Problem A. Tree and special vertex

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given a tree with n vertices, numbered from 0 to $n-1$.

It is known, that over time the edges in a tree can collapse.

The vertex with index r is special — from each vertex of the tree there should exist a path to the vertex with index r.

It is assumed that today exactly one of the edges will collapse, but it is unknown which one.

Your task is to add to the graph the minimum number of edges so that when destroying any edge, it still exists a path from each vertex to the vertex with index r.

Input

The first line contains two integers n, r ($2 \le n \le 10^5$, $0 \le r < n$).

The following $n-1$ lines describe the edges of tree in the format u v ($0 \le u, v < n$, $u \ne v$) — the indices of vertices, which are connected by the corresponding edge.

Output

In the first line print one integer k — the minimum number of edges that must be added to the tree to fulfill the required condition.

In the following k lines print two integers u, v ($0 \le u, v < n$, $u \ne v$) — the indices of vertices, which should be connected by the corresponding edge.

If there exist several solutions, you can print any of them.

Examples

standard input	standard output

4 0
0 1
0 2
0 3
	2
	3 2
	3 1

6 0
0 1
0 2
0 3
1 4
1 5
	2
	3 5
	2 4

Problem B. Lottery

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

You have decided to take part in a lottery where there are n chests with gold in a row. You can take away one chest, so you want to choose the chest with the maximum amount of gold.

To find out how much gold each chest contains, you can buy hints from sellers. There are a total of m sellers, i-th of which trades c_i hints. Each hint is described by the numbers l_j, r_j and costs w_j coins. Buying a hint allows you to find out the total amount of gold in chests with numbers from l_j to r_j.

In order to not offend the i-th seller, you must buy **exactly** k_i $(1 \le k_i \le c_i)$ hints from him.

Find the minimum total number of coins that must be paid in order to uniquely determine the amount of gold in each chest.

Input

The first line contains a single integer t $(1 \le t \le 10)$ — the number of tests.

The first line of each test contains two integers n, m $(1 \le n, m \le 80)$ — the number of chests and sellers respectively.

Next lines describe sellers. The first line of the seller's description contains two integers c_i, k_i $(1 \le k_i \le c_i \le 80)$ — the number of hints that the seller trades and the number of hints that you need to buy from him. It is guaranteed that the total number of hints in each test does not exceed 80 $(\sum\limits_{i=1}^{m} c_i \le 80)$.

This is followed by c_i lines describing the hints. In j-th of them there are three integers l_j, r_j, w_j $(1 \le l_j \le r_j \le n,\ 1 \le w_j \le 10^6)$ — parameters and cost of the hint.

Output

For each test, print a single integer — the minimum total number of coins that must be paid in order to uniquely determine the amount of gold in each chest. If it's not possible, print -1.

Example

standard input	standard output

```
2
2 2
1 1
1 2 1
3 2
1 1 10
2 2 100
1 2 1000
2 2
1 1
1 1 1
1 1
1 1 2
```
	```
	111
	-1
	```

Problem C. The set

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2.5 seconds
Memory limit:	256 megabytes

You have an empty multiset of numbers S. Your task is to process n queries. There are two types of queries:

1. `+` x w — add the number x with weight w to S.
2. `-` x w — remove the number x with weight w from S. It is guaranteed that before performing this query, the number x with weight w is present in the multiset S.

After performing each query, you have to find the maximum total weight of such a subset of numbers $A \subset S$ that `xor` of elements of any non-empty subset $B \subset A$ is not equal to zero.

Input

The first line contains a single integer n ($1 \le n \le 300\,000$) — the number of queries.

The following n lines describe queries in the format c x w ($c \in \{$'+', '-'$\}$, $1 \le x, w \le 10^9$).

Output

Print n lines — the maximum total weight of a linearly independent subset of numbers from S after each query.

Example

standard input	standard output
8 + 1 4 + 1 7 - 1 7 + 2 5 + 3 2 + 3 9 - 2 5 - 3 9	4 7 4 9 9 14 13 6

Problem D. Game on matroid

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There is a set X of n elements, i-th of them has it's own weight a_i $(0 \leq i < n)$. On this set matroid[1] $M = \langle X, I \rangle$ has been constructed. Namely, for each set of elements $A \subset X$ you know, whether the set A is independent (it's known whether $A \in I$, or not). It's guaranteed, that **constructed matroid is colorful**[2].

Two players take turns adding to an initially empty set S one element so that the set of selected elements remains independent. The game ends when it is impossible to add an element to the set S so that it remains independent. The first player wants to maximize the total weight of the selected elements, while the second to minimize. Find the total weight of the selected items at the end of the game.

Find results of m games if both players are playing optimally. j-th game differs from the previous one with the weight of element numbered p_j. Note, that constructed matroid M remains unchanged.

Input

First line contains two integers n, m $(1 \leq n \leq 20,\ 1 \leq m \leq 10^5)$ — number of elements and number of games.

The second line contains 2^n symbols c_{mask} $(c_{mask} \in \{0,\ 1\})$, describing matroid. Namely, c_{mask} is equal to 1 iff the corresponding subset of elements is independent $(A_{mask} \in I)$. Elements are numbered starting from zero in bitmask, and bits are written in order from least significant to most significant. For example, bitmask $mask = 5 = 101000\ldots0_2$ describes set of elements $\{X_0,\ X_2\}$.

The third line contains n integers $a_0, a_1, \ldots, a_{n-1}$ $(1 \leq a_i \leq 10^8)$ — the initial weight of each element.

The next m lines describe games. j-th of them contains two integers p_j, x_j $(0 \leq p_j < n,\ 1 \leq x_j \leq 10^8)$ — number of the element and it's new weight.

Output

Print m lines — the total weight of selected elements at the end of each game.

Example

standard input	standard output

3 2
11101110
1 1 5
0 10
1 100
	15
	105

Note

[1]Note, that set I of independent subsets of any matroid meets following axioms:

1. Empty set is independent $(\varnothing \in I)$.
2. Any subset of independent set is independent (if $B \in I$ and $A \subset B$, then $A \in I$).
3. If independent set A has smaller size than independent set B, there exists at least one element in B that can be added into A without loss of independency (if $A, B \in I$ and $|A| < |B|$, then $\exists x \in B \setminus A, A \cup \{x\} \in I$).

[2] Note, that matroid $M = \langle X, I \rangle$ is colorful iff it's possible to color all elements from X in certain colors so that each set A of elements from X is independent iff all the colors of objects from A are different.

Problem E. Hard tasks

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Recently, Oleg found an interesting set of problems. This set contains n problems numbered from 1 to n. Since Oleg likes prime numbers, he decided to solve only those problems which indices are prime.

But Oleg noticed that each problem with a prime index p is quite difficult for him in case when $(p^2 - 1)$ is not divisible by 24. Help Oleg to understand how many difficult problems he will have to solve.

Input

The first line contains a single integer n ($1 \le n \le 0.75 \cdot 10^{10}$) — the number of problems in the Oleg's set.

Output

Print a single integer — the number of prime numbers p from 1 to n such that $(p^2 - 1)$ is not divisible by 24.

Examples

standard input	standard output
2	1
4	2

Problem F. Guess matroid

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

There is a set X of n elements, numbered from 0 to $n-1$. On this set jury thought of matroid[1] $M = \langle X, I \rangle$. You have to guess it. In order to do this, you can ask questions 'whether some subset of elements from X is independent'. You can ask no more than 50000 questions.

Input

First line contains the only integer n $(1 \le n \le 18)$ — number of elements in a set.

Output

When you program is ready to answer, print one line of format '`!` s', where s is a sequence of 2^n symbols, and s_{mask} $(0 \le mask < 2^n)$ is equal to 1 iff the corresponding subset of elements is independent. Elements are numbered starting from zero in bitmask, and bits are written in order from least significant to most significant. For example, bitmask $mask = 5 = 101000\ldots0_2$ describes set of elements $\{X_0, X_2\}$. After giving an answer your program should terminate immediately.

Interaction Protocol

In order to ask a question, print a single line of format '`?` $mask$', where $mask$ $(0 \le mask < 2^n)$ means subset of elements from X. After that your program should read a single integer f $(0 \le f \le 1)$. The corresponding subset of elements is independent iff $f = 1$.

If you ask more than 50000 queries, you will get verdict `Wrong answer`.

Example

standard input	standard output
2 1 0 1 0	 ? 0 ? 1 ? 2 ? 3 ! 1010

Note

[1]Note, that set I of independent subsets of any matroid meets following axioms:

1. Empty set is independent $(\varnothing \in I)$.
2. Any subset of independent set is independent (if $B \in I$ and $A \subset B$, then $A \in I$).
3. If independent set A has smaller size than independent set B, there exists at least one element in B that can be added into A without loss of independency (if $A, B \in I$ and $|A| < |B|$, then $\exists x \in B \setminus A, A \cup \{x\} \in I$).

Problem G. Spanning trees

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	256 megabytes

You are given an undirected graph, consisting of n vertices and m edges. The task is to distribute these edges into k disjoint groups, such that edges in each group represent a tree on n vertices. Note, that **some edges may not belong to any group**.

Input

The first line contains three integers n, m, k ($2 \le n, m \le 200$, $1 \le k \le 10$) — number of vertices and edges in the graph, and quantity of groups for distribution.

Then m lines follow. Each line contains two integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) — the indices of vertices, which are connected by the corresponding edge. Note, that there might be multiple edges and self loops.

Output

If it's impossible to distribute edges into the k groups, print the only line '`No`'.

Otherwise, print '`Yes`' in the first line. Then print k lines, describing all groups. Each line should contain $n-1$ integers $num_1, num_2, \ldots, num_{n-1}$ ($1 \le num_i \le m$) — indices of edges in the corresponding group. Edges are numbered starting from one.

Note, that each index num_i ($1 \le num_i \le m$) may belong to no more than one group, and edges in each group should form a tree on n vertices.

Examples

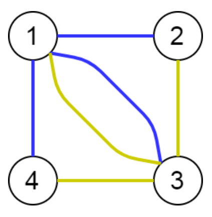
standard input	standard output

4 5 1
1 2
2 3
1 3
2 4
3 4
	Yes
	1 2 4

4 6 2
1 2
2 3
3 4
4 1
1 3
3 1
	Yes
	2 3 6
	1 4 5

Note

The answer for the second test is shown on the picture below. Yellow edges represent first group, while blue — second.



Problem H. Game with one winner

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	256 megabytes

There's an undirected graph with n vertices and m edges. Each edge contains some number of stones. Also, each edge has a certain weight.

Two players play a game. Initially, the first player chooses some subset of edges S_1 from the given graph.

Then the second player chooses some non-empty subset S_2 of edges from S_1. Edges from S_2 should form edge-disjoint edge-simple cycles (in other words, each connected component should have an Euler circuit). If an appropriate subset S_2 does not exist, the second player loses immediately.

If the second player has successfully chosen appropriate subset S_2, players start to play a game of 'Nim' with the piles of stones on the chosen edges from S_2. In a single move, a player may remove an arbitrary positive number of stones from a single pile. The player who can't make a move loses. The first player makes a first move in the 'Nim' game.

Find the maximum total weight of edges from S_1, that first player can choose so that he could always win no matter how the second player acts.

Input

The first line contains two integers n, m ($2 \le n \le 64$, $1 \le m \le 200\,000$) — number of vertices and edges in the graph.

Then m lines follow. Each line contains four integers u_i, v_i, a_i, w_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $0 \le a_i < 2^{60}$, $1 \le w_i \le 10^9$) — the indices of vertices, which are connected by the corresponding edge, number of stones and weight of the edge.

Output

Print the single integer — the maximum total weight of edges from S_1, that first player can choose in order to win.

Examples

standard input	standard output

`3 3`
`1 2 0 1`
`2 3 0 1`
`3 1 0 2`
	`3`

`6 6`
`1 2 1 1`
`2 3 1 2`
`3 4 1 3`
`4 1 1 4`
`5 6 1 2`
`6 5 1 1`
	`11`

`7 1`
`4 7 47 47`
	`47`

Note

In the first sample, the first player can not choose all edges, because in that case the second player also leaves all edges chosen (all 3 edges form Euler-circuit). The following 'Nim' game first player will lose, because he wouldn't be able to make even the first move.

If the first player choose only two last edges with total weight 3, the second player will not be able to choose appropriate subset that form Euler-circuit and will lose immediately.

Problem I. Matroid?

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

There is a set X of n elements, numbered from 0 to $n-1$. For each set of elements $A \subset X$ you know, whether the set A is independent (it's known whether $A \in I$, or not). Determine, whether pair $M = \langle X, I \rangle$ forms a valid matroid.

More formally, you should check, whether set I of independent subsets meets following 3 axioms:

1. Empty set is independent ($\varnothing \in I$).
2. Any subset of independent set is independent (if $B \in I$ and $A \subset B$, then $A \in I$).
3. If independent set A has smaller size than independent set B, there exists at least one element in B that can be added into A without loss of independency (if $A, B \in I$ and $|A| < |B|$, then $\exists x \in B \setminus A, A \cup \{x\} \in I$).

Input

The first line contains the only integer n ($1 \le n \le 22$) — number of elements in set X.

The second line contains 2^n symbols c_{mask} ($c_{mask} \in \{0, 1\}$), describing matroid. Namely, c_{mask} is equal to 1 iff the corresponding subset of elements is independent ($A_{mask} \in I$). Elements are numbered starting from zero in bitmask, and bits are written in order from least significant to most significant. For example, bitmask $mask = 5 = 101000\ldots0_2$ describes set of elements $\{X_0, X_2\}$.

Output

If the input data describes a valid matroid, print the only line '`Yes`'.

Otherwise in the first line print '`No`'.

If the first axiom fails, print '`1`' in the second line.

If the second axiom fails, print '`2 A B`', where A and B — two bitmasks, for corresponding subsets of which the second axiom fails.

If the third axiom fails, print '`3 A B`', where A and B — two bitmasks, for corresponding subsets of which the third axiom fails.

If several axioms fail at the same time, you can print any one of them.

Bitmasks should be printed as a sequence of symbols $f_0, f_1, \ldots, f_{n-1}$ ($f_i \in \{0, 1\}$), that means the presence of the corresponding element in the set.

Examples

standard input	standard output

`3`
`11101110`
	`Yes`

`4`
`0111111111111111`
	`No`
	`1`

`2`
`1001`
	`No`
	`2 01 11`

`3`
`11111000`
	`No`
	`3 001 110`

Problem J. Martoids intersection

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Martoid is a binary matrix of size $n \times m$, in each cell of which is written symbol '`0`' or '`1`'. In order for a matrix to be a valid martoid, it should meet two additional properties:

1. There exists at least one cell with a symbol '`1`' written on it.
2. There exists a path between any pair of cells with symbols '`1`' written on them, such that adjacent cells in the path share a common side, and all cells in the path are marked with '`1`' symbol.

Martoids intersection of two martoids A and B is a binary matrix C of size $n \times m$, in each cell of which is written symbol '`1`' iff two corresponding symbols in that martoids are also equal to '`1`'. Please note, that martoids intersection is not necessary a valid martoid.

You are given a binary matrix C. It's guaranteed that the **outermost elements of it are equal to zero**, i.e. $C_{i,j} = 0$ for $i = 1$, $i = n$, $j = 1$, $j = m$. Find two martoids A and B, such that their intersection is equal to binary matrix C.

Input

The first line contains two integers n, m ($3 \le n, m \le 500$).

Then n lines follow. Each of them contains m symbols $C_{i,1}, C_{i,2}, \ldots, C_{i,m}$ ($C_{i,j} \in \{$'`0`', '`1`'$\}$).

Output

If the desired martoids do not exist, print the only line '`Impossible`'. Otherwise print two martoids A and B, such that their intersection is equal to given matrix.

First n lines should describe martoid A in the same format as the input. The following one line should be empty. Last n lines should describe martoid B in the same format as the input.

If there are several possible answers, you can print any of them.

Examples

standard input	standard output

5 5
00000
01010
00000
01010
00000
	00000
	11111
	10001
	11111
	00000
	
	01110
	01010
	01010
	01010
	01110

3 7
0000000
0110110
0000000
	0111110
	0110110
	0111110
	
	0000000
	1111111
	0000000

Problem K. Unique sums

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	256 megabytes

You are given an array a consisting of n integers. Each element of the array equals to -1, 0, or 1. Your task is to find the number of different values that the expression $\sum\limits_{i=l}^{r} a_i$ can take, where $1 \le l \le r \le n$.

Input

The first line contains a single integer n ($1 \le n \le 10^6$) — the length of the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-1 \le a_i \le 1$).

Output

Print a single integer — the answer to the problem.

Example

standard input	standard output
5 1 -1 1 0 -1	3

Problem L. If you get bored

Input file:	`standard input`
Output file:	`standard output`
Time limit:	15 seconds
Memory limit:	256 megabytes

When Martoid gets bored, he starts playing with his favorite toy — an electronic display. This display shows five digits, so it can show any five-digit integer from 0 to 99999. Initially, the display shows the number a ($0 \le a \le 99999$).

The game with the display is as follows. The display chooses two numbers L, R ($1 \le L < R \le 99999$) and reports them to Martoid. After that, the display thinks of a number x from the interval $[L; R]$. Martoid has to guess it. To do this, he can perform the following operations:

1. change the value of any digit of the display;
2. ask the result of the comparison of the number x and the number y, which is currently displayed: whether the number x is less than, equal to, or greater than the number y.

Your task is to find the minimum number of operations needed for Martoid to make a guess regardless of the thought number x.

Input

The first line contains a single integer t ($1 \le t \le 50$) — the number of tests.

Each of the following t lines contains three integers a, L, R ($0 \le a \le 99999$, $1 \le L < R \le 99999$) — the initial number shown on the display and the interval from which the number x is chosen.

Output

For each test, print a single integer — the minimum number of operations needed for Martoid to make a guess regardless of the thought number x.

Example

standard input	standard output

4
47 46 48
0 97 107
77744 12043 12045
100 61 69
	1
	6
	4
	8

Note

In the first test, you can perform the second type of operation immediately. If the result is 'less than' $x = 46$, else if the result is 'greater than' $x = 48$, otherwise $x = 47$. That is why one operation is enough to guess the number x.