Problem A. Automat

Input file:	Standard input
Output file:	Standard output
Time limit:	2 seconds
Memory limit:	256 mebibytes

Byteasar studies computer science at the University of Bytetown. There is a snack vending machine at his faculty that sells n types of snacks, numbered 1 through n. Snacks of different types may have different price, since they differ in size and flavor.

Recently Byteasar discovered that the vending machine is broken. If one buys a snack of type i, the vending machine additionally dispenses one snack of each of the types 1, 2, ..., $i-1$, provided that snacks of these types are available (if there are no snacks of some of the types $1, 2, \ldots, i-1$, simply no snack of this type is dispensed). Buing snack of type i is possible only if at least one snack of this type is available.

Byteasar decided to take advantage of the fault he discovered. He would like to find out what is the maximum total value (that is, the sum of prices) of snacks that he can obtain in the vending machine using a given amount of money. He does not have to use all the money.

Input

The first line of input contains two integers n and k ($1 \le n \le 40$, $1 \le k \le 64\,000$): the number of types of snacks and the amount of money that Byteasar has at his disposal. The second line holds n integers $c_1, \ldots, c_n$ ($1 \le c_i \le 40$), the prices of snacks of respective types. The third line holds n integers $l_1, \ldots, l_n$ ($0 \le l_i \le 40$), the quantities of snacks of respective types that are available in the vending machine.

Output

The only line of output should contain one integer: the total price of snacks that Byteasar can obtain in the vending machine using at most k units of money.

Examples

Standard input	Standard output
6 8 7 2 3 5 7 2 1 3 0 3 2 1	30

Explanation of the example: Byteasar buys a snack of type 6. The vending machine dispenses one snack of each of the types 1, 2, 4, 5 and 6. Next, Byteasar buys a snack of type 4. In addition to this snack, the vending machine dispenses one snack of type 2.

Problem B. Touristic Bureau

Input file:	Standard input
Output file:	Standard output
Time limit:	5 seconds
Memory limit:	256 mebibytes

Byteasar works as a teacher in a primary school in Byteburg. The weather is currently really nice, so Byteasar would like to take his class for a bus trip to Bytetown — the capital of Byteland. Byteasar has decided to hire a travel agency to plan the trip.

The streets in Bytetown form a regular grid of East-West and North-South streets. The distance between any pair of adjacent parallel streets is equal to 1 kilometer. There are tourist attractions located at several junctions. Bytean guides have assigned *attractiveness* coefficient to each tourist attraction: the greater the attractiveness, the more interesting is the attraction for the visitors. Byteasar knows that the pupils in the class he teaches get bored easily. Because of that he requires that the attractions visited on the trip have *increasing* attractiveness.

The travel agency has agreed to Byteasar's requirements. In addition, the agency would like to earn as much money as possible for organizing the trip. The agency gets a fixed rate of 1 bythaler for each kilometer of the trip. The bus always chooses the shortest route along the streets of Bytetown when driving between the attractions on the trip. Moreover the agency gets additional money from the managers of attractions that are visited along the trip.

Help the agency plan a trip that satisfies Byteasar's requirements and grants the agency the highest profit. Please note that driving next to a tourist attraction (without stopping) does not count as visiting the attraction.

Input

The first line of input contains two integers n and m ($2 \le n, m \le 1000$), the number of East-West streets and the number of North-South streets.

n lines follow with a description of the tourist attractions in Bytetown. The i-th of these lines holds m integers $w_{i,j}$ ($0 \le w_{i,j} \le 10^6$) that give the attractiveness of each of the attractions located at the junctions of the i-th East-West street with the respective North-South streets. Attractiveness 0 means that there is no tourist attraction at the respective junction. You can assume that there is at least one tourist attraction in Bytetown.

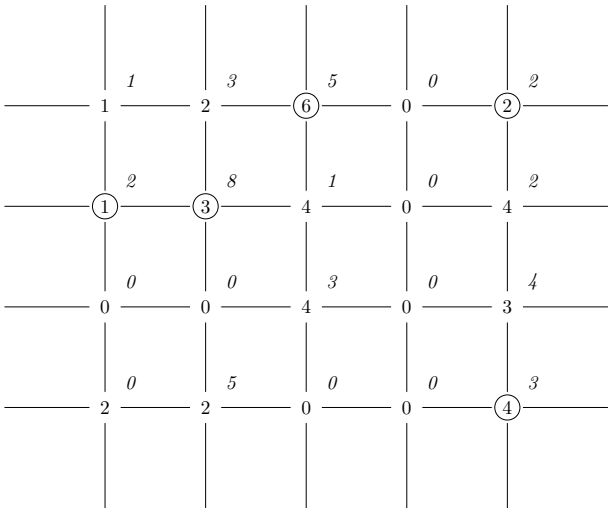
Each of the following n lines holds n integers $c_{i,j}$ ($0 \le c_{i,j} \le 10^9$). The number $c_{i,j}$, that is, the j-th number in the i-th of the considered lines, represents the amount of money (in bythalers) that the agency receives for making a trip lead through the tourist attraction described by the attractiveness $w_{i,j}$. If there is no tourist attraction at a junction, the corresponding number $c_{i,j}$ equals 0.

Output

The only line of output should contain one integer: the maximum profit (in bythalers) that the agency can make for organizing a trip leading through a number of attractions with strictly increasing attractiveness.

Examples

Standard input	Standard output
4 5 1 2 6 0 2 1 3 4 0 4 0 0 4 0 3 2 2 0 0 4 1 3 5 0 2 2 8 1 0 2 0 0 3 0 4 0 5 0 0 3	39



Explanation of the example: The numbers in the junctions represent the attractiveness of the respective attractions. The numbers written in italic represent the profit of the travel agency for making a trip lead through the respective attractions. The most profitable trip for the agency is highlighted with circles: the agency receives 2, 2, 8, 3 and 5 bythalers for these attractions respectively, and moreover it receives 19 bythalers as the bus fare.

Problem C. Sequence

Input file:	Standard input
Output file:	Standard output
Time limit:	2 seconds
Memory limit:	256 mebibytes

We say that an integer sequence $a_1, a_2, \ldots, a_n$ is *k-even* if the sum of any k consecutive terms of the sequnce is even.

For a given sequence we would like to find out how many of its terms need to be changed so that the sequence becomes k-even.

Input

The first line of input contains two integers n and k ($1 \le k \le n \le 1\,000\,000$). The second line contains a sequence composed of n integers $a_1, a_2, \ldots, a_n$. For each of the a_i's it holds that $0 \le a_i \le 1\,000\,000\,000$.

Output

The only line of output should hold one integer: the minimum number of terms of the sequence that need to be changed so that it becomes k-even.

Examples

Standard input	Standard output
8 3 1 2 3 4 5 6 7 8	3
8 3 2 4 2 4 2 4 2 4	0

Problem D. DNA

Input file:	Standard input
Output file:	Standard output
Time limit:	2 seconds
Memory limit:	256 mebibytes

The mad scientist Byteasar would like to give birth to a new kind of creatures. For this, he has decided to modify the genome of a Bytean mouse.

A DNA can be described as a sequence nucleotides denoted by letters `A`, `C`, `G` and `T`. Byteasar's master plan is quite simple: using the DNA of a mouse, he is going to create a new DNA of the same length that is *as little* similar to the mouse's DNA as possible. The similarity of two DNAs is the length of their longest common subsequence. The longest common subsequence of two words x, y is defined as the longest word that can be obtained from each of x, y by removing some (possibly none) letters from both words. (Note that two words may have several longest common subsequences. For example, the longest common subsequences of the words `CACCA` and `CAAC` are: `CAA` and `CAC`.) Write a program that computes the requested DNA.

Input

The first line of input contains one integer n ($1 \le n \le 10\,000$), the length of the DNA of a Bytean mouse. The second line contains a description of nucleotides in the DNA: a sequence of n uppercase letters `A`, `C`, `G`, `T`.

Output

The first line of output should contain one integer: the similarity of the Bytean mouse's DNA and the DNA computed by your program. The second line should hold a sequence of n letters `A`, `C`, `G`, `T`. This should be a DNA that is as little similar to the DNA from the input as possible. Should there be many correct answers, your program may output any one of them.

Examples

Standard input	Standard output
4 GACT	1 TCAG

Problem E. Evaluation

Input file:	Standard input
Output file:	Standard output
Time limit:	15 seconds
Memory limit:	256 mebibytes

Consider an expression E, containing integer constants from 0 to 9, variables from a to z, and the following operations: addition, multiplication and exponentiation with a constant exponent. Quite surprisingly, **each of the variables** $a, b, \ldots, z$ appears in the expression E **at most once**. For a given prime number p, we would like to know how many roots modulo p the polynomial represented by this expression has. In other words, we want to count the number of ways in which integers from 0 to $p-1$ can be assigned to the variables in E, so that the value of E is divisible by p. Since the number of such roots can turn out large, it suffices to output it modulo 30 011.

For example, the polynomial represented by the following expression:

$$E = ((a+y)\cdot(z+8))^2$$

has 15 roots modulo $p = 3$, among which the roots:

$$(a=0,\ y=0,\ z=0), \qquad (a=1,\ y=2,\ z=0), \qquad (a=2,\ y=0,\ z=1)$$

can be found.

More formally, an *expression* is defined as follows:

- Each integer constant `0`, `1`, ..., `9` is an expression.
- Each variable `a`, `b`, ..., `z` is an expression.
- If `A` and `B` are any expressions, then each of `(A+B)` and `(A*B)` is also an expression: the first is the sum of expressions `A` and `B`, and the second is their product.
- If `A` is any expression, and `B` is an integer constant from `2`, `3`, ..., `9`, then `(A^B)` is also an expression: the expression `A` raised to the power of `B`.

Input

The first line of input contains one prime number p ($2 \le p < 15\,000$). The second line contains an expression E as specified above, described by a sequence of at most 300 characters `0`, `1`, ..., `9`, `a`, `b`, ..., `z`, `+`, `*`, `^`, `(`, `)`, without any white space.

Output

Let k denote the number of roots modulo p of the polynomial E. Your program should output one non-negative integer, k mod 30 011.

Examples

Standard input	Standard output
3 (((a+y)*(z+8))^2)	15

Problem F. Formula-1

Input file:	Standard input
Output file:	Standard output
Time limit:	4 seconds
Memory limit:	256 mebibytes

Little Bytie really enjoys watching Formula One races which are held annually on a track between Bytetown and Byteburg. The most exciting moments for him are overtakes. He would like to see as many of them as possible.

Bytie is dreaming of a race in which n Formula One cars compete and the car that started the race at the i-th position (for each $1 \le i \le n$) performs a_i overtakes during the race. We assume for simplicity that at each moment of time at most one overtaking takes place, in which exactly two cars participate (that is, one car goes past another car).

Bytie is wondering whether such a race is possible at all. Could you help him figure this out?

Input

The first line of input contains one positive integer t that represents the number of test cases that follow.

Each test case is described in two lines. The first line contains one integer n ($1 \le n \le 1\,000\,000$): the number of cars that participate in the race. The second line holds a sequence of n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$) that gives the number of overtakes performed by the respective cars.

The size of a single input file does not exceed 20 MB.

Output

Your program should output t lines containing answers to the respective test cases. Each line should hold a single word `TAK` (Polish for *yes*) or `NIE` (Polish for *no*) depending on whether the race described by the test case is possible or not.

Examples

Standard input	Standard output
3 2 0 1 3 0 1 4 3 1 1 3	TAK NIE TAK

Problem G. General

Input file:	Standard input
Output file:	Standard output
Time limit:	2 seconds
Memory limit:	256 mebibytes

Breaking news from Bytetown: the palaeontologists have discovered fossil remains of dinosaurs near the city! After hearing the news, several citizens of Bytetown started willing to pick up a bone or two for themselves. To save the priceless remains of dinosaurs, the mayor of Bytetown has decided to protect the excavation area and hired army for this purpose.

General Byteasar has located n soldiers in several *strategic positions* of the excavation area. We say that a point of the area is *protected* if moving from that point in any direction one unavoidably reduces the distance to at least one of the soldiers.

Byteasar has just been assigned a new rookie soldier. The general has decided to place the soldier in one of the m remaining vacant strategic positions. For each of the possible placements he would like to know what is the total area of the protected part of the excavations.

Input

The first line of input contains two integers n and m ($3 \le n \le 100\,000$, $1 \le m \le 100\,000$): the number of soldiers that are already stationed in the excavation area and the number of vacant strategic positions. The following n lines provide a description of the soldier's positions. The i-th of those lines contains two integers x_i, y_i ($-10^8 \le x_i, y_i \le 10^8$) that represent the coordinates of the position occupied by the i-th soldier (in a rectangular coordinate system). The following m lines provide a description (in the same format) of the vacant strategic positions. All the points listed in the input are distinct.

You may assume that the area of the part of the excavations protected by the n soldiers is positive.

Output

Your program should output exactly m lines. The i-th line should contain the total area of the protected part of the excavations in the case that the rookie soldier is located in the i-th previously vacant strategic position. All numbers should be written with a single digit after the dot.

Examples

Standard input	Standard output
3 2 0 0 2 -1 1 2 3 1 1 0	5.0 2.5

Problem H. Hydra

Input file:	Standard input
Output file:	Standard output
Time limit:	3 seconds
Memory limit:	256 mebibytes

Little Bytie got a gift for his birthday. The gift contained a computer game called *The Amazing Adventures of Knight Byteasar*. The purpose of this game is to lead the knight through numerous challenges to defeat villains and evil witches and rescue damsels in distress.

Bytie managed to complete almost all levels of the game. Now he is stuck at the last level in which Byteasar needs to fight a giant serpent, the Bytean Hydra.

Byteasar will use his sword to fight the monster. Two sword strokes are available in the game. Byteasar can either cut off the serpent's head or slaughter the head (the latter stroke, obviously, requires more effort). Cutting the head off is simpler, however, it results in new heads growing back from the serpent's neck. Hydra is defeated only when it has no more heads and no new heads can grow back from its neck.

The Bytean Hydra may have n types of heads that we number from 1 to n. In the beginning the serpent has one head of type 1. A head of type i (for $1 \le i \le n$) has the following characteristics: the number of sword swipes necessary to cut a head of this type off, u_i, the number of sword swipes necessary to slaughter a head of this type, z_i, and a list of r_i types of heads that grow back in place of a head of this type if it is cut off, $g_{i,1}, \ldots, g_{i,r_i}$.

Help Bytie compute the minimum number of sword swipes that are necessary to defeat the Hydra.

Input

The first line of input contains one integer n ($1 \le n \le 200\,000$), the number of types of heads of the Hydra. The following n lines hold a description of the respective types of heads. The i-th of those lines describes heads of type i. It starts with three integers u_i, z_i, r_i ($1 \le u_i < z_i \le 10^9$, $1 \le r_i$) followed by a list of integers $g_{i,1}, \ldots, g_{i,r_i}$ ($1 \le g_{i,j} \le n$). The sum of all r_i's does not exceed $1\,000\,000$.

Output

The only line of output should contain one integer: the minimum number of sword swipes that are necessary to complete the game.

Examples

Standard input	Standard output
4 4 27 3 2 3 2 3 5 1 2 1 13 2 4 2 5 6 1 2	26

Problem I. Inversions

Input file:	Standard input
Output file:	Standard output
Time limit:	2 seconds
Memory limit:	256 mebibytes

Byteasar discovered a new family of undirected graphs that can be represented using permutations. Let $V = \{1, 2, \ldots, n\}$ be the set of vertices. The description of a graph is given as a permutation $a_1, a_2, \ldots, a_n$ of the set V (that is, a sequence of distinct numbers from V). The vertices a_i and a_j are connected by an edge if the pair (i, j) forms an *inversion* in the permutation, that is, $i < j$ and $a_i > a_j$.

For example, let $n = 4$ and consider the permutation $2, 3, 1, 4$. From this permutation we obtain the following graph:

Byteasar would like to check if the representation that he invented is useful indeed. He has decided to write a program that finds all the *connected components* of the graph. Recall that two vertices $u, v \in V$ belong to the same connected component if there exists a sequence of vertices starting with u and ending with v such that every two subsequent vertices in the sequence are connected by an edge. In our example we have two connected components: $\{1, 2, 3\}$ and $\{4\}$.

Help Byteasar!

Input

The first line of input contains one integer n ($1 \le n \le 1\,000\,000$), the number of vertices of the graph. The second line contains n integers $a_1, a_2, \ldots, a_n$.

Output

The first line of output should contain the number of connected components in the graph. Denote this number by m. Each of the following m lines should hold a description of one connected component. First a number k should be written: the size of the component. Then, an *increasing* sequence of k vertex numbers of the component should be written. The components should be listed in such order that the numbers of the first nodes of the components form an increasing sequence. In other words, if S and S' are two connected components, $u \in S$, $v \in S'$ are their nodes with the smallest number and $u < v$, then the component S should be listed earlier than S'.

Examples

Standard input	Standard output
4 2 3 1 4	2 3 1 2 3 1 4

Problem J. Procrastination

Input file:	Standard input
Output file:	Standard output
Time limit:	2 seconds
Memory limit:	256 mebibytes

procrastination (latin *procrastinatus*, from *pro-* forward, *cras* tomorrow)
is the act of putting off important tasks to a later time

Byteasar tends to postpone all the tasks he needs to perform. However, if he promises to do something, you can certainly count on him.

Byteasar woke up early today and prepared a list of n tasks that he needs to perform in near future. The i-th task will take him d_i consecutive days to perform and has to be completed within the next t_i days, starting from today. Byteasar would like to know how much time he can spend doing nothing until he really has to start performing some tasks. Could you write a program that will help him find that out? Byteasar could also write such a program himself, however this could disturb his procrastination period.

Input

The first line of input contains one integer n ($1 \le n \le 1\,000\,000$): the number of tasks that Byteasar has to perform. The following n lines hold a description of the tasks. The i-th of those lines contains two integers d_i and t_i ($1 \le d_i, t_i \le 10^9$). We assume that Byteasar is able to perform all the tasks on time.

Output

Your program should output one integer k: the maximum number of days during which Byteasar can avoid working. In other words, on the day number $k + 1$ at latest Byteasar must start performing one of the tasks in order to be able to eventually complete all the tasks on time.

Examples

Standard input	Standard output
3 2 8 1 13 3 10	5

Explanation of the example: For the first 5 days Byteasar rests. On the following 5 days he performs the first and the third task (in that order). Afterwards he rests for 1 day and performs the second task, which takes him 2 days.

Problem K. Rabbits

Input file:	Standard input
Output file:	Standard output
Time limit:	2 seconds
Memory limit:	256 mebibytes

Byteasar has decided to grow lettuce in his garden. As you can imagine, Bytean rabbits simply love lettuce. So, not surprisingly, they instantly arrived in Byteasar's garden.

In the garden there are n beds of lettuce numbered 1 through n. Every two subsequent beds are adjacent, that is, for each $i = 1, 2, \ldots, n-1$ the beds number i and $i+1$ are adjacent and, moreover, the bed number n is adjacent to the bed number 1. Right now there are a_i rabbits staying at the bed number i and eating away Byteasar's lettuce.

Byteasar wants to chase out of the garden as many rabbits as possible. For this he is going to use his good old gun. The gun has k bullets inside. Rabbits are extremely timid, so whenever Byteasar shoots towards the bed number i, all the rabbits from that bed leave Byteasar's garden for good. What is more, the rabbits from both adjacent beds are so frightened that they all move to the adjacent bed (obviously, we mean the adjacent bed different from the one towards which was the shot).

Help Byteasar to find the maximum number of rabbits that he can chase out of his garden with at most k shots.

Input

The first line of input contains two integers n and k ($5 \le n \le 2000$, $1 \le k \le n$): the number of beds of lettuce in the garden and the number of bullets Byteasar has in his gun. The second line holds n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 1\,000\,000$): the number of rabbits staying at the subsequent beds.

Output

Your program should output one integer: the maximum number of rabbits that can be chased out of Byteasar's garden using at most k shots.

Examples

Standard input	Standard output
5 2 6 1 5 3 4	13

Explanation of the example: First, Byteasar chases out the 6 rabbits from the bed number 1 (as a result, the rabbits from the bed number 5 move to the bed number 4, whereas the rabbits from the bed number 2 move to the bed number 3). Next, Byteasar chases out the 7 rabbits from the bed number 4.