A. String Reconstruction

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Ivan had string s consisting of small English letters. However, his friend Julia decided to make fun of him and hid the string s. Ivan preferred making a new string to finding the old one.

Ivan knows some information about the string s. Namely, he remembers, that string t_i occurs in string s at least k_i times or more, he also remembers exactly k_i positions where the string t_i occurs in string s: these positions are $x_{i, 1}, x_{i, 2}, \ldots, x_{i, k_i}$. He remembers n such strings t_i.

You are to reconstruct **lexicographically minimal** string s such that it fits all the information Ivan remembers. Strings t_i and string s consist of small English letters only.

Input

The first line contains single integer n $(1 \le n \le 10^5)$ — the number of strings Ivan remembers.

The next n lines contain information about the strings. The i-th of these lines contains non-empty string t_i, then positive integer k_i, which equal to the number of times the string t_i occurs in string s, and then k_i distinct positive integers $x_{i, 1}, x_{i, 2}, \ldots, x_{i, k_i}$ in increasing order — positions, in which occurrences of the string t_i in the string s start. It is guaranteed that the sum of lengths of strings t_i doesn't exceed 10^6, $1 \le x_{i, j} \le 10^6$, $1 \le k_i \le 10^6$, and the sum of all k_i doesn't exceed 10^6. The strings t_i can coincide.

It is guaranteed that the input data is not self-contradictory, and thus at least one answer **always** exists.

Output

Print lexicographically minimal string that fits all the information Ivan remembers.

Examples

input

```
3
a 4 1 3 5 7
ab 2 1 5
ca 1 4
```

output

```
abacaba
```

input

```
1
a 1 3
```

output

```
aaa
```

input

```
3
ab 1 1
aba 1 3
ab 2 3 5
```

output

```
ababab
```

B. High Load

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Arkady needs your help again! This time he decided to build his own high-speed Internet exchange point. It should consist of n nodes connected with minimum possible number of wires into one network (a wire directly connects two nodes). Exactly k of the nodes should be exit-nodes, that means that each of them should be connected to exactly one other node of the network, while all other nodes should be connected to at least two nodes in order to increase the system stability.

Arkady wants to make the system as fast as possible, so he wants to minimize the maximum distance between two exit-nodes. The distance between two nodes is the number of wires a package needs to go through between those two nodes.

Help Arkady to find such a way to build the network that the distance between the two most distant exit-nodes is as small as possible.

Input

The first line contains two integers n and k ($3 \le n \le 2 \cdot 10^5$, $2 \le k \le n - 1$) — the total number of nodes and the number of exit-nodes.

Note that it is always possible to build at least one network with n nodes and k exit-nodes within the given constraints.

Output

In the first line print the minimum possible distance between the two most distant exit-nodes. In each of the next $n - 1$ lines print two integers: the ids of the nodes connected by a wire. The description of each wire should be printed exactly once. You can print wires and wires' ends in arbitrary order. The nodes should be numbered from 1 to n. Exit-nodes can have any ids.

If there are multiple answers, print any of them.

Examples

input

```
3 2
```

output

```
2
1 2
2 3
```

input

```
5 3
```

output

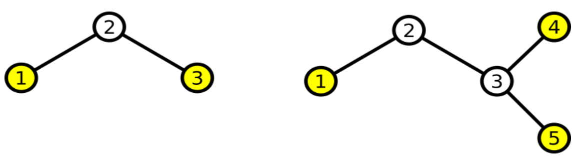
```
3
1 2
2 3
3 4
3 5
```

Note

In the first example the only network is shown on the left picture.

In the second example one of optimal networks is shown on the right picture.

Exit-nodes are highlighted.



C. DNA Evolution

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Everyone knows that DNA strands consist of nucleotides. There are four types of nucleotides: "A", "T", "G", "C". A DNA strand is a sequence of nucleotides. Scientists decided to track evolution of a rare species, which DNA strand was string s initially.

Evolution of the species is described as a sequence of changes in the DNA. Every change is a change of some nucleotide, for example, the following change can happen in DNA strand "AAGC": the second nucleotide can change to "T" so that the resulting DNA strand is "ATGC".

Scientists know that some segments of the DNA strand can be affected by some unknown infections. They can represent an infection as a sequence of nucleotides. Scientists are interested if there are any changes caused by some infections. Thus they sometimes want to know the value of impact of some infection to some segment of the DNA. This value is computed as follows:

- Let the infection be represented as a string e, and let scientists be interested in DNA strand segment starting from position l to position r, inclusive.
- Prefix of the string $eee...$ (i.e. the string that consists of infinitely many repeats of string e) is written under the string s from position l to position r, inclusive.
- The value of impact is the number of positions where letter of string s coincided with the letter written under it.

Being a developer, Innokenty is interested in bioinformatics also, so the scientists asked him for help. Innokenty is busy preparing VK Cup, so he decided to delegate the problem to the competitors. Help the scientists!

Input

The first line contains the string s ($1 \le |s| \le 10^5$) that describes the initial DNA strand. It consists only of capital English letters "A", "T", "G" and "C".

The next line contains single integer q ($1 \le q \le 10^5$) — the number of events.

After that, q lines follow, each describes one event. Each of the lines has one of two formats:

- `1 x c`, where x is an integer ($1 \le x \le |s|$), and c is a letter "A", "T", "G" or "C", which means that there is a change in the DNA: the nucleotide at position x is now c.
- `2 l r e`, where l, r are integers ($1 \le l \le r \le |s|$), and e is a string of letters "A", "T", "G" and "C" ($1 \le |e| \le 10$), which means that scientists are interested in the value of impact of infection e to the segment of DNA strand from position l to position r, inclusive.

Output

For each scientists' query (second type query) print a single integer in a new line — the value of impact of the infection on the DNA.

Examples

input

```
ATGCATGC
4
2 1 8 ATGC
2 2 6 TTT
1 4 T
2 2 6 TA
```

output

```
8
2
4
```

input

```
GAGTTGTTAA
6
2 3 4 TATGGTG
1 1 T
1 6 G
2 5 9 AGTAATA
1 10 G
2 2 6 TTGT
```

output

```
0
3
1
```

Note

Consider the first example. In the first query of second type all characters coincide, so the answer is 8. In the second query we compare string "TTTTT..." and the substring "TGCAT". There are two matches. In the third query, after the DNA change, we compare string "TATAT..."' with substring "TGTAT". There are 4 matches.

D. Best Edge Weight

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a connected weighted graph with n vertices and m edges. The graph doesn't contain loops nor multiple edges. Consider some edge with id i. Let's determine for this edge the maximum integer weight we can give to it so that it is contained in all minimum spanning trees of the graph if we don't change the other weights.

You are to determine this maximum weight described above for each edge. You should calculate the answer for each edge independently, it means there can't be two edges with changed weights at the same time.

Input

The first line contains two integers n and m ($2 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 2 \cdot 10^5$), where n and m are the number of vertices and the number of edges in the graph, respectively.

Each of the next m lines contains three integers u, v and c ($1 \le v, u \le n$, $v \ne u$, $1 \le c \le 10^9$) meaning that there is an edge between vertices u and v with weight c.

Output

Print the answer for each edge in the order the edges are given in the input. If an edge is contained in every minimum spanning tree with any weight, print `-1` as the answer.

Examples

input

```
4 4
1 2 2
2 3 2
3 4 2
4 1 3
```

output

```
2 2 2 1
```

input

```
4 3
1 2 2
2 3 2
3 4 2
```

output

```
-1 -1 -1
```

E. Rusty String

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Grigory loves strings. Recently he found a metal strip on a loft. The strip had length n and consisted of letters "V" and "K". Unfortunately, rust has eaten some of the letters so that it's now impossible to understand which letter was written.

Grigory couldn't understand for a long time what these letters remind him of, so he became interested in the following question: if we put a letter "V" or "K" on each unreadable position, which values can the period of the resulting string be equal to?

A period of a string is such an integer d from 1 to the length of the string that if we put the string shifted by d positions to the right on itself, then all overlapping letters coincide. For example, 3 and 5 are periods of "VKKVK".

Input

There are several (at least one) test cases in the input. The first line contains single integer — the number of test cases.

There is an empty line before each test case. Each test case is described in two lines: the first line contains single integer n $(1 \le n \le 5 \cdot 10^5)$ — the length of the string, the second line contains the string of length n, consisting of letters "V", "K" and characters "?". The latter means the letter on its position is unreadable.

It is guaranteed that the sum of lengths among all test cases doesn't exceed $5 \cdot 10^5$.

For hacks you can only use tests with one test case.

Output

For each test case print two lines. In the first line print the number of possible periods after we replace each unreadable letter with "V" or "K". In the next line print all these values in increasing order.

Example

input

```
3

5
V??VK

6
??????

4
?VK?
```

output

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

Note

In the first test case from example we can obtain, for example, "VKKVK", which has periods 3 and 5.

In the second test case we can obtain "VVVVVV" which has all periods from 1 to 6.

In the third test case string "KVKV" has periods 2 and 4, and string "KVKK" has periods 3 and 4.

F. Dirty Arkady's Kitchen

time limit per test: 6 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Arkady likes to walk around his kitchen. His labyrinthine kitchen consists of several important places connected with passages. Unfortunately it happens that these passages are flooded with milk so that it's impossible to pass through them. Namely, it's possible to pass through each passage in any direction only during some time interval.

The lengths of all passages are equal and Arkady makes through them in one second. For security reasons, Arkady can never stop, also, he can't change direction while going through a passage. In other words, if he starts walking in some passage, he should reach its end and immediately leave the end.

Today Arkady needs to quickly reach important place n from place 1. He plans to exit the place 1 at time moment 0 and reach the place n as early as he can. Please find the minimum time he should spend on his way.

Input

The first line contains two integers n and m ($1 \le n \le 5 \cdot 10^5$, $0 \le m \le 5 \cdot 10^5$) — the number of important places and the number of passages, respectively.

After that, m lines follow, each of them describe one passage. Each line contains four integers a, b, l and r ($1 \le a, b \le n$, $a \ne b$, $0 \le l < r \le 10^9$) — the places the passage connects and the time segment during which it's possible to use this passage.

Output

Print one integer — minimum time Arkady should spend to reach the destination. If he can't reach the place n, print `-1`.

Examples

input

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

output

```
3
```

input

```
2 1
1 2 1 100
```

output

```
-1
```

Note

In the first example Arkady should go through important places $1 \to 3 \to 4 \to 5$.

In the second example Arkady can't start his walk because at time moment 0 it's impossible to use the only passage.