The lazy programmer

A new web-design studio, called SMART (Simply Masters of ART), employs two people. The first one is a web-designer and an executive director at the same time. The second one is a programmer. The director is so a nimble guy that the studio has already got **N** contracts for web site development. Each contract has a deadline $\mathbf{d_i}$.

It is known that the programmer is lazy. Usually he does not work as fast as he could. Therefore, under normal conditions the programmer needs $\mathbf{b_i}$ of time to perform the contract number **i**. Fortunately, the guy is very greedy for money. If the director pays him $\mathbf{x_i}$ dollars extra, he needs only $\mathbf{(b_i - a_i*x_i)}$ of time to do his job. But this extra payment does not influence other contracts. This means that each contract should be paid separately to be done faster. The programmer is so greedy that he can do his job almost instantly if the extra payment is $\mathbf{(b_i/a_i)}$ dollars for the contract number **i**.

The director has a difficult problem to solve. He needs to organize programmer's job and, may be, assign extra payments for some of the contracts so that all contracts are performed in time. Obviously he wishes to minimize the sum of extra payments. Help the director!

Input

First line of the input contains an integer **t** ($1 \le \mathbf{t} \le 45$), equal to the number of testcases. Then descriptions of **t** testcases follow.

First line of description contains the number of contracts **N** ($1 \le \mathbf{N} \le 100000$, integer). Each of the next **N** lines describes one contract and contains integer numbers $\mathbf{a_i}$, $\mathbf{b_i}$, $\mathbf{d_i}$ ($1 \le \mathbf{a_i}, \mathbf{b_i} \le 10000$; $1 \le \mathbf{d_i} \le 1000000000$) separated by spaces.

At least *90%* of testcases will have $1 \le \mathbf{N} \le 10000$.

Output

For each testcase in the input your program should output one line with a single real number S. Here S is the minimum sum of money which the director needs to pay extra so that the programmer could perform all contracts in time. The number must have two digits after the decimal point.

Example

Input:
1
2
20 50 100
10 100 50

Output:
5.00