Problem 1. LongInt

(Time Limit: 1 second)

Problem Description

The range of an integer variable is determined by two factors: its size (in bits), and its sign, which can be “signed” or “unsigned”. Integer data type (unsigned short int) represents only to the largest positive integer 65535, even with (unsigned long int) only 4294967295, it is not enough, and we must be able to write a long integer sum. Trying to figure out how making it capable of dealing with integers greater than 4294967295.

Input Format

There are several test cases. In each test case, each row represents one input VeryLongInt (are all positive), the last line contains only one 0, ending representatives. Each string max length= 30.

Output Format

All VeryLongInt sum output.

Example

Sample Input:	Sample Output:
123456 123456 0 785236941 123456789 121212121212 0	246912 122120814942

Problem 2. Odd-Even Tree

(Time Limit: 1 second)

Problem Description

Let $T = (V, E)$ be a tree graph, i.e., an undirected connected graph without cycle. Each of the nodes is assigned either "Even" or "Odd", and the nodes are called even or odd nodes, respectively. This problem asks for a maximum cardinality subset F of E such that $d(F, v)$ is even for each even node and $d(F, v)$ is odd for each odd node, where $d(F, v)$ is the degree of node v in the subgraph $Y = (V, F)$.

Technical Specification

- The number of test cases is at most 16.
- For each test case, the number of nodes N satisfies $1 \leq N \leq 50000$.

Input Format

The test file contains several test cases. The first line contains an integer indicating the number of test cases. The first line of each test case is the number of nodes N. The nodes are labelled by an integer from 0 to $N - 1$. The next line starts with the number K of even nodes and followed by K labels of the even nodes. Starting from the third line, there are $N - 1$ lines, and each line contains two integers v and then $p[v]$, which means $p[v]$ is the parent of v.

Output Format

For each test case, output the maximum number of edges satisfying the requirement in one line. If there is no solution, output -1.

Example

Sample Input:	Sample Output:
3 2 2 1 0 0 1 2 0 1 0 2 1 1 1 0	0 1 -1

Problem 3. Edit Distance

(Time Limit: 3 second)

Problem Description

Given two strings s_1 and s_2, find the minimum number of editing operations (i.e., insert, delete, or replace a char at each operation) required to convert s_1 into s_2. For instance, given $s_1 =$ "Sunday" and $s_2 =$ "Saturday", the minimum edit distance is 3 since we can convert s_1 into s_2 by replacing 'n' (in s_1) with 'r', insert t, and finally insert a.

Technical Specification

- The length of s_1 and s_2 is between 1 to 10000.
- The characters are within standard alphabet {a~z}.

Input Format

The first line contains one integer indicating the number of test cases (≤ 10). Each of the following two lines represent one test case, where the first line is s_1 and second line is s_2.

Output Format

For each test case, output the minimum number of edit operations at each line.

Example

Sample Input:	Sample Output:
2 Sunday Saturday alignment algorithm	3 7

Problem 4. MMC

(Time Limit: 2 second)

Problem Description

MMC does not stand for M&M's Chocolate; instead, it is short for Minimum Mean Cycle, one of the basic building blocks for many important graph algorithms. You are given a weighted directed graph G with n vertices and m edges. G is strongly connected, that is, there exists a path from u to v for every u and v. G does not contain a self-loop, an edge that connects a vertex to itself. For every ordered pair (u, v) there is at most one edge from u to v in G. An MMC of G is a directed cycle with minimum average edge weight.

Input Format

The first line contains an integer T indicating the number of graphs. Each graph starts with a line containing two integers n and m and is followed by m lines listing the edges. Each edge is described by three integers u, v and w indicating that there is an edge of weight w from u to v.

Output Format

For each graph, output the average edge weight of an MMC. Your answer will be considered correct if and only if the relative or absolute error does not exceed 10^{-6}.

Technical Specification

- $1 \leq T \leq 40$
- $2 \leq n \leq 10^3$
- $n \leq m \leq 10^4$
- $1 \leq u, v \leq n\ (u \neq v)$
- $|w| \leq 10^9$

Example

Sample Input:	Sample Output:
1 5 8 1 3 10 4 3 6 2 1 5 3 2 3 2 3 20 3 5 1 5 4 6 5 2 8	4.333333333333

Problem 5. Longest Common Prefix Array

(Time Limit: 1 second)

Problem Description

In pattern matching, the longest common prefix (LCP) array is an auxiliary data structure to the suffix array for improving matching performance. Suffix array represents the lexicographic rank of all suffixes of a text, whereas LCP array contains the maximum length prefix match between two consecutive suffixes (after sorted lexicographically).

For instance, given a text "banana", the original indices of all suffixes $\{\text{banana}, \text{anana}, \text{nana}, \text{ana}, \text{na}, \text{a}\}$ are $\{0, 1, 2, 3, 4, 5\}$. The corresponding suffix array (called $\text{suffix}[i]$) is $\{5, 3, 1, 0, 4, 2\}$, which corresponds to the lexicographically-sorted suffixes $\{\text{a}, \text{ana}, \text{anana}, \text{banana}, \text{na}, \text{nana}\}$. On the other hand, the LCP array $\text{lcp}[i]$ indicates length of the LCP of two suffixes indexed by $\text{suffix}[i]$ and $\text{suffix}[i+1]$, where i ranges from 0 to $n-1$ (n: length of text).

Following the above example of suffix array, the corresponding LCP array (called $\text{lcp}[i]$) is $\{1, 3, 0, 0, 2, 0\}$. $\text{lcp}[1] = 1$ since the length of longest common prefix between $\text{suffix}[1]$ (i.e., "a") and $\text{suffix}[2]$ (i.e., "ana") is 1. $\text{lcp}[2] = 3$ since the length of longest common prefix between $\text{suffix}[2]$ (i.e., "ana") and $\text{suffix}[3]$ (i.e., "anana") is 3. Note that $\text{lcp}[n-1]$ is always defined to be zero as there is no suffix after $\text{suffix}[n-1]$. Given a text string, compute the sum of absolute differences between any two adjacent values in the LCP array. For instance, the sum of absolute differences of the above LCP array is $-(1-3) + (3-0) + (0-0) + -(0-2) + (2-0) = 9$.

Technical Specification

- The length of text n ranges from 5 to 100000.
- Each character in the text is from the standard alphabet {a~z}.
- The number of test cases is equal or less than 35.

Input Format

The first line contains the number of test cases. Each of the following lines

represents a test case which stores the text string.

Output Format

The output should consist of one line for each test case. Each line contains an integer indicating the sum of absolute differences between two adjacent indices in the LCP array.

Example

Sample Input:	Sample Output:
2 banana mississippi	9 17