Fake Coin and Lying Scales

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

Suppose you have c coins. One of them is fake, and is lighter than a real coin. All other coins are real, and have the same weight. You also have two-pan balance scales. In one weighing, you can put some coins on one pan and some coins on the other pan (and the remaining coins on neither pan), and see which pan is lighter, if any. Your task is to determine which coin is fake.

Sounds familiar, right? Well, there are good news and bad news. The bad news is that the scales may lie to you up to k times, and you have no way to know which weighing results are true. The good news is that you can make up to $3k$ mistakes. That is, you can make up to $3k + 1$ guesses, and you will win if at least one of them is correct.

Let $f(n, k)$ be the maximum number c such that, if you have c coins and the scales may lie up to k times, and you can make up to $3k$ mistakes, there exists a weighing strategy such that you can win by making no more than n weighings, whatever results you get.

You should find $f(n, k)$ **approximately**. In particular, output $\ln f(n, k)$, where ln is the natural logarithm. Your answer will be considered correct if the **absolute** difference between your answer and the correct one is no more than **10**.

Input

The first line contains an integer t, the number of test cases ($1 \le t \le 10^5$).

Each of the next t lines contains two integers n and k ($1 \le n \le 10^9$; $0 \le k \le 10^9$).

Output

For each test case, print a line with a single real number: the answer $\ln f(n, k)$ for that test case. The answer will be accepted if it differs from the right one by at most 10.

Example

standard input	*standard output*
2 100 0 100 1	109.8612289 106.1174552

Note

In the first test case, $f(100, 0) = 3^{100}$. Here, the answer is $\ln(3^{100}) = 100 \cdot \ln(3) = 109.8612289\ldots$.

Natural logarithm (logarithm with base $e = 2.71828182845904523536\ldots$) can be calculated using `log` in C++, `math.log` in Python, and `Math.log` in Java.