Problem K. K Tasks and the Random Scheduler

Source file name:	KTasks.c, KTasks.cpp, KTasks.java, KTasks.py
Input:	`Standard`
Output:	`Standard`
Time / Memory limit:	`1/2/2 (C++/Java/Python) second(s) / 64 megabytes`
Author(s):	`Gabriel Gutiérrez Tamayo - UTP Colombia`

You are a computer science student, and at your university, they still keep an old computer with two processing units. The first uses a standard scheduler from that era for CPU allocation, while the second has a scheduler implemented by former students as part of an Operating Systems course project from several years ago, which has not been modified since then.

The university has made this computer available to students for testing purposes, so every time a student runs their own programs, these are executed exclusively on the second processing unit.

The scheduler developed by the students uses a fairly simple algorithm for CPU allocation and works as follows. It always assigns a fixed CPU time of T_C milliseconds to each process and, when that time is up, it selects another process uniformly at random from all processes currently running, including the one that just released the CPU. A process can also lose its turn before exhausting its assigned CPU time if it makes certain system calls.

As part of a project in your Data Structures course, you have developed a program that must complete K tasks, and you now wish to determine the expected execution time on that computer. However, you have already conducted some tests and know that each task requires T_P ($1 \leq T_P \leq T_C$) milliseconds of computation to be completed on that computer. Since the behavior of the processes running may vary greatly, you apply a simple and pessimistic approach, assuming there are P other processes running in addition to your program, and that each process uses its full assigned CPU time during its turn. However, your program loses its turn every time it completes a task, since it prints a small result to the screen. You also assume that the time to select the next process and to print to the screen is negligible.

Input

The first line contains an integer q $\left(1 \leq q \leq 10^5\right)$, indicating the number of queries. Each of the following q lines contains 4 integers P, K, T_C, and T_P ($1 \leq P,\ K,\ T_C,\ T_P \leq 10^4,\ T_P \leq T_C$), indicating the number of previously running processes, the number of tasks your program must complete, the CPU time in milliseconds assigned to each process, and the time in milliseconds your program takes to complete one task, respectively.

Output

For each test case, print a line with a number indicating the expected time in seconds to complete all tasks. For each query, your answer is considered correct if its absolute or relative error does not exceed 10^{-4}.

Example

Input	Output
3 1 1 60 30 2 5 60 30 4 4 50 20	0.09 0.75 0.88

Use fast I/O methods