上海大学
SHANGHAI UNIVERSITY

2020 ICPC

The 45th ICPC Asia Regional Programming Contest Shanghai Site

第45届ICPC国际大学生程序设计竞赛亚洲区域赛（上海）

2020年12月12-13日

正式赛试题册

Problem A. Wowoear

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Wowo is a solo adventurer who completed many dangerous journeys on his own foot in forests, deserts and even glaciers. The Shanghai ICPC (Shanghai Invitational Contest on Programmable Cheating) committee invited Wowo as a tester of their new running trial.

The trial can be described as a 2D simple polyline $(p_1, \ldots, p_n)$. In other words, the trial consists of $n-1$ line segments $(p_1, p_2), \ldots, (p_{n-1}, p_n)$. The line segments do not intersect with each other except that two consecutive line segments (p_i, p_{i+1}) and (p_{i+1}, p_{i+2}) intersect at the point p_{i+1}. Any two consecutive segments have different directions. The committee wants Wowo to run from p_1 to p_n along the line segments $(p_1, p_2), \ldots, (p_{n-1}, p_n)$ in order.

However, Wowo has a smart device that can hack the committee's system for an interval of time. Wowo is able to choose 2 points a, b on the trial and run directly from a to b along the line segment (a, b). Each of these a and b can be some p_i $(1 \le i \le n)$ and can be some point on some line segment (p_i, p_{i+1}) $(1 \le i < n)$ as well. Before reaching a and after reaching b, Wowo has to run along the original trial. Wowo does not want to be caught cheating, so he decided that the line segment (a, b) should not intersect or touch any line segment of the trial at any point other than a and b. Help Wowo to choose a and b wisely and output the shortest distance Wowo need to run from p_1 to p_n using his smart cheating device.

Input

The first line includes a single integer n indicating the number of points on the running trial $(2 \le n \le 200)$.

The $i+1$-th line $(1 \le i \le n)$ contains two integers x and y separated by a single space indicating the coordinates of p_i $(-1000 \le x, y \le 1000)$.

It is guaranteed that the line segments do not intersect with each other except that two consecutive line segments (p_i, p_{i+1}) and (p_{i+1}, p_{i+2}) intersect at the point p_{i+1}. In other words, $(p_i, p_{i+1}) \cap (p_j, p_{j+1}) = \begin{cases} \emptyset & i \ne j-1 \\ \{p_j\} & i = j-1 \end{cases}$ holds for any integers i, j satisfying $1 \le i < j < n$. Here (s, t) represents all points on the line segment from s to t including s and t.

It is guaranteed that each line segment has nonzero length. In other words, $p_i \ne p_{i+1}$ for any integer $i \in [1, n)$.

It is guaranteed that adjacent line segments are not collinear. In other words, for any integer $i \in [1, n-2]$ and any real number λ, $p_i - p_{i+1}$ is **not** equal to $\lambda(p_{i+1} - p_{i+2})$.

Output

Output the shortest distance Wowo needs to run. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

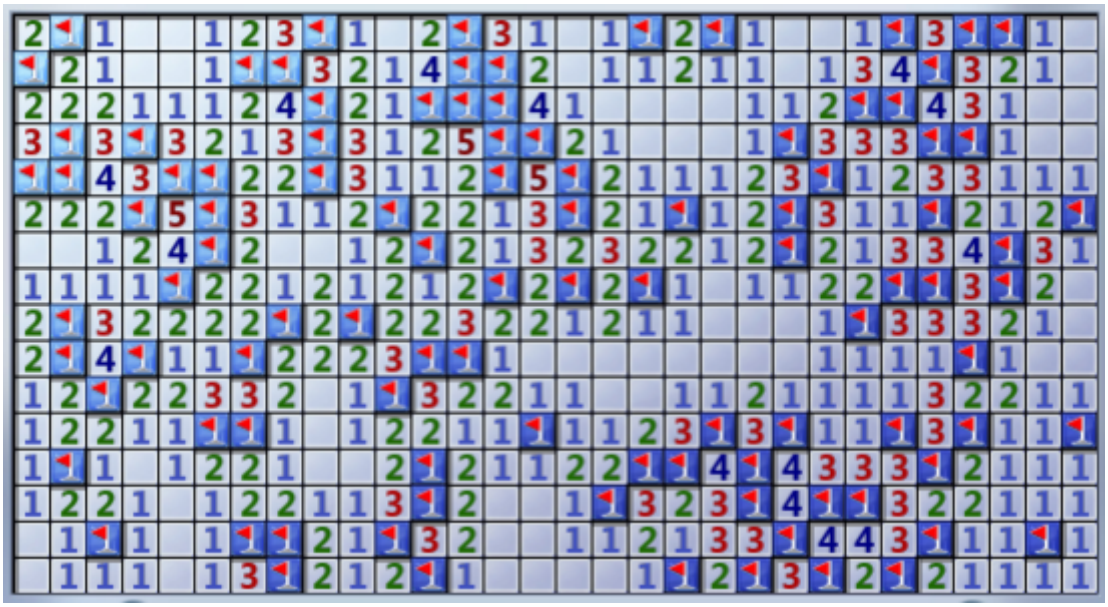
Example

standard input	standard output
5 0 0 1 10 2 0 3 10 4 0	22.099751242242

Problem B. Mine Sweeper II

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

A mine-sweeper map X can be expressed as an $n \times m$ grid. Each cell of the grid is either a mine cell or a non-mine cell. A mine cell has no number on it. Each non-mine cell has a number representing the number of mine cells around it. (A cell is around another cell if they share at least one common point. Thus, every cell that is not on the boundary has 8 cells around it.) The following is a 16×30 mine-sweeper map where a flagged cell denotes a mine cell and a blank cell denotes a non-mine cell with number 0.



Given two mine-sweeper maps A, B of size $n \times m$, you should modify at most $\lfloor \frac{nm}{2} \rfloor$ (i.e. the largest nonnegative integer that is less than or equal to $\frac{nm}{2}$) cells in B (from a non-mine cell to a mine cell or vice versa) such that the sum of numbers in the non-mine cells in A and the sum of numbers in the non-mine cells in B are the same. (If a map has no non-mine cell, the sum is considered as 0.)

If multiple solutions exist, print any of them. If no solution exists, print "-1" in one line.

Input

The first line contains two integers n, m $(1 \leq n, m \leq 1000)$, denoting the size of given mine-sweeper maps.

The i-th line of the following n lines contains a length-m string consisting of "." and "X" denoting the i-th row of the mine-sweeper map A. A "." denotes for a non-mine cell and an "X" denotes for a mine cell.

The i-th line of the following n lines contains a length-m string consisting of "." and "X" denoting the i-th row of the mine-sweeper map B. A "." denotes for a non-mine cell and an "X" denotes for a mine cell.

Output

If no solution exists, print "-1" in one line.

Otherwise, print n lines denoting the modified mine-sweeper map B. The i-th line should contain a length-m string consisting of "." and "X" denoting the i-th row of the modified map B. A "." denotes for a non-mine cell and an "X" denotes for a mine cell.

Please notice that you need not print the numbers on non-mine cells since these numbers can be determined by the output mine-sweeper map.

Example

standard input	standard output
2 4 X..X X.X. X.X. .X..	X.XX .X..

Note

We modify one cell in B. Then the sums of the numbers on non-mine cells in A and B both equal 10.

Problem C. Sum of Log

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Given two non-negative integers X and Y, determine the value of

$$\sum_{i=0}^{X} \sum_{j=[i=0]}^{Y} [i\&j = 0]\lfloor \log_2(i+j) + 1 \rfloor$$

modulo $10^9 + 7$ where

- & denotes bitwise AND;
- $[A]$ equals 1 if A is true, otherwise 0;
- $\lfloor x \rfloor$ equals the maximum integer whose value is no more than x.

Input

The first line contains one integer T ($1 \le T \le 10^5$) denoting the number of test cases.

Each of the following T lines contains two integers X, Y ($0 \le X, Y \le 10^9$) indicating a test case.

Output

For each test case, print one line containing one integer, the answer to the test case.

Example

standard input	standard output
3 3 3 19 26 8 17	14 814 278

Note

For the first test case:

- Two (i, j) pairs increase the sum by 1: $(0, 1), (1, 0)$
- Six (i, j) pairs increase the sum by 2: $(0, 2), (0, 3), (1, 2), (2, 0), (2, 1), (3, 0)$

So the answer is $1 \times 2 + 2 \times 6 = 14$.

Problem D. Walker

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

As a world-famous traveler, Prof. Pang's research interest is to travel as many places as possible in his life.

We have a segment $[0, n]$. There are two travelers on it. The first one is on position p_1 with velocity v_1 (which means s/he can walk v_1 unit on the segment per second). The second one is on position p_2 with velocity v_2.

From their respective beginning points, travelers can walk on the segment. They cannot walk outside the segment. Whenever they want to change their direction, they can turn around immediately.

Please help Prof. Pang to calculate the minimum possible time by which every position of the segment is passed by at least one traveler.

Input

The first line contains one integer *test* ($1 \le test \le 10000$) – the number of test cases.

The i-th of the next *test* lines contains five numbers $n, p_{1,i}, v_{1,i}, p_{2,i}, v_{2,i}$ ($0 < n \le 10000$, $0 \le p_{1,i}, p_{2,i} \le n$, $0.001 \le v_{1,i}, v_{2,i} \le 1000$). All numbers have at most 3 digits after the decimal point.

Output

For each test case, we should output one number – the minimum time that every position of the segment is passed by at least one traveler.

Your answer is considered correct if its absolute or relative error does not exceed 10^{-6}.

Example

standard input	standard output
2 10000.0 1.0 0.001 9999.0 0.001 4306.063 4079.874 0.607 1033.423 0.847	5001000.0000000000 3827.8370013755

Problem E. The Journey of Geor Autumn

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Once upon a time, there was a witch named Geor Autumn, who set off on a journey across the world. Along the way, she would meet all kinds of people, from a country full of ICPC competitors to a horse in love with dota—but with each meeting, Geor would become a small part of their story, and her own world would get a little bit bigger.

Geor just arrived at the state of Shu where people love poems. A poem is a permutation $(a_1, \ldots, a_n)$ of $[n]$. ($(a_1, \ldots, a_n)$ is a permutation of $[n]$ means that each a_i is an integer in $[1, n]$ and that $a_1, \ldots, a_n$ are distinct.) One poem is *good* if for all integer i satisfying $i > k$ and $i \leq n$, $a_i > \min(a_{i-k}, \ldots, a_{i-1})$. Here $\min(a_{i-k}, \ldots, a_{i-1})$ denotes the minimum value among $a_{i-k}, \ldots, a_{i-1}$.

Help Geor calculate how many good poems there are, given n and k. To avoid huge numbers, output the answer modulo 998244353.

Input

The first line contains two integers n and k separated by a single space ($1 \leq n \leq 10^7$, $1 \leq k \leq 10^7$).

Output

Output only one integer in one line—the number of good poems modulo 998244353.

Examples

standard input	standard output
1 1	1
2 3	2
3 2	4
4 2	10

Problem F. Fountains

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	1024 megabytes

Suppose you and your teammate Mixsx will attend the Namomo Camp. The Namomo Camp will happen in n consecutive days. We name the i-th day as day i ($1 \le i \le n$). The cost of day i is s_i.

Unfortunately, the schedule of the Namomo Camp conflicts with Mixsx's final exams. Mixsx has final exams every day between day L and day R. The exact value of L and R have not been announced by his college so we assume that every pair of integers L and R satisfying $1 \le L \le R \le n$ will be chosen with probability $1/(n(n+1)/2)$. He decides to take all the exams and thus be absent from the Namomo Camp from day L to day R. His *loss* will be $\sum_{i=L}^{R} s_i$ in this case.

As Mixsx's teammate, you want Mixsx to give up his final exams and come back to the Namomo Camp. You can prepare k plans before L and R are announced. In the i-th plan ($1 \le i \le k$), you shut the electricity off to his college every day from day l_i to day r_i. You can choose the values of l_i and r_i as long as they are two integers satisfying $1 \le l_i \le r_i \le n$.

Once L and R are announced, you can choose a plan x ($1 \le x \le k$) such that $L \le l_x \le r_x \le R$. Then Mixsx will come back to the Namomo Camp on every day from day l_x to day r_x. His loss becomes $\sum_{i=L}^{R} s_i - \sum_{i=l_x}^{r_x} s_i$ in this case. You will choose a plan that minimizes Mixsx's loss. If no plan x satisfies $L \le l_x \le r_x \le R$, Mixsx will attend his final exams normally and his loss is $\sum_{i=L}^{R} s_i$.

Please calculate the minimum possible expected loss ans_k of Mixsx if you choose the k plans optimally. Output $ans_k \cdot n(n+1)/2$ for every k from 1 to $n(n+1)/2$.

Formally, given a list of n numbers s_i ($1 \le i \le n$), define a loss function $C(L, R) = \sum_{i=L}^{R} s_i$. Given an integer k ($1 \le k \le n(n+1)/2$), you should select $2k$ integers $l_1, \ldots, l_k, r_1, \ldots, r_k$ satisfying $1 \le l_i \le r_i \le n$ for all $1 \le i \le k$, such that

$$\sum_{1 \le L \le R \le n} \left[C(L, R) - \max_{1 \le i \le k, L \le l_i \le r_i \le R} C(l_i, r_i) \right]$$

is minimized. ($\max_{1 \le i \le k, L \le l_i \le r_i \le R} C(l_i, r_i)$ is defined as 0 if no i satisfies $1 \le i \le k$ and $L \le l_i \le r_i \le R$.) Output the minimized value for every integer k in $[1, n(n+1)/2]$.

Input

The first line contains an integer n ($1 \le n \le 9$). The second line contains n space separated integers s_i ($1 \le s_i \le 10^9$).

Output

The output contains $n(n+1)/2$ integers in their own lines, the expectations when $k = 1, \ldots, n(n+1)/2$ multiplied by $n(n+1)/2$. It can be shown that the results are always integers.

Examples

standard input	standard output
1 1	0
2 13 24	26 13 0
3 6 4 7	33 21 12 8 4 0

Note

For the first test case, we only need to consider the case $k = 1$. We can only choose $l_1 = r_1 = 1$. Then the expected loss is $C(1, 1) - C(1, 1) = 0$ and the result is $0 \times 1 \times (2)/2 = 0$.

For the third test case, consider the case when $k = 3$. We choose $l_1 = r_1 = 1$, $l_2 = r_2 = 3$ and $l_3 = 1, r_3 = 3$. The expected loss is 2. And the result is $2 \times 6 = 12$.

Problem G. Fibonacci

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

In mathematics, the Fibonacci numbers, commonly denoted as f_n, is a sequence such that each number is the sum of the two preceding numbers, starting with 1 and 1. That is, $f_1 = 1, f_2 = 1$ and $f_n = f_{n-2} + f_{n-1}$ $(n \geq 3)$.

Thus, the beginning of the sequence is $1, 1, 2, 3, 5, 8, 13, 21, \ldots$.

Given n, please calculate $\sum_{i=1}^{n} \sum_{j=i+1}^{n} g(f_i, f_j)$, where $g(x, y) = 1$ when $x \cdot y$ is even, otherwise $g(x, y) = 0$.

Input

The only line contains one integer n $(1 \leq n \leq 10^9)$.

Output

Output one number – $\sum_{i=1}^{n} \sum_{j=i+1}^{n} g(f_i, f_j)$.

Examples

standard input	standard output
3	2
10	24
100	2739

Problem H. Rice Arrangement

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Wowo is a hospitable Xinjiang uncle. k guests will have Uyghur Polo (a traditional Uyghur food) in Wowo's house around a big round table. n ($n \geq k$) chairs are placed around the table uniformly. Each guest sits on a chair and no two guests sit on the same chair. k bowls of Uyghur Polo are on the table. Each bowl is next to some chair (**with or without** some guest sitting on it). No two bowls locate at the same position.

As a waiter, you are supposed to assign each person with exactly one bowl of Uyghur Polo. The table can be rotated, so each time you can turn it $\frac{2\pi}{n}$ degrees clockwise or counterclockwise. The bowls turn with the table while the chairs and guests do not move. When one bowl of Uyghur Polo is in front of a guest, he can either take it or wait for another.

You want to minimize the total times of table rotating so that everybody can have meals as quickly as possible.

(Formal definition: The boundary of the table is a circle. n chairs are at n points on the circle whose convex hull is a regular polygon with n vertices. We name the points $0, \ldots, n-1$ in counterclockwise order. The i-th bowl is at point b_i ($0 \leq b_i < n$) initially. The i-th guest is at point a_i ($0 \leq a_i < n$) initially. If you turn the table counterclockwise, the bowl at point b_i ($1 \leq i \leq k$) will be moved to point $(b_i + 1) \bmod n$ after the rotation. If you turn the table clockwise, the bowl at point b_i ($1 \leq i \leq k$) will be moved to point $(b_i - 1) \bmod n$ after the rotation. ($x \bmod n$ is defined as the smallest nonnegative integer r such that $x - r$ is a multiple of n.))

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains two integers n, k ($1 \le n \le 10^9, 1 \le k \le \min(n, 1000)$) indicating the size of the table and the number of persons and bowls of Uyghur Polo.

In the second line, there are k integers $a_1, a_2, \ldots, a_k$ ($0 \le a_i < n$), indicating the positions of the persons. No two guests share the same position.

In the third line, there are k integers $b_1, b_2, \ldots, b_k$ ($0 \le b_i < n$), indicating the initial positions of the bowls. No two bowls of Uyghur Polo locate at the same position.

It is guaranteed that the sum of k over all test cases does not exceed 5000.

Output

For each test case, output the minimal total times of rotations such that each guest can have exactly one bowl of Uyghur Polo.

Examples

standard input	standard output
1 4 2 0 3 1 2	2
1 14 5 0 12 13 8 9 9 2 6 13 5	6

Problem I. Sky Garden

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Prof. Du and Prof. Pang plan to build a sky garden near the city of Allin. In the garden, there will be a plant maze consisting of straight and circular roads.

On the blueprint of the plant maze, Prof. Du draws n circles indicating the circular roads. All of them have center $(0, 0)$. The radius of the i-th circle is i.

Meanwhile, Prof. Pang draws m lines on the blueprint indicating the straight roads. All of the lines pass through $(0, 0)$. Each circle is divided into $2m$ parts with equal lengths by these lines.

Let Q be the set of the $n + m$ roads. Let P be the set of all intersections of two different roads in Q. Note that each circular road and each straight road have two intersections.

For two different points $a \in P$ and $b \in P$, we define $dis(\{a, b\})$ to be the shortest distance one needs to walk from a to b along the roads. Please calculate the sum of $dis(\{a, b\})$ for all $\{a, b\} \subseteq P$.

Input

The only line contains two integers n, m $(1 \le n, m \le 500)$.

Output

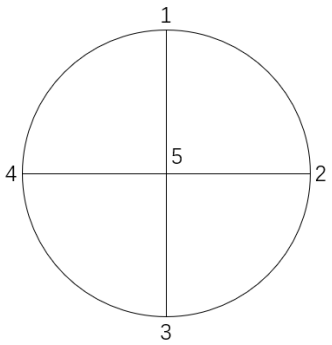
Output one number – the sum of the distances between every pair of points in P.

Your answer is considered correct if its absolute or relative error does not exceed 10^{-6}.

Examples

standard input	standard output
1 2	14.2831853072
2 3	175.4159265359

Note



$dis(p_1, p_2) = dis(p_2, p_3) = dis(p_3, p_4) = dis(p_1, p_4) = \frac{\pi}{2}$

$dis(p_1, p_5) = dis(p_2, p_5) = dis(p_3, p_5) = dis(p_4, p_5) = 1$

$dis(p_1, p_3) = dis(p_2, p_4) = 2$

Problem J. Octasection

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

At the Namomo Camp, a cute volunteer celebrates her birthday. Wowo buys her a huge cake. (The cake is so big that it has a 3D coordinate system inside.) There are n cuboid shaped pieces of chocolates **in** the cake. The i-th ($1 \le i \le n$) chocolate consists of all points (x, y, z) such that $min\_x[i] \le x \le max\_x[i], min\_y[i] \le y \le max\_y[i], min\_z[i] \le z \le max\_z[i]$. $min\_x, max\_x, min\_y, max\_y, min\_z, max\_z$ are 6 arrays of integers. Chocolates may overlap or touch each other.

The volunteer wants to distribute the cake to the campers of the Namomo Camp. To show off his knife skill, Wowo decides to cut the cake into pieces by exactly 3 cuts such that:

1. The first cut is a plane whose equation is $x = a$ for some integer a decided by Wowo.
2. The second cut is a plane whose equation is $y = b$ for some integer b decided by Wowo.
3. The third cut is a plane whose equation is $z = c$ for some integer c decided by Wowo.
4. Each chocolate is **touched** by at least one cut (i.e. each cuboid has a nonempty intersection with at least one plane).

Decide whether Wowo can cut the cake under the rules. If the answer is yes, output any possible solution.

Input

The first line contains a single integer n ($1 \le n \le 100000$).

The i-th line of the next n lines contains 6 integers $min\_x[i], max\_x[i], min\_y[i], max\_y[i], min\_z[i], max\_z[i]$ ($-10^9 \le min\_x[i], max\_x[i], min\_y[i], max\_y[i], min\_z[i], max\_z[i] \le 10^9$, $min\_x[i] < max\_x[i]$, $min\_y[i] < max\_y[i]$, $min\_z[i] < max\_z[i]$).

Output

If Wowo can cut the cake under the rules, the first line of the output should contain "YES" and the second line should contain 3 integers a, b and c ($-10^9 \le a, b, c \le 10^9$). If Wowo cannot cut the cake under the rules, output only one line containing "NO".

Examples

standard input	standard output
3 0 1 0 1 0 1 10 11 10 11 10 11 999999999 1000000000 999999999 1000000000 999999999 1000000000	YES 0 10 999999999
4 0 1 0 1 0 1 999999999 1000000000 0 1 0 1 0 1 999999999 1000000000 0 1 0 1 0 1 999999999 1000000000	YES 0 0 0

Problem K. Traveling Merchant

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Mr. Lawrence is a traveling merchant who travels between cities and resells products. Basically, to earn from it, he needs to buy products at a very low price and sell them at a higher price. Your task is to tell him whether there exists an endless traveling path that can earn money all the time.

To make things simple, suppose there are n cities named from 0 to $n-1$ and m undirected roads each of which connecting two cities. Mr. Lawrence can travel between cities along the roads. Initially he is located at city 0 and each of the city i has a starting price c_i, either Low or High. Due to the law of markets, the price status at city i will change (i.e. High price will become Low price, or vice versa) after he departs for a neighboring city j from i. (City j is a neighboring city of city i when one of the m roads connects city i and city j.) For some reasons (e.g. product freshness, traveling fee, tax), he **must**:

1. Start at city 0 and buy products at city 0. It is guaranteed that c_0 is Low.
2. When he arrives some city, he either sells products or buys products. It is not allowed for him to do nothing before he leaves the city.
3. After buying products at some city i, he must travel to some neighboring city j whose price c_j is High and sell the products at city j.
4. After selling products at some city i, he must travel to some neighboring city j whose price c_j is Low and buy the products at city j.

As a result, the path will look like an alternation between "buy at low price" and "sell at high price".

An endless earning path is defined as a path consisting of an endless sequence of cities $p_0, p_1, ...$ where city p_i and city p_{i+1} has a road, $p_0 = 0$, and the price alternates, in other words $c_{p_{2k}} =$ Low (indicates a buy-in) and $c_{p_{2k+1}} =$ High (indicates a sell-out) for $k \geq 0$. Please note here c_{p_i} is the price when **arriving** city p_i and this value may be different when he arrives the second time.

Your task is to determine whether there exists any such path.

Input

There are several test cases. The first line contains a positive integer T indicating the number of test cases. Each test case begins with two positive integers n and m indicating the number of cities and the number of roads.

The next line is a string c of length n containing 'H' or 'L'. The i-th ($0 \leq i < n$) charactor of c is H if the starting price c_i at city i is High. The i-th ($0 \leq i < n$) charactor of c is L if the starting price c_i at city i is Low.

The i-th line ($1 \leq i \leq m$) of the following m lines contains two different cities u_i and v_i, indicating a road between u_i and v_i.

The sum of the values of n over all test cases is no more than $200,000$. The sum of the values of m over all test cases is no more than $200,000$. For each test case, $c_i \in \{\mathrm{H}, \mathrm{L}\}$ holds for each $i \in \{0, \ldots, n-1\}$. c_0 is always L. $0 \leq u_i, v_i < n$ and $u_i \neq v_i$ hold for each $i \in \{1, \ldots, m\}$. No two roads connect the same pair of cities.

Output

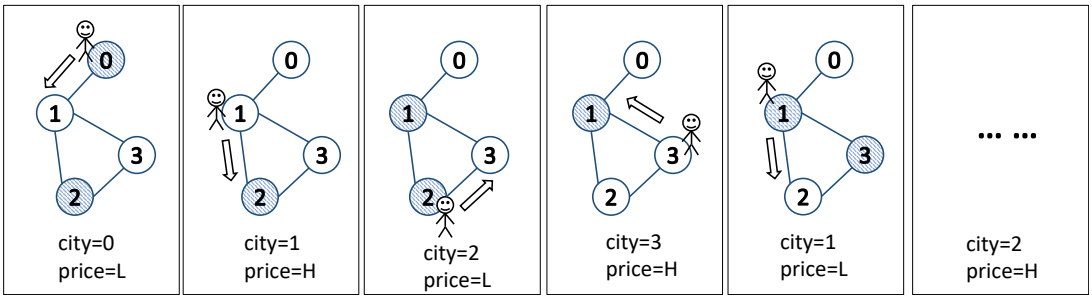
For each test case, output a line of "yes" or "no", indicating whether there exists an endless earning path.

Example

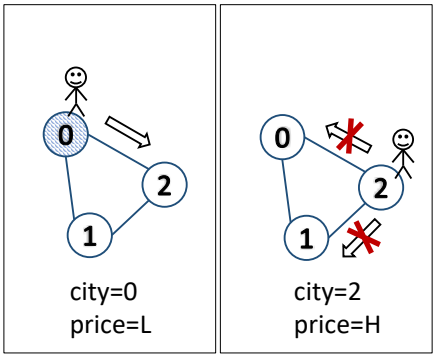
standard input	standard output
2 4 4 LHLH 0 1 1 2 1 3 2 3 3 3 LHH 0 1 0 2 1 2	yes no

Note

In the first sample test case, the endless earning path is $0 \to 1 \to 2 \to 3 \to 1 \to 2 \to 3 \to \ldots$. In the illustration, cities with Low price are filled with stripe.



In the second sample test case, Mr. Lawrence can only make one move from city 0 and after that all cities will have High price. Thus, no further moves can be made.



Problem L. Traveling in the Grid World

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Consider a grid pattern with n rows and m columns. There are $(n+1) \times (m+1)$ grid points in total which is the intersections of $n+1$ horizontal lines and $m+1$ vertical lines. We number the horizontal lines from 0 to n from top to bottom. We number the vertical lines from 0 to m from left to right. The intersection of horizontal line i and vertical line j is named (i, j) $(0 \le i \le n, 0 \le j \le m)$.

There are some constraints when you travel in the grid world. When you are located at point (x, y), you can choose a destination (x', y') and walk to it along the line segment between (x, y) and (x', y'). We call this operation a *walk*. A walk is forbidden if there exists another grid point different from (x, y) and (x', y') lying on the line segment between them. You can walk as many times as you want but the directions of two consecutive walks cannot be the same. (Specifically, if you walk from (x_0, y_0) to (x_1, y_1) and then walk from (x_1, y_1) to (x_2, y_2), you must make sure that $(x_0 - x_1)(y_1 - y_2) \neq (x_1 - x_2)(y_0 - y_1)$.) The length of a walk from (x, y) to (x', y') is defined as the Euclidean distance between the two endpoints, $\sqrt{(x - x')^2 + (y' - y)^2}$.

Starting from $(0, 0)$, you are planning to arrive at (n, m) by several walks. Because of the annoying rules, you may need some turning points to achieve your goal. Please find the minimum total length of your walks.

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains two integers n, m $(1 \le n, m \le 10^6)$ indicating the size of the grid graph.

It is guaranteed that the sum of the values of $\max(n, m)$ over all test cases does not exceed 10^6.

Output

For each test case, output the minimum total length of walks. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-9}.

Example

standard input	standard output
2 2 2 2 3	3.236067977499790 3.605551275463989

Problem M. Gitignore

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Your git project (you don't need to be familiar with git to solve this problem) has some files that should be ignored from synchronizing. You need to calculate the minimum number of lines needed for gitignore.

Formally, your project is a folder. A folder can have files and sub folders. There are no empty folders (i.e. folders without any files or sub folders inside). Initially, the git software will synchronize all the files in your project. However, you can specify some files and folders in the settings (which is called gitignore) to exclude them from synchronizing. For each line in gitignore, you can specify either a file or all the files in a folder. You can **not** ignore the whole project folder (i.e. an empty line in gitignore).

You are given paths for all the files in the project and whether they should be ignored or shouldn't. Your task is to calculate the minimum number of lines for gitignore.

Input

The input contains several test cases. The first line contains a single positive integer T which is the number of test cases. For each test case, you are first given two non-negative numbers n and m. And then n non-empty lines of file paths that should be **ignored**, and m non-empty lines of file paths that should **not** be ignored.

The paths are strings containing lower-cased English alphabets and slashes ('/') only. Slashes are used to separate folders, sub folders and file name. For exapmle, "a/b/c/d" indicates folder "a" in the project folder, folder "b" in folder "a", folder "c" in "b" and file "d" in folder "c". All the paths are valid, specifically:

- The path is non-empty and it always indicates a file (i.e. the path does not end with a slash).
- The path does not start with a slash.
- Folder names and file names are non-empty (i.e. there are no consecutive slashes).
- File paths are always unique (i.e. all the paths in a test case are different).
- In a folder, no sub folders and files share the same names. For example, there won't be two files "a/b/a" and "a/b/a/d" in one test case. However, files "a/b/a" and "a/b/b" are allowed.

$1 \leq n + m \leq 100$ holds and in the whole input there are no more than $1,000$ characters in file paths (i.e. the sum of lengths of file path strings in the whole input file is no more than $1,000$).

Output

T lines of non-negative integers, the minimum number of gitignore lines for each test case.

Example

standard input	standard output
2 3 0 data/train data/test model 3 1 data/train data/test model data/sample	2 3

Note

In the first sample test case, the corresponding gitignore file contains 2 lines: a folder line "data/" and a file name "model".

In the second sample test case, the corresponding gitignore file contains 3 file lines: "data/train", "data/test" and "model".