money.
3. Failed to win back enough money in 5000 turns.

Examples

standard input	standard output
1145141 91 9810 1 0 2	114514 1145141 11451419
100 100 2333 2	200