Problem A. A Hero Named Magnus

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Dota 2 is a multiplayer online battle arena (MOBA) video game developed and published by Valve. Dota 2 is played in matches between two teams of five players, with each team occupying and defending their own separate base on the map. Each of the ten players independently controls a powerful character, known as a 'hero', who all have unique abilities and differing styles of play. During a match players collect experience points and items for their heroes to successfully defeat the opposing team's heroes in player versus player combat. A team wins by being the first to destroy the other team's 'Ancient', a large structure located within their base.

The International is an annual esports world championship tournament for the video game Dota 2, hosted and produced by the game's developer, Valve. The tournament consists of 18 teams; 12 based on final results from the Dota Pro Circuit and six more from winning regional playoffs from North America, South America, Southeast Asia, China, Eastern Europe, and Western Europe regions.

In Year 3021, The International is held in Guilin, China. Once again, just like 1000 years ago, Team LGD from China will compete against Team Spirit from Russia. But as the championship developing, the rule is that whoever wins the best of n (n is an odd positive integer) games will win the champion, so a team should win at least $\frac{n+1}{2}$ games. (In 2021, n equals to only 5 and Team Spirit won by $3:2$).

Before the game starts, teams can choose to ban specific heroes from being used by the opponent team. Among these 1000 years, everyone knows that Team Spirit is very good at using a hero called Magnus, which once helped them defeat Team LGD in 2021.

Although everyone thinks Team LGD will choose to ban Magnus from the beginning, team LGD thinks differently. Somehow they think that they are strong enough to beat the opponent's Magnus and they will only start to ban Magnus in the x-th game if there is one.

To simplify the problem, if team LGD choose to ban Magnus, they will certainly win the game. Otherwise, they may have a 50% possibility to win the game.

As one of Team LGD's fans, JB wants to know what's the minimum number of n that team LGD can win the champion in the worst case.

Input

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

In the next following T lines, each line contains an integer x ($1 \le x \le 2 \times 10^9$), indicating that Team LGD will start to ban Magnus in the x-th game.

Output

For each test case, please output an integer in one line, indicating the minimum total number of games to let Team LGD win.

Example

standard input	standard output
2 1 3	1 5

Note

Ignoring everyone's strongest wish, there exists $x > 1$ in the test data, which means Team LGD won't always choose to ban Magnus from the beginning.

Problem B. A Plus B Problem

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

JB gets a machine that can solve "A Plus B Problem" and feels curious about the mechanism. He hears that you are proficient in competitive programming and have learned many advanced data structures and algorithms such as Link-Cut tree, Lagrange Inversion formula, Sweepline Mo, and so on. Hence, he asks you to help implement a program that can solve "A Plus B Problem" as same as the machine.

The machine consists of $3 \times n$ digits. The digits of the first two rows can be changed arbitrarily, and the third row always equals the decimal sum of the first two rows. The third row only consists of the lowest n digits even if the sum exceeds n digits.

For example, when $n = 5$, the three rows can be "01234", "56789", "58023" or "56789", "58023", "14812".

To test your function, you are given q queries. In the i-th query, the c_i-th digit of the r_i-th row is updated to d_i (the digit may not change). Because the digits are too many and JB has no time to check your answer, he only asks you to find the c_i-th digit of the third row after the query and how many digits of the machine change in the query.

Input

The first line contains two integers n and q ($1 \le n, q \le 10^6$).

The second line contains a string consisting of n digits, representing the first row of the machine.

The third line contains a string consisting of n digits, representing the second row of the machine.

There are q lines in the following. The i-th of the following line consists of three integers r_i, c_i and d_i ($1 \le r_i \le 2$, $1 \le c_i \le n$, $0 \le d_i \le 9$).

Output

Output q lines. In the i-th line, output two integers - the c_i-th digit of the third row after the query and how many digits of the machine change in the query.

Example

standard input	standard output
5 5 01234 56789 2 1 0 2 2 1 2 3 2 2 4 3 2 5 4	0 2 3 2 5 3 7 3 8 3

Note

In the example, the initial rows are "01234", "56789", "58023".

After the 1-st query, the rows are "01234", "06789", "08023".

After the 2-nd query, the rows are "01234", "01789", "03023".

After the 3-th query, the rows are "01234", "01289", "02523".

After the 4-th query, the rows are "01234", "01239", "02473".

After the 5-th query, the rows are "`01234`", "`01234`", "`02468`".

Problem D. Assumption is All You Need

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

JB holds the belief that assumption is all you need to solve a problem. In order to prove that, JB has given you two permutations of numbers from 1 to n: A and B, and JB wants you to output a sequence of element swapping operation (x_i, y_i) on A, so that:

1. every pair of swapped element forms an inversion pair (i.e. $x_i < y_i$ and $A_{x_i} > A_{y_i}$ when the i-th operation is being performed)
2. A will become B at the end of the swapping sequence.

or determine it is impossible. Help prove JB's belief by solving this problem!

Input

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

The first line contains one integer n ($1 \le n \le 2\,021$), indicating the number elements in A and B.

The second line contains n distinct integers $A_1, A_2, \ldots, A_n$ ($1 \le A_i \le n$), indicating the array A.

The third line contains n distinct integers $B_1, B_2, \ldots, B_n$ ($1 \le B_i \le n$), indicating the array B.

It is guaranteed that the sum of n in all test cases will not exceed $2\,021$.

Output

For each test case, if there doesn't exist a sequence, output the one line containing one integer "`-1`".

Otherwise, in the first line output one integer k ($0 \le k \le \frac{n(n-1)}{2}$), indicating the length of the swapping sequence. Then, output k line each containing two integers x_i and y_i ($1 \le x_i < y_i \le n$), indicating the i-th operation swap(A_{x_i}, A_{y_i}).

Example

standard input	standard output
3 2 1 2 2 1 4 4 1 2 3 1 3 2 4 8 8 7 6 5 4 3 2 1 1 8 7 6 5 4 3 2	-1 2 1 2 2 4 7 7 8 6 7 5 6 4 5 3 4 2 3 1 2

Problem E. Buy and Delete

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Alice and Bob are playing a game on a directed graph G. There are n vertices in G, labeled by $1, 2, \ldots, n$. Initially, there are no edges in G. Alice will first buy some direct edges from the shop and then add them into G. After that, Bob needs to delete edges until there are no edges in G. In a deletion round, Bob can delete a subset of edges S from G, such that when only keeping edges in S, the graph is acyclic. Note that Alice can buy nothing, and in such a case the number of deletion rounds is 0.

There are m edges in the shop. Alice has c dollars, so the total price of edges she will buy should not exceed c. Alice wants to maximize the number of deletion rounds while Bob wants to minimize it. Both Alice and Bob will play optimally. Please write a program to predict the number of deletion rounds.

Input

The input contains only a single case.

The first line of the input contains three integers n, m and c ($2 \leq n \leq 2\,000$, $1 \leq m \leq 5\,000$, $1 \leq c \leq 10^9$), denoting the number of vertices in G, the number of edges in the shop, and how many dollars Alice has.

In the next m lines, the i-th line ($1 \leq i \leq m$) contains three integers u_i, v_i and p_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$, $1 \leq p_i \leq 100\,000$), denoting a directed edge in the shop. Alice can pay p_i dollars to buy it, and add an edge from vertex u_i to vertex v_i in G.

Output

Print a single line containing an integer, denoting the number of deletion rounds.

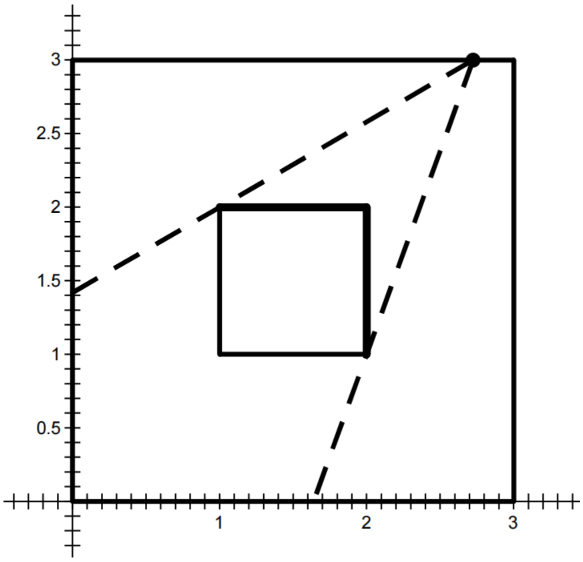
Examples

standard input	standard output
3 2 4 1 2 5 2 3 6	0
3 3 3 1 2 1 2 3 1 1 3 1	1

Problem F. Illuminations II

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

You are given two convex polygons, where the larger polygon has n vertices and the smaller polygon has m vertices. All the vertices of the smaller polygon locate strictly inside the larger polygon, thus the smaller polygon fully locates strictly inside the larger polygon.



Our dear friend JB is going to install an illuminant on the interior boundaries of the larger polygon to light up some exterior boundaries of the smaller polygon. Feeling indecisive, JB decides to choose where to install the illuminant uniformly at random on the interior boundaries of the larger polygon, and you need to calculate the expected length of the illuminated boundaries of the smaller polygon.

Input

The first line contains two integers n and m ($3 \le n, m \le 2 \times 10^5$).

The following n lines describe the vertices on the larger convex polygon, each of which contains two integers x and y ($|x|, |y| \le 10^9$), indicating the coordinates of a vertex on the polygon. All these n vertices are given in counter-clockwise order and any three of them are not collinear.

Then the following m lines describe the vertices on the smaller convex polygon, each of which contains two integers x and y ($|x|, |y| \le 10^9$), indicating the coordinates of a vertex on the polygon. All these m vertices are also given in counter-clockwise order and any three of them are not collinear.

It is guaranteed that all the vertices of the smaller polygon locate strictly inside the larger polygon.

Output

Output the expected length of the illuminated boundaries of the smaller polygon when JB chooses where to install the illuminant uniformly at random on the interior boundaries of the larger polygon.

Your answer is acceptable if its absolute or relative error does not exceed 10^{-9}. Formally speaking, suppose that your output is x and the jury's answer is y, your output is accepted if and only if $\frac{|x-y|}{\max(1,|y|)} \le 10^{-9}$.

Example

standard input	standard output
4 4 0 0 3 0 3 3 0 3 1 1 2 1 2 2 1 2	1.666666666666667

Note

For the sample case, it can be shown that the length of the illuminated boundaries of the smaller polygon is 1 with probability $\frac{1}{3}$ and 2 with probability $\frac{2}{3}$, thus the expected length is $1 \times \frac{1}{3} + 2 \times \frac{2}{3} = \frac{5}{3}$.

Problem G. Occupy the Cities

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

JB is playing a game. There are n cities in the game, numbered as $1, 2, \cdots, n$. The i-th city and the j-th city are adjacent if and only if $i = j - 1$ or $i = j + 1$. Initially, some of the cities are occupied by JB.

The game runs in rounds. At the beginning of a round, each occupied city can mark at most one adjacent unoccupied city as the target of attack. At the end of the round, all the attack targets marked become occupied. The game ends when all the cities are occupied.

JB wants to occupy all the cities in minimum rounds. Can you help him?

Input

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

The first line contains an integer n ($1 \le n \le 10^6$), indicating the number of cities.

The next line contains a string s of length n. It's guaranteed s only contains '0' and '1'. The i-th character describes the initial state of the i-th city: if $s_i =$ '1', the i-th city is occupied by JB initially. Otherwise, the i-th city is not occupied initially.

It's guaranteed that the sum of n over all the test cases doesn't exceed 10^6. It's also guaranteed that there is at least one '1' in s.

Output

For each test case, output one line, containing the minimum number of rounds to occupy all the cities.

Example

standard input	standard output
5 3 010 4 0100 7 0001000 5 11111 6 010101	2 2 4 0 1

Note

For the second test case, the best way is $0100 \to 0110 \to 1111$.

Problem I. PTSD

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

There are n soldiers in JB kingdom, numbered as $1, 2, \cdots, n$. The i-th soldier has a power value of i.

There is a tournament in the kingdom now. The soldiers need to be divided into several groups where each soldier belongs to exactly one group. Note that it's allowed for a group to contain only one single soldier. For some unknown reason, some soldiers have a disease called PTSD (post-traumatic stress disorder). The soldiers with PTSD don't like being the **second** strongest soldier in their groups. Formally speaking, a soldier with PTSD will be upset if there is exactly one other soldier with a larger power value than him in his group.

JB, the king of JB kingdom, wants to maximize the sum of the power values of the soldiers who feel upset because of PTSD. You are asked to help him divide the soldiers.

Input

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

The first line contains an integer n ($1 \le n \le 10^6$), indicating the number of soldiers.

The second line contains a string s of length n. It's guaranteed that s only contains '0' and '1'. The i-th character describes the i-th soldier: If s_i = '1', the i-th soldier has PTSD. Otherwise, the i-th soldier doesn't have PTSD.

It's guaranteed that the sum of n of all test cases doesn't exceed 10^6.

Output

For each test case, output one line containing an integer, indicating the maximum sum of power values of the upset soldiers.

Example

standard input	standard output
4 5 10101 8 11111111 4 1100 4 0110	4 16 3 3

Note

For the first test case, a valid division is [1, 2], [3, 4], [5], which makes the 1-st soldier and the 3-rd soldier upset. [1, 2], [3, 5], [4] is also valid.

For the second test case, a valid division is [1, 2], [3, 4], [5, 6], [7, 8].

For the third test case, a valid division is [1, 3], [2, 4].

For the fourth test case, a valid division is [1, 2, 3, 4].

Problem J. Suffix Automaton

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	512 mebibytes

JB is the National Olympiad Tutor of Suffix Automaton. Today he comes up with the following problem.

Suppose you have a string S, we write down all the distinct substrings in S. Then we sort the strings according to their length in increasing order. For two strings with the same length, the one that has the smaller lexicographical order comes first. Now we have a sorted string sequence A.

JB has Q questions, for each question, he will give you one integer k and suppose you to tell him the k-th string in A.

To simplify the problem, you just need to tell him the left and right positions in S of the first occurrence of the string.

Input

The first line contains one string S ($1 \leq |S| \leq 10^6$), containing only lowercase letters.

The second line contains one integer Q ($1 \leq Q \leq 10^6$).

The following Q lines describe the questions, each of which contains one integer k ($1 \leq k \leq 10^{12}$).

Output

For each question, print two integers l, r, denoting the left and right positions in S of the first occurrence of the answer string. If k is greater than the length of A, just print "`-1 -1`".

Examples

standard input	standard output
ccpcguilin 5 1 10 4 8 26	1 1 2 3 8 8 1 2 4 7
banana 3 5 10 16	1 2 2 5 -1 -1

Problem K. Tax

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

JB received his driver's license recently. To celebrate this fact, JB decides to drive to other cities in Byteland. There are n cities and m bidirectional roads in Byteland, labeled by $1, 2, \ldots, n$. JB is at the 1-st city, and he can only drive on these m roads. It is always possible for JB to reach every city in Byteland.

The length of each road is the same, but they are controlled by different engineering companies. For the i-th edge, it is controlled by the c_i-th company. If it is the k-th time JB drives on an edge controlled by the t-th company, JB needs to pay $k \times w_t$ dollars for tax.

JB is selecting his destination city. Assume the destination is the k-th city, he will drive from city 1 to city k along the shortest path, and minimize the total tax when there are multiple shortest paths. Please write a program to help JB calculate the minimum number of dollars he needs to pay for each possible destination.

Input

The input contains only a single case.

The first line of the input contains two integers n and m ($2 \leq n \leq 50$, $n-1 \leq m \leq \frac{n(n-1)}{2}$), denoting the number of cities and the number of bidirectional roads.

The second line contains m integers $w_1, w_2, \ldots, w_m$ ($1 \leq w_i \leq 10\,000$), denoting the base tax of each company.

In the next m lines, the i-th line ($1 \leq i \leq m$) contains three integers u_i, v_i and c_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$, $1 \leq c_i \leq m$), denoting denoting an bidirectional road between the u_i-th city and the v_i-th city, controlled by the c_i-th company.

It is guaranteed that there are at most one road between a pair of city, and it is always possible for JB to drive to every other city.

Output

Print $n-1$ lines, the k-th ($1 \leq k \leq n-1$) of which containing an integer, denoting the minimum number of dollars JB needs to pay when the destination is the $(k+1)$-th city.

Example

standard input	standard output
5 6 1 8 2 1 3 9 1 2 1 2 3 2 1 4 1 3 4 6 3 5 4 4 5 1	1 9 1 3

Problem M. Pots

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

In a cooking pot shop sales decreased dramatically. Marketing managers of the shop did a research and found out that the reason was frying pans. People stopped buying pots as pans are both cheaper and more compact during storage. The Board of Directors made a decision to extent assortment and start selling also pans. The first batch is already ordered.

Warehouse logistics department was given a task to find a place for the new goods. Now there are N pots in the warehouse. Every pot has a diameter D_i. There is the only way to save space — it is possible to embed into any pot another one of a smaller diameter, into which in turn other can be embedded.

Help the logistics specialist to find a minimal number of pots in the warehouse in which all other pots can be embedded.

Input

The first line contains a single number N ($1 \le N \le 1000$). The second line contains N integers D_i separated by spaces ($1 \le D_i \le 10\,000$).

Output

Output the obtained number.

Examples

standard input	standard output
5 7 5 2 5 2	2

Problem N. The smallest fraction

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Given n fractions. Find *the smallest* positive fraction that, if divided by each of n given fractions, results in an integer number.

Input

The first line contains an integer n ($1 \le n \le 6$). Each of the following n lines contain two integers a_i, b_i that are numerator and denominator of irreducible fraction ($1 \le a_i \le 10^3$, $1 \le b_i \le 10^9$).

Output

Print two positive integers separated by space that are numerator and denominator of the smallest irreducible fraction satisfying the condition of the problem.

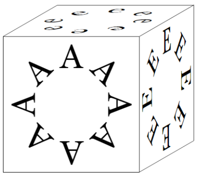
Examples

standard input	standard output
2 1 2 3 4	3 2
2 2 3 4 5	4 1

Problem O. Cubes

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

The slot machine contains N cubes with the English letters written in a special way on each face.



At the beginning of the game, the slot machine puts the cubes on each other so that a player sees only letters on one side of the tower. Then the screen displays the word of N letters, which the player must draw on the visible side of the tower, by turning the cubes around the vertical axis. The word should be read from top to bottom.

Byteazar remembers all cubes produced for those types of slot machines and can distinguish them from each other in the slot machine. However, he cannot know the exact orientation of the cube in the tower as he sees just one cube face. Help Byteazar calculate the probability of that the task set by the slot machine is executable.

Input

The first two lines contain words with a length of N characters – the initial and target words on the visible side of the cube tower ($1 \le N \le 10$). The next N lines contained six characters each describing the letters on the cube faces in the following order: front, top, right, bottom, left and rear ones. Cubes are listed in order from top to bottom. Uppercase and lowercase letters are considered to be different.

Output

Output the probability. The answer is considered to be correct if the absolute or relative error doesnt exceed 10^{-6}.

Examples

standard input	standard output
HALLW HELLO XABCDH XAECDe AbcdeL AbcdeL ABOWCD	0.25

Note

We see the face with the ‘A letter on the second cube, and we need to rotate the cube so that we see the face with the ‘E’ letter. Depending on the rotation of the face with the ‘A’ letter, the face with the ‘E’ letter may be “on the side” or “on the top/bottom” towards Byteazar. In the first case, we can rotate the cube as required, but in the second and third cases — we cannot. The probability of finding ‘E’ “on the side” is equal to the probability of finding it “on the top/bottom” and amounts to 1/2. Therefore, the probability to rotate the cube the right way is 1/2. The ‘W’ and ‘O’ letters are in a similar manner, so the overall probability is 1/4.