Problem A: Airlines

Task code: AVIA

There are **M** airports and **K** airline companies in a country. The **K** companies operate a total of **N** direct flights between the **M** airports. Naturally, all the flights carry passengers in both directions, and every flight is operated by one company only. There may be several direct flights between any two airports, but each company runs at most one of these. For every pair of airports a passenger can fly from one to another using either a direct flight or through several intermediate airports.

An officer of the aviation administration has to evaluate the quality of service of the airlines. For the evaluation, the officer has to fly as a passenger once on each of the **N** flights (in one direction only). To avoid any suspicion of cooperation between the evaluator and the airlines, the evaluator is not allowed to fly on two flights of the same company one after another. The first and the last flights should be operated by different companies too.

The officer wants to eliminate any unnecessary expenses and plan the route so that:

- the evaluation route starts from the airport 1;
- each subsequent flight starts in the same airport where the previous one finishes;
- the evaluation route finishes at the airport 1.

Write a program to help him plan the route!

Input data. The first line of the text file AVIA.IN contains three integers **M, N,** and **K** ($2 \le M \le 1{,}000$; $1 \le N \le 10{,}000$; $1 \le K \le 10$). Each of the following **N** lines describes one flight. Each line contains three integers: the numbers of the airports the flight connects and the number of the company that operates the flight. The airports are numbered from 1 to **M** and the companies from 1 to **K**. The flights are numbered from 1 to **N** in the order in which they appear in the input file.

Output data must be written to the text file AVIA.OUT. If there is no solution for some reason, output the word 'No' on the first and only line of the file. If there is a solution, the first line of the file should contain the word 'Yes', and each of the following **N** lines should contain the numbers of the flights in the order in which the evaluator tests them. If there are several solutions, output any one of them.

Input data examples	Output data examples
4 7 3 3 2 1 1 2 2 3 4 1 3 4 2 3 4 3 1 2 3 2 4 2	Yes 2 1 5 4 3 7 6
4 7 2 3 2 1 1 2 1 3 4 1 3 1 2 3 1 1 4 2 1 4 2 2	No

Problem B: Red and White Balls

Task code: BALLS

You have an unlimited supply of red and white balls. The balls of the same color have the same weight and red ones are heavier than the white ones.

You need to lay out a line of **N** balls such that every ball on an odd-numbered position is no heavier and every balls on an even-numbered position is no lighter than its immediate neighbors. In other words, you need to build a sequence a_1, a_2, …, a_N, where a_i is the weight of the ball on the position **i** so that the inequality

$$a_1 \le a_2 \ge a_3 \le a_4 \ge \ldots$$

holds. How many different sequences can you build?

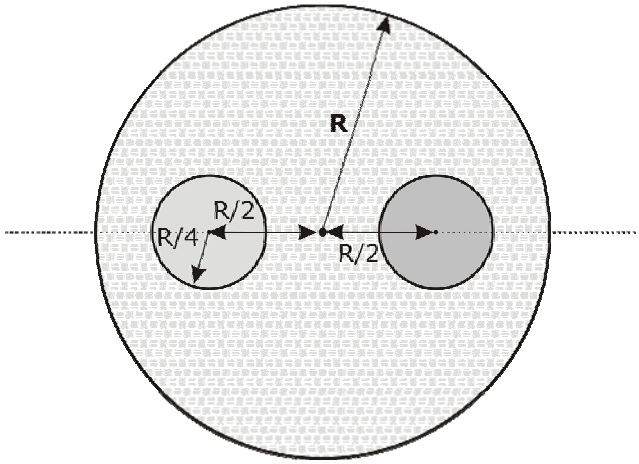
Input data are given in the text file BALLS.IN. The first and only line of the file consists of the integer **N** ($1 \le \mathbf{N} \le 50{,}000$).

Output data must be written to the file BALLS.OUT. The first and only line of the file must consist of the number of possible sequences.

Input data examples	Output data examples	Explanation
1	2	The possible sequences are: W, R
3	5	The possible sequences are: WWW, RRR, WRW, WRR, RRW

Problem C: The Bob

Task code: BOB



A spherical bob with the radius **R** has the density **P**. Inside the bob are two spherical voids of the radius **R/4**, filled with substances of density $\mathbf{P_1}$ and $\mathbf{P_2}$, respectively. The centers of these voids and the center of the bob are on a straight line and the distance form the center of each void to the center of the bob is **R/2**.

A cylindrical glass with the radius $\mathbf{R_0}$ is first filled with a liquid of density $\mathbf{P_0}$ up to the level **H**, and then the bob is dropped into the glass. What is the new level of the liquid in the glass, assuming the glass is tall enough and does not overflow? You should ignore any physical effects except for the Archimedes' principle.

Input data The first and only line of the text file BOB.IN contains the positive real numbers **R, P,** $\mathbf{P_1}$, $\mathbf{P_2}$, $\mathbf{R_0}$, $\mathbf{P_0}$, and **H**, not exceeding 10,000 and given with at most three places after the decimal point.

It is also known that $\mathbf{P_1} \neq \mathbf{P_2}$, $\mathbf{P} < \max(\mathbf{P_1}, \mathbf{P_2})$, $\mathbf{R_0} > \mathbf{R}$.

Output data The first and only line of the text fie BOB.OUT should consist of a single number – the final level of the liquid in the glass with the precision up to 0.0001.

Input data examples	Output data examples
5 3 8 2 7 3 20	23.4014
5 3 8 2 7 3 3.45	5.3073

Problem D: Bikers

Task code: BIKERS

There are **N** ($2 \le N \le 1{,}000$) cities in a country. The cities are connected by **M** ($1 \le M \le 10{,}000$) two-way roads. There may be any number of roads between any two cities. A group of bikers from the city **1** plans to attend a bikers' meeting held in the city **N** and return home after that. The cost for a single use of the road **I** ($1 \le I \le M$) is a_I ($1 \le a_I \le 100$). However, because of the noise, the aggressive driving style, and the permanent disregard for code of conduct of the bikers, using the same road twice may cause them trouble. For some roads, using the road **I** for the second time will result in additional costs (fines, bribes, etc.) of b_I ($0 \le b_I \le 100$), and some roads can't be used twice at all as there the bikers would face jail time…

Can you find the minimal possible cost of the trip?

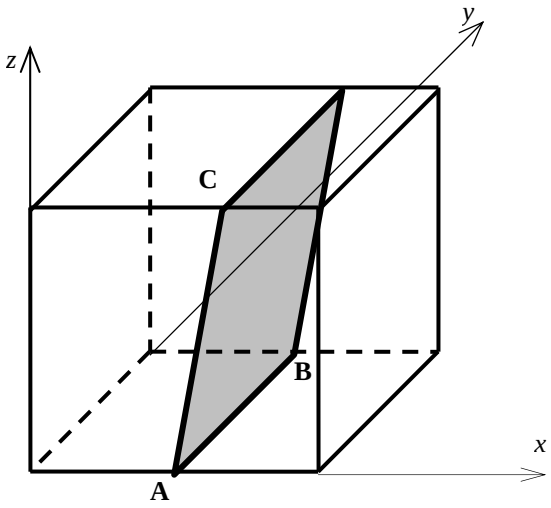
Input data are given in the text file BIKERS.IN. The first line of the file contains the values **N** and **M**. Each of the following **M** lines describes one road and contains the numbers of the cities connected by the road, and the costs a_I and b_I, where $b_I = -1$ indicates that the road can't be used twice.

Output data must be written to the text file BIKERS.OUT. The first and only line of the file should contain the minimal possible total cost of the round-trip. If the trip is not possible for any reason, output zero as the cost.

Input data examples	Output data examples
6 8 1 2 10 1 1 3 5 -1 2 3 10 10 3 4 10 10 4 5 10 10 2 5 2 0 5 6 10 1 4 6 5 -1	42
2 1 1 2 10 -1	0

Problem E: Cube Intersection

Task code: CUBE



Write a program to compute the area of the intersection of a unit cube with the plane given by the three different points **A**, **B**, and **C**. It is known that the points are located on the edges of the cube, but not on the same line. The intersection may coincide with a face of the cube.

Input data. The text file CUBE.IN contains three lines, each of which describes the coordinates of one of the points **A**, **B**, and **C**. On each line, two out of the three coordinate values are either 0 or 1, and the remaining one is in the interval [0; 1]. All the values are given with at most 3 digits after the decimal point.

Output data. The single line of the output file CUBE.OUT must contain the area of the intersection with the precision up to 0.0001.

.

Input data examples	Output data examples
0.5 0 0 0.5 1 0 0.7 0 1	1.0198
0 0 0 0 0.01 0 0 0 0.01	1.0000

Problem F: Restoring Communication Links

Task code: LINK

The civil war that broke up the First Human Empire has finally ended… The mankind is spread out over several planets and has lost connections between those planets. Of course, restoring the communication links is now one of the most important tasks!

The engineers have discovered and repaired **M** *sub-space communicators* which were built at the times of the First Empire. The devices are capable of instantaneously transmitting messages from one to another. Each communicator has several *ports* (the number of ports of one communicator doesn't exceed **M-1** and can vary from device to device). Two ports of two distinct communicators may be tuned to each other, and then a sub-space communication link is established between the two communicators. At most one link may be established between different communicators. A communicator can be turned on only after all its ports are tuned.

Unfortunately, the communicators were repaired only partially, and from time to time the communication links fail. Naturally, the users want to be able to exchange messages between any two communicators (possibly using other communicators as intermediate waypoints). They also want this requirement to be satisfied even when up to **K** communication links fail simultaneously!

Is it possible to build a network that satisfies this reliability requirement?

Input data. The first line of the text file LINK.IN contains two integers, **M** and **K** ($1 \le \mathbf{K} < \mathbf{M} \le 1{,}000$). Each of the following **M** lines contains the number of ports of one communicator. The communicators are numbered from 1 to **M** in the order in which they are described in the input file.

Output data must be written to the text file LINK.OUT. If it is impossible to build a network that satisfies the users' requirements, output the word 'No' on the first and only line of the file. If the network can be built, the first line of the file should contain the word 'Yes' and each of the following lines should contain two integers – the numbers of two communicators with a direct link between them. The number of these pairs should be equal to the total number of ports on all communicators, divided by 2. If there are several solutions, output any one of them.

Input data examples	Output data examples
6 2 3 3 3 5 3 3	Yes 1 2 2 3 3 6 4 5 4 1 4 2 4 3 4 6 5 1 5 6
4 2 2 3 3 2	No

Problem G: Palindromes

Task code: PALIND

A string is a *palindrome* if it reads the same in both directions, e.g. `'bob'`, `'rotor'`. There is a non-empty string **S**. How many different palindromes of the length **K** does **S** contain? We say two palindromes are different when they start from different positions.

The length of **S** does not exceed 30,000 symbols. **S** consists of Latin letters only. Lowercase and uppercase letters differ (the palindromes are case sensitive).

Input data. The first line of the text file PALIND.IN contains **K** ($2 \le \mathbf{K} \le 30{,}000$). The second line contains **S**. **K** does not exceed the length of **S**.

Output data. The first and only line of the text file PALIND.OUT should consist of a single number – the number of palindromes found.

Input data examples	Output data examples
5 `rororo`	2
3 `babcbab`	3

Problem H: S-transformations

Task code: S_TRANS

You are given a sequence $a_1, \ldots, a_n$, consisting of integers whose absolute values do not exceed 1,000,000. We define the *S-transformation* of depth k $(1 \le k < n)$ as an operation that replaces the elements $a_1, a_2, \ldots, a_k$ with the values $a_{k+1}-a_k$, $a_{k+1}-a_{k-1}, \ldots$, $a_{k+1}-a_1$.

Using a chain of S-transformations $(S_1, \ldots, S_P)$, where S_i means the depth of the transformation with number i, you need to convert the sequence $a_1, \ldots, a_n$ into a new sequence $b_1, \ldots, b_n$ such that

$$b_i \ge \frac{b_{i-1}+b_{i+1}}{2}, \quad i = 2,\ldots n-1 \qquad (*)$$

Constraints as follows should be met

1. $P \le 2 \cdot n - 1$;
2. For any T, $1 \le T < P$, subchain $(S_1, \ldots, S_T)$ **does not** convert $a_1, \ldots, a_n$ into a sequence with satisfies condition (*);
3. If the initial sequence $a_1, \ldots, a_n$ already satisfies condition (*), no S-transformations are required (lets assume $P = 0$ in such a case)

You need to build such a chain $(S_1, \ldots, S_P)$.

Input data. The first line of the text file S_TRANS.IN contains the integer n ($3 \le n \le 5{,}000$). Each of the following n lines contains one integer – the elements of the initial sequence.

Output data must be written to the text file S_TRANS.OUT. The first line of the file should contain the integer P – the number of S-transformations needed. Each of the following P lines should contain the depth of one of the transformations – in the order in which the transformations should be applied. If there are several solutions, output any one of them.

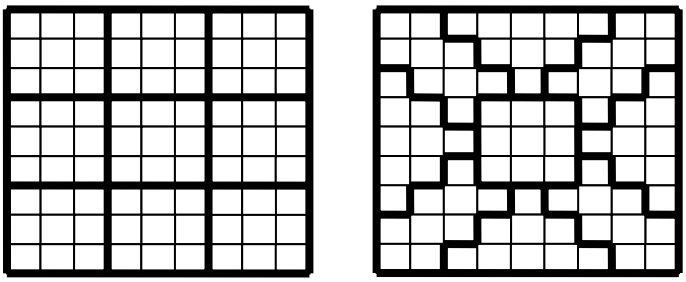
Input data examples	Output data examples
5 2 2 2 2 6	2 3 4
5 2 2 2 2 2	0

Problem I: Unusual Sudoku Task code: SUDOKU

Sudoku is a well-known puzzle with the following rules. The digits 1 to 9 have to be laid out in a 9×9 square so that each digit is present exactly once in every row, every column and each of the nine 3×3 sub-squares. Some of the cells are pre-filled and those may not be changed.

In addition to the classic version described above, several non-standard variations can be considered. In one of the variations, the 9×9 square is divided into 9 continuous regions of 9 cells each that may have any shape. The numbers still have to be laid out so that each of them appears exactly once in every row, every column and every region.

In the figure below, the standard Sudoku field is given on the left, and one of the possible non-standard fields on the right.



The task is to write a program to solve the puzzle for any non-standard field.

Input data. The text file SUDOKU.IN consists of 18 lines, each containing exactly 9 characters. The first 9 lines describe the structure of the field. Each cell is marked with a Latin letter from A to I so that the cells belonging to the same region are marked with the same letter and the cells belonging to different regions are marked with different letters. The following 9 lines describe the initial state of the field, where digits indicate pre-filled fields and points indicate initially empty fields.

Output data must be written to the text file SUDOKU.OUT. The file must consist of 9 lines, each containing exactly 9 digits – the completed puzzle. If the puzzle in the input file has several solutions, output any one of them. If the puzzle in the input file has no solutions, output a single line containing the word 'impossible'.

Input data example	Output data example
AABBBBBCC AAABBBCCC DAAABCCCE DDAFFFCEE DDDFFFEEE DDGFFFIEE DGGGHIIIE GGGHHHIII GGHHHHHII 9..8.7..3 ...2.9... ..7...6.. 26.....89 .... 17.....65 ..5...8.. ...4.5... 4..7.3..6	926857413 653219748 587132694 264371589 849526137 172948365 315694872 738465921 491783256