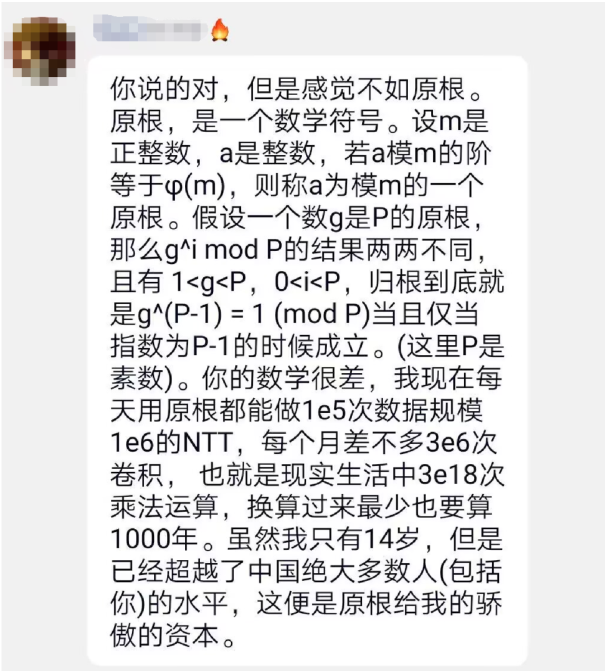
Primitive Root

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

BaoBao has just learnt about primitive roots in number theory and is showing off his knowledge to Little Cyan Fish through an instant messaging software.



This image is only for amusement and has nothing to do with the problem itself. You can safely skip this image if you can't read Chinese.

Based on the fact that if a non-negative integer g is a primitive root modulo P (where P is a prime), then $g^{P-1} \equiv 1 \pmod{P}$, BaoBao decided to use the expression `(g ^ (P - 1)) % P == 1` to check if g is a primitive root modulo P. Unfortunately, in most programming languages (for example C and C++), `^` is the bitwise exclusive-or (XOR) operator, not the power operator. Little Cyan Fish spotted this issue at once and now he is interested in the following problem:

Given a prime number P and a non-negative integer m, how many non-negative integers g satisfies $g \le m$ and $g \oplus (P-1) \equiv 1 \pmod{P}$? Here $\oplus$ is bitwise exclusive-or (XOR) operator.

Please help Little Cyan Fish solve this problem.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^5$) indicating the number of test cases. For each test case:

The first and only line contains two integers P and m ($2 \le P \le 10^{18}$, $0 \le m \le 10^{18}$, P is a prime).

Output

For each test case, output one line containing one integer indicating the number of g satisfying the constraints.

Example

standard input	standard output
3 2 0 7 11 1145141 998244353	1 2 872