ACM-ICPC Live Archive

8187 Matrix Multiplication

Winning an ICPC regional has expanded your employment opportunities. The keywords “ICPC Finalist” have been flagged by the resume filters set up by *Numerical Concoctions, Inc.*, an early stage startup aiming to disrupt the market for loud-based linear algebra services. As a result you have been offered a position as a “Matrix Ninja Intern”, working on the next generation matrix multiplication library functions.

When it comes to matrix multiplications, the company has already solved the problem for $N = 2$. Their function $Mul(A, B)$ for multiplying two matrices is dominating all benchmarks. The company’s next goal is to be the best at multiplying N matrices in a row for any N. The plan is to leverage the existing technology and simply call $Mul(A, B)$ $N - 1$ times.

As you recall from Linear Algebra 101, the product of two matrices $A[n, m]$, and $B[m, l]$, is a matrix $X[n, l]$. Things become more interesting with three or more matrices, because even though matrix multipliation is associative, that is, $(A \times B) \times C = A \times (B \times C)$, different ways of computing the final product require different amounts of intermediate storage. For example, if we have $A[1000, 2]$, $B[2, 1000]$, and $C[1000, 2]$, computing $A \times B$ first requires a matrix with $1000 \times 1000 = 1000000$ elements to be allocated, whereas computing $B \times C$ first requires a matrix with only $2 \times 2 = 4$ elements to be allocated.

Your first assignment at your new job is to implement a helper function that, given the sizes of the N matrices to be multiplied together, determines how to compute the product allocating as little intermediate storage as possible. Assume that all intermediate storage cannot be reused and that it will be deallocated only after the final product is computed.

Input

For this problem, the input will consist of lines describing individual test cases in the $[n_1, m_1] * [n_2, m_2] * \ldots * [n_N, m_N]$ format, where $2 \leq N \leq 1000$, $1 \leq m_i \leq 10000$, and $1 \leq n_i \leq 10000$.

Output

For each input line, your program is to print the number of elements that need to be allocated as intermediate storage. Your program is not to count the input matrices or the final product as intermediate storage. No leading or trailing whitespace or leading zeroes or signs are to appear on output lines.

Sample Input

```
[1000,2]*[2,1000]*[1000,2]
[20,30]*[30,20]
[2,1]*[1,1]*[1,1]*[1,1]*[1,2]
[22,22]*[22,55]*[55,76]*[76,29]
```

Sample Output

```
4
0
4
2233
```