Shortest Circuit Evaluation

Short circuit evaluation of Boolean expressions denotes the semantic in which the second argument of some Boolean operator is not evaluated if the value of the first argument is enough to have the result. This technique is used in many programming languages to optimize the evaluation of Boolean expressions. Specifically for "A and B", if A is false we know that the whole expression is false and we don't need to evaluate B. For "A or B", if A is true we know the result to be true. Now having that those Boolean operations are commutative we may actually evaluate B first and not evaluate A in case B gives us the result. Moving the idea further if we have "A1 or A2 or...or An" we can evaluate the variables in any order and as soon as we have one of them as true we know that the whole expression is true. We can do similarly for and operation. Now let's consider some complex Boolean expression. We will fix the order in which we will evaluate the variables of the expression. Then we evaluate those variables in that order and we won't evaluate the variables that give us no new information about the value of the whole expression in the process. For example, assume we have "A and B or C" and we fix the order of evaluation B, A, C. First we evaluate B, if it's false we don't have to evaluate A and only evaluate C. However if B is true we will need to evaluate A. If A is true we know the expression is true and won't evaluate C, otherwise we evaluate C to have the value of the expression. Now your task is having some complex Boolean expression containing and, or, not operations and for each variable having the probability that this variable is true, you need to find the order of evaluation for which the expected number of evaluations in the process described above will be minimal.

Input

The first line of input file contains number t - the amount of test cases. Then t test cases follow. The first line of each test case will contain the Boolean expression. The expression will be valid and will consist of and, or, not operations, brackets and variable names. All the variable names in one expression will be distinct. Then for each variable present in the expression there will be a line in the input in the format s p, where s - the name of the variable and p - is the probability that the variable will be true. The names of the variables will consist of small Latin letters only.

Constraints

$1 <= t <= 50$
$0 < p < 1$
The length of the expression won't exceed 30000 characters.
There will be no more than 1000 variable in the expression.
The length of the variable names won't exceed 5 characters.
Also the expression will be in the form of either conjunctive or disjunctive normal form.

Output

For each test case output the expected number of evaluations for the optimal order of evaluation of variables for short circuit evaluation process described above. Output the answer with 6 digits after the dot.

Example

Input:
3
(a and b) or c
a 0.3
b 0.4
c 0.5
(a or b) and (not d or c)
a 0.5
b 0.3
c 0.8
d 0.25
ab or bc or cd
ab 0.3
bc 0.1
cd 0.2

Output:
1.650000
2.280000
2.260000

Explanation

In the first test case the best order is c, a, b.