Fall 2020 mBIT Advanced Division

November 14, 2020

These problems are roughly ordered by difficulty. However, you should read and think about as many problems as you can in the time given. Good luck and happy coding!

Contents

Thanks to Evan Chen for letting us use his style file

Program Specifications

The memory limit for every problem is 256 MB, unless otherwise specified.

Time Limits (seconds)			
Problem	**C++**	**Java**	**Python**
Climbing Trees	1	1	1
Stone Piles	1	1	2
Calendars	1	1	1
The Duplicator	1	1	1
Locked in the Past	1	1	2
Night of the Candles	1	2	2
Gemstones	2	2	6
The Flock of Rams	2	4	4
Textile Display	1	1	3
Tanya's Revenge	1	3	3
Sphinx Economics	1	1	2
Building Atlantis	2	2	3

Advice

Look at the pretests. You can access all 10 pretests for each problem once you've made a submission. *Some of the pretests are reduced in size to help you debug your program.* Keep in mind that your final submission will be judged on a separate set of 40 hidden system tests for the official rankings.

Take advantage of subtasks. Many problems have subtasks which allow you to earn points for solving easier cases. Each subtask guarantees that a certain proportion of the system tests satisfies some additional constraint. Remember, you will get one point for each system test you pass, **plus 20 points** if you get all 40 system tests correct.

Watch out for integer overflow. When a problem uses large values, make sure you use `long` (in Java) or `long long` (in C++). Python integers cannot overflow.

Consider using fast I/O. For problems with large input sizes, you may want to use faster I/O methods to prevent a time limit error. Here is how to use fast I/O in each language:

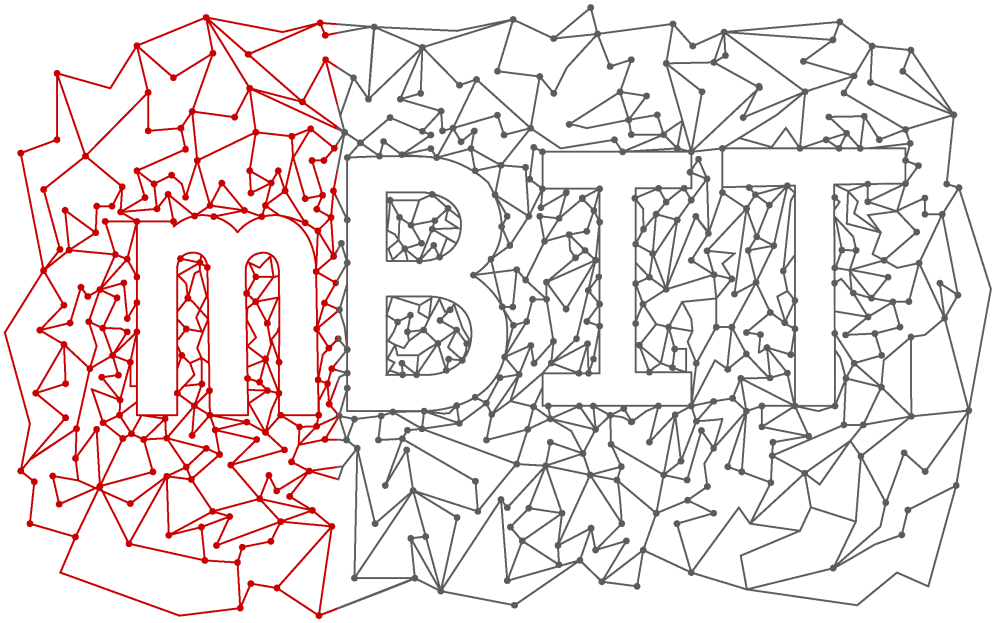
- In Python, write `from sys import stdin, stdout` at the top of your program. When reading input, use `stdin.readline()`. To write output, use `stdout.write()`.
- In Java, use a custom Scanner class as shown here.
- In C++, write `ios_base::sync_with_stdio(false); cin.tie(NULL);` at the top of your `main` method. Then you can use `cin` and `cout` as usual. Printing a single newline character (`\n`) is faster than `endl`, which flushes the output stream (`endl` is only needed in interactive problems.)

Print extra digits for non-integer values in C++. If you are printing a `double` value in C++, by default it will only output a few digits (which may result in a wrong answer from our grader). To output real values with more precision, write `cout << setprecision(16);` at the top of your program. Our grader will accept real values in

fixed **or** scientific notation (so `1234.56789`, `1.23456789E3`, and `1.23456789e+003` are treated the same). There will always be a tolerance for small relative errors between your solution and the correct answer.

Special considerations for Python. In Python, avoid multidimensional lists. Nesting lists may slow your program down. You can use a trick known as array flattening instead. Additionally, if you are coding in Python, consider submitting your solution in PyPy. It behaves in the same way as Python, but is often faster (the time limits for PyPy are the same as Python.) Putting your code in a `main` method can speed up run time as well. Finally, **do not** use `exit()`.

Ask for clarifications! If you are confused about a problem statement, do not hesitate to message us.



"Time is the longest distance between two places."
—Tennessee Williams

"We all have our time machines. Some take us back, they're called memories. Some take us forward, they're called dreams."
—Jeremy Irons

"Time limit exceeded on pretest 3."
—The grader

§1 Climbing Trees

Scientists believe that the last common ancestor of humans and modern chimpanzees lived around 10 million years ago.

Joe the monkey wants to get some exercise by climbing trees in the forest. The forest has N trees with heights $A_1, \ldots, A_N$ meters. Joe is rather unfit, so he can only climb a tree if its height is within K meters (inclusive) of the height of the previous tree he climbed. He also refuses to visit any tree more than once. More formally, Joe will climb a sequence of trees $T_1, \ldots, T_t$ ($1 \le T_i \le N$) such that $|A_{T_{i+1}} - A_{T_i}| \le K$ for $i = 1, \ldots, t-1$ and $T_i \neq T_j$ for $i \neq j$. What is the maximum sum $\sum_{i=1}^{t} A_{T_i}$ of the heights of the trees that Joe climbs? He may start by climbing any tree.

Input Format:

The first line contains N and K ($1 \le N \le 10^5$; $1 \le K \le 10^9$), the number of trees in the forest and the permitted height difference between consecutive trees that Joe climbs.

The next line contains N integers $A_1, \ldots, A_N$ ($1 \le A_i \le 10^9$) representing each tree's height.

Output Format:

Print a single integer representing the maximum total height that Joe can climb.

Sample Input:

```
5 5
12 18 2 7 4
```

Sample Output:

```
25
```

Joe can start by climbing the 4-meter tree, then the 2-meter tree, then the 7-meter tree, and finally the 12-meter tree for a total height of 25 meters. This is the optimal total height. Note that there is more than one order in which Joe can climb these trees.

§2 Stone Piles

The stone circle in Nabta Playa (a desert site in southern Egypt) actually predates the Stonehenge by at least two millennia.

Gabe is building a replica of the Stonehenge, but he must first organize the stones properly. There are currently N stones arranged into M piles. In each pile, the stones are stacked on top of each other so that there is always one stone on top (unless the pile is empty, in which case there is no top stone). Each stone has a certain *type*, described by an integer between 1 and M, inclusive. In one action, Gabe can move the top stone from any non-empty pile to the top of another pile. He wants to find a sequence of actions that will result in all stones ending up in their proper stack. That is, pile t must contain all the stones of type t for $t = 1, 2, \ldots, M$. Can you help him find such a sequence? **Your solution does not have to be the shortest possible, but it cannot include more than $5 \cdot 10^5$ actions.**

Input Format:

The first line contains two integers N and M ($1 \leq N \leq 10^5; 3 \leq M \leq 10^5$) representing the number of stones and the number of piles.

The next M lines each contain an integer R_i representing the number of stones in the i-th pile, followed by R_i integers $T_{i,1}, \ldots, T_{i,R_i}$ describing the types of the stones in the pile from **top to bottom** ($0 \leq R_i \leq N$; $1 \leq T_{i,j} \leq M$). It is guaranteed that there will be N stones in total among all the stacks.

Output Format:

On the first line print K, the number of actions in your solution ($0 \leq K \leq 5 \cdot 10^5$).

The next K lines should each contain two integers a_i and b_i ($1 \leq a_i, b_i \leq M$), meaning that as his i-th action Gabe should move the stone on the top of pile a_i to the top of pile b_i.

It can be proven that a valid solution always exists under these conditions.

Subtasks:

1. $N, M \leq 1000$ (25% of system tests)
2. No additional constraints (75% of the system tests)

Sample Input:

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

Sample Output:

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

The initial configuration of the piles is as follows (an underscore indicates an empty pile):

```
1
2 3 _
```

The following sequence of 4 actions places each stone into its correct pile as desired:

- $2 \longrightarrow 3$

```
1
2 _ 3
```

- $1 \longrightarrow 3$

```
    1
2 _ 3
```

- $1 \longrightarrow 2$

```
    1
_ 2 3
```

- $3 \longrightarrow 1$

```
1 2 3
```

Although this solution can be shown to be the shortest possible, any valid sequence with less than or equal to $5 \cdot 10^5$ actions will be accepted.

§3 Calendars

The Aztecs and the Mayans both used a 365-day solar calendar cycle and a 260-day ritual calendar.

Clarence the historian has become interested in the calendars of ancient civilizations. During his research, he found two mysterious calendars A and B belonging to early cultures. Both calendars have N-day weeks, but the days may appear in different orders in the two calendars. Formally, each calendar can be represented as a permutation of the numbers $1, \ldots, N$.

He has defined some vocabulary to describe calendars for his research. Given any permutation P, let $pos_P(k)$ be the index of k in P (indexing starts at 1). For example, if $P = [2, 4, 1, 5, 3]$ then $pos_P(4) = 2$ because 4 is the second element of P. Clarence then defines the *permutational distance* between any two permutations P and Q (both of size N) to be

$$dist(P, Q) := \sum_{k=1}^{N} |pos_P(k) - pos_Q(k)|.$$

Clarence wants to know how similar his two mysterious calendars are. Unfortunately, the second calendar B is written on a circular tablet, so he doesn't where it starts and ends. Thus, Clarence wants to know the smallest permutational distance that any rotation of B could have with A. That is, he wants to calculate the minimum value of $dist(A, C)$ for any permutation C that is a rotation of B (meaning C can be created from B by successively moving B's first element to the back). Can you help him find this value? Note that A is fixed; Clarence only wants to rotate B.

Input Format:

The first line contains the integer N ($1 \leq N \leq 10^5$).

The second line contains N distinct integers $A_1, \ldots, A_N$ ($1 \leq A_i \leq N$) describing the first calendar cycle.

The third line contains N distinct integers $B_1, \ldots, B_N$ ($1 \leq B_i \leq N$) describing the second calendar cycle.

Output Format:

Print the minimum permutational distance between A and any rotation of B.

Subtasks:

1. $N \leq 1000$ (25% of system tests)
2. No additional constraints (75% of system tests)

Sample Input:

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

Sample Output:

```
2
```

The permutation `1 2 4 3` is the best choice for C. The permutational distance between `1 4 2 3` and `1 2 4 3` is 2, which is the minimum over all rotations of B.

Note that a permutation is considered a 0-rotation of itself, so C is allowed to be equal to B.

§4 The Duplicator

Dolly the sheep was the first mammal cloned from an adult somatic cell, using the process of nuclear transfer.

In the year 2050, Halvin and Cobbes have realized their childhood dreams with their newest invention: the Duplicator! Halvin has used the Duplicator to make a clone of himself named Wholevin. Although Halvin and Wholevin certainly look the same, Cobbes wants to measure how effective the Duplicator was at cloning Halvin on the genetic level.

Cobbes uses hair samples from the look-alikes to analyze their DNA. The genomes of Halvin and Wholevin can each be represented as an ordered list of N integers. Cobbes labels Halvin's sequence $A_1, \ldots, A_N$ and Wholevin's sequence $B_1, \ldots, B_N$. The similarity of the two genomes is measured by the number of pairs of integers i, j satisfying $1 \leq i < j \leq N$ with the property that $A_i \neq A_j$ and $B_i \neq B_j$. Can you help Cobbes determine the number of such pairs?

Input Format:

The first line contains N ($1 \leq N \leq 10^5$).

The next line contains N integers $A_1, \ldots, A_N$ ($1 \leq A_i \leq N$).

The final line contains N integers $B_1, \ldots, B_N$ ($1 \leq B_i \leq N$).

Output Format:

Print one line containing the number of pairs i, j with $1 \leq i < j \leq N$ such that $A_i \neq A_j$ and $B_i \neq B_j$.

Sample Input:

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

Sample Output:

```
4
```

The ordered pairs $(i, j) = (1, 4), (1, 5), (2, 3)$, and $(2, 4)$ are the only ones that satisfy the conditions, thus the answer is 4.

§5 Locked in the Past

The asteroid that caused the dinosaurs to go extinct was only 15 kilometers wide, but it produced a crater 150 kilometers in diameter.

Jo has traveled back in time to see the dinosaurs! After narrowly escaping a hungry T-rex, he realizes that the giant asteroid is about hit Earth. Jo needs to get back to the present as fast as possible, but unfortunately the door to his time machine is fastened with a combination lock (to keep out nosy velociraptors). The lock is made of N wheels in a row numbered $1, \ldots, N$. Each wheel displays an integer from 0 to K, inclusive (think of a bicycle lock like this). In a single second, Jo can increase or decrease all of the numbers in a **contiguous** interval of wheels by 1. Decreasing a wheel at 0 turns it to K, and increasing a wheel at K turns it to 0.

Fortunately, Jo's password is very simple: it's just N zeroes in a row. The wheels of the combination lock are currently set to the values $A_1, \ldots, A_N$ when reading from left to right. What is the shortest amount of time Jo needs to change all of the values to zero? The clock is ticking!

Input Format:

The first line contains N and K ($1 \leq N \leq 1000$; $1 \leq K \leq 10^9$), the length of the password and the highest value that can be shown on the wheels.

The next line contains N integers $A_1, \ldots, A_N$ ($0 \leq A_i \leq K$) representing the current wheel values.

Output Format:

Print one line with the minimum amount of time it takes for Jo to enter his password of N zeroes.

Subtasks:

1. $N \leq 300$ (50% of system tests)
2. No additional constraints (50% of system tests)

Sample Input:

```
5 10
1 6 3 2 7
```

Sample Output:

```
10
```

Jo performs the following moves:

- Decrease wheels 1 through 4
- Decrease wheels 2 through 4
- Decrease wheels 2 through 3

- Three times in a row, decrease wheel 2
- Four times in a row, increase wheel 5

After these 10 moves, all of the wheels will be 0.

§6 Night of the Candles

Since the 4th century, the Macedonians have had a tradition of lighting candles after a death to protect the deceased's soul from ghosts and demons.

The memory limit for this problem is 512 MB.

An ancient Macedonian graveyard is filled with $N \times M$ candles neatly arranged in a grid with N rows and M columns, all evenly spaced. K of the candles are initially lit. On a windy night there are B breezes occurring one after another, each blowing in one of the four cardinal directions: north, south, east, or west.

When a breeze blows, a flame on a lit candle will spread to the adjacent candle in that direction, and set it on fire if it is currently unlit. Each breeze lasts just long enough for every flame to spread to an adjacent candle, but no further. In other words, **a newly lit candle does not continue to spread during the breeze in which it was lit.** Each breeze affects the whole grid at once, and flames cannot spread outside of the grid. Once a candle is lit, it will never stop burning. Can you report how many candles in the grid are lit after each breeze?

If a candle is in the r-th row from the top and the c-th column from the left (denoted as (r, c)), then a breeze can spread its flame as follows:

- North: move from (r, c) to $(r-1, c)$.
- South: move from (r, c) to $(r+1, c)$.
- East: move from (r, c) to $(r, c+1)$.
- West: move from (r, c) to $(r, c-1)$.

Input Format:

The first line contains four integers N, M, K, and B ($1 \leq N, M \leq 1000; 1 \leq K \leq \min(N \cdot M, 10^5); 1 \leq B \leq 10^5$) denoting the number of rows, columns, initially lit candles, and breezes.

The next line contains one string of length B denoting the breezes in the order of their occurrences. Each character will be one of $\{\texttt{N}, \texttt{S}, \texttt{E}, \texttt{W}\}$.

Each of the final K lines contains of a pair of integers r_i and c_i ($1 \leq r_i \leq N; 1 \leq c_i \leq M$) denoting the row and column of an initially lit candle in the grid.

Output Format:

Print B lines containing the number of lit candles after each breeze.

Sample Input:

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

Sample Output:

```
4
8
12
```

The state of the grid initially looks as follows (0 is an unlit candle, 1 is lit):

```
0 1 0 1
0 0 0 0
0 0 0 0
```

The first breeze causes flames to spread south, so there are now 4 lit candles:

```
0 1 0 1
0 1 0 1
0 0 0 0
```

The second breeze causes flames to spread west, so there are now 8 lit candles:

```
1 1 1 1
1 1 1 1
0 0 0 0
```

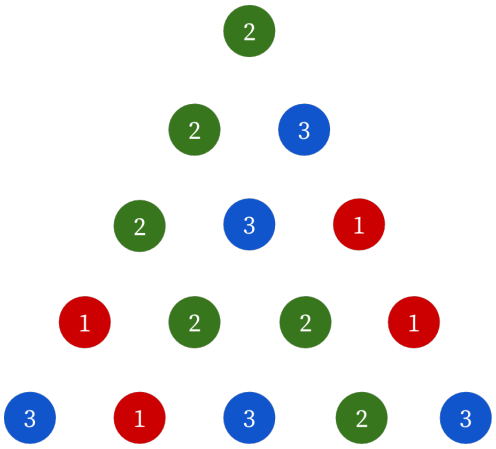
The final breeze causes flames to spread south, so there are now 12 lit candles:

```
1 1 1 1
1 1 1 1
1 1 1 1
```

§7 Gemstones

Under the Delhi Sultanate, the Gujarati city of Cambay exported vast quantities of luxury products, including polished gemstones, pearls, and silk cloth.

Maxwell hears of the wonderful business opportunities in Cambay and resolves to pursue a career as a gemstone merchant. Owing to his idiosyncrasies, he has opened a gem mine in the shape of an equilateral triangle with N gems on each side. Each gem comes in one of three colors. An example array with $N = 5$ is shown below:



Maxwell wants to use pieces of string to connect the gems into bracelets, but he has some important criteria (again owing to his idiosyncrasies):

- Each string must be a straight line segment passing through **at least two gems**.
- Each string must start on a gem and end on a gem.
- Each string must be parallel to one side of the triangular mine.
- No two strings may intersect, including at endpoints (no gem may touch more than one string).
- For variety, each string must have a **different** length.
- The total number of strings must be maximized.

After Maxwell is done, he turns each piece of string into a bracelet containing all of the gems it connects. Each bracelet is assigned a price equal to *the number of times the most common color of the bracelet occurs*. In other words, if a string connects r red gems, b blue gems, and g green gems, the corresponding bracelet has a price of $\max(r, g, b)$. Find the maximum sum of bracelet prices Maxwell can obtain under these conditions.

Input Format:

The first line contains the integer N ($1 \leq N \leq 400$).

Among the next N lines, the i-th line contains i integers $a_{i,1}, \ldots, a_{i,i}$ ($1 \leq a_{i,j} \leq 3$) specifying (from left to right) the color of the gems in the i-th row from the top.

Output Format:

Print the maximum total price of the bracelets that Maxwell can obtain.

Subtasks:

1. $N \leq 80$ (25% of system tests)
2. $N \leq 200$ (25% of system tests)
3. No additional constraints (50% of system tests)

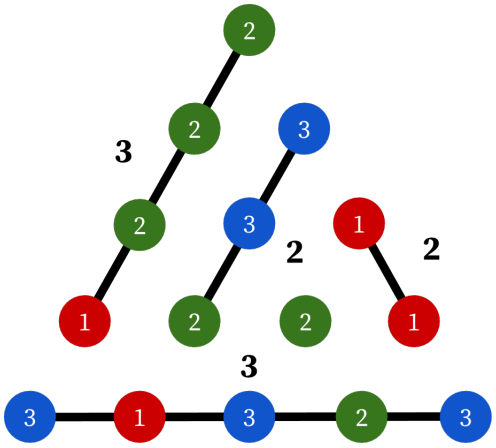
Sample Input:

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

Sample Output:

```
10
```

This array is the same as that in the problem description. It can be shown that the maximum total price is 10, which can be achieved when the gems are strung as shown:



The number next to each string is the price of the bracelet created from that string. It can be shown that this configuration maximizes the total number of strings, which is one of Maxwell's requirements.

§8 The Flock of Rams

In the Odyssey, *Odysseus escapes Polyphemus by hanging on to the undersides of his sheep as they are let out to graze.*

The cyclops Polyphemus has a flock of K rams. After suffering a humiliating defeat from Odysseus, Polyphemus has decided to guard his flock inside a fortress! His fortress is represented by an $N \times M$ grid of cells with `#`'s and `.`'s, where each `#` character represents an obstacle and each `.` character represents open space. All K rams are located at distinct points inside the fortress.

The hero Barusu wants to destroy some of the obstacles to rescue **at least two of the rams**. A ram is considered rescued if it can walk out of the fortress by itself. Rams can only move in the four cardinal directions, and they cannot move through obstacles. Formally, Barusu wants there to be empty paths from at least two of the rams to the edge of the fortress (these two paths may intersect). What is the minimum number of obstacles Barusu needs to destroy to complete his mission? Rams will not be located on obstacles.

Input Format:

The first line contains three integers N, M, and K ($1 \leq N \cdot M \leq 10^5; 1 \leq K \leq \min(N \cdot M, 100)$) denoting the number of rows, columns, and rams.

The next N lines contain M characters each, describing the layout of the fortress. Each character will either be a period (`.`) or a pound (`#`).

Each of the final K lines contains two numbers x_i and y_i, meaning there is a ram located in the x_i-th row and y_i-th column of the fortress (counting from the top left). It is guaranteed that the locations of the rams are distinct, and no ram will be located on an obstacle.

Output Format:

Print one line with the minimum number of obstacles Barusu has to destroy to free at least two rams.

Sample Input:

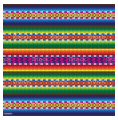
```
11 7 3
#######
#######
#.####.
######.
#######
#####.#
#####.#
##..##.
##.#.##
####.##
..#####
3 2
8 3
9 5
```

Sample Output:

```
2
```

If Barusu destroys the obstacles at locations $(8, 5)$ and $(8, 6)$, the second ram and the third ram will be able to escape the fortress.

§9 Textile Display



The Incans often used alpaca, llama, and vicuña wool to weave clothing and textiles.

The memory limit for this problem is 512 MB.

Huitaca wants to show off her expert weaving skills to the village. She has prepared N textiles for her local fair. Each textile can be one of M colors, where two textiles of the same color are considered **distinguishable**. On the day of the fair she will put all of her textiles on a display table. Then, N villagers will come up one by one to inspect her work.

When a villager comes to the table, he or she will first take a moment to admire the variety of the textiles. The *impression factor* of a villager is the number of distinct colors of textiles they see when visiting Huitaca's table. Once the villager has determined their impression factor, Huitaca will choose a textile and give it to the villager to take home (Huitaca is a generous weaver). This will happen for all N villagers, so by the end of the day there will be no more textiles left. Huitaca's *happiness* is be the sum of the impression factors of all N villagers.

Huitaca wants to compute the sum of her happiness across all $N!$ orders in which she can hand out the textiles. Can you help her find this number? Compute the answer $(\text{mod } 10^9 + 7)$.

Input Format:

The first line contains N and M $(1 \leq N \leq 5 \cdot 10^6; 1 \leq M \leq 10^5)$.

The next line contains M positive integers $C_1, \ldots, C_M$, where C_i is the number of textiles of color i. It is guaranteed that $\sum_{i=1}^{M} C_i = N$.

Output Format:

Print one line with the sum of Huitaca's happiness across all $N!$ possible ways she could give out her textiles, taken $(\text{mod } 10^9 + 7)$.

Subtasks:

1. $N \leq 10^5$ (50% of system tests)
2. No additional constraints (50% of system tests)

Sample Input:

```
6 3
1 2 3
```

Sample Output:

```
9660
```

§10 Tanya's Revenge

General Tanya is a recurring mBIT character, first appearing in a problem from the 2019 contest called Secret Base.

The memory limit for this problem is 512 MB.

The year is 2200. The planet is enveloped in a great war between humans and AI. General Tanya oversees her country with N villages numbered $1, \ldots, N$. These villages are connected by $N-1$ roads, such that there exists a path between any two villages along these roads.

In case of an emergency attack, having undirected roads could lead to chaos with people running around in a panic. Thus, Tanya decides to direct all of the roads so that any given road can only be traveled along in one direction. Several of the villages are designated *battle forts*, and village 1 is designated Tanya's headquarters. The headquarters **may or may not be a battle fort**. Tanya calls the path between any pair of villages u and v ($u \neq v$) a *battle path* if you can get from u to v by following the one-way roads, and either:

- all roads on the path point away from the headquarters, or
- all roads on the path point towards the headquarters.

After directing each of the $N-1$ roads, Tanya calls the *battle-readiness* the number of battle paths that **end on a battle fort**. Can you help Tanya determine the maximum battle-readiness she can achieve by directing the roads?

Input Format:

The first line contains N ($1 \leq N \leq 5000$).

The next line contains a binary string of length N composed of 1's and 0's. The i-th village is a battle fort if and only if the i-th character in this string is a 1.

The next $N-1$ lines each contain two integers u_i and v_i ($1 \leq u_i, v_i \leq N$), where villages u_i and v_i are initially connected by an undirected road.

Output Format:

Print one line with the maximum battle-readiness Tanya can achieve by directing the roads.

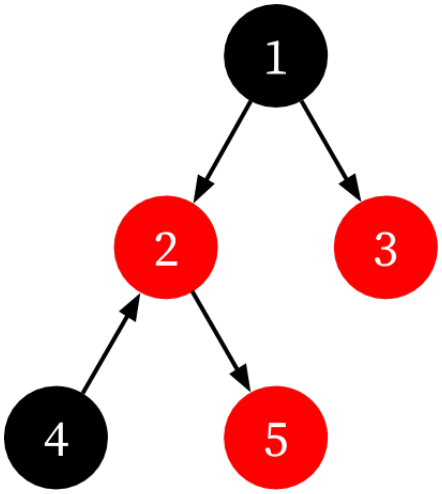
Sample Input:

```
5
01101
1 2
1 3
2 4
2 5
```

Sample Output:

```
5
```

Tanya can achieve 5 battle paths that end at a battle fort ($1 \to 2; 1 \to 3; 2 \to 5; 4 \to 2; 1 \to 2 \to 5$) by directing the roads like this (battle forts in red):



Note that $4 \to 2 \to 5$ is *not* a battle path because the road from 4 to 2 points towards the headquarters while the road from 2 to 5 points away from the headquarters.

§11 Sphinx Economics

The Sphinx is a mythical creature of legendary power with the head of a human, the body of a lion, and the wings of an eagle.

Faris has been whisked all the way back to ancient Egypt, where she encounters a magical Sphinx! The Sphinx has agreed to play a betting game her. Faris starts with 1 dollar. Faris must then answer a sequence of N riddles given to her by the Sphinx. Fortunately for Faris, **the Sphinx only asks yes-or-no questions**, so there are only two possible answers to each riddle.

Before giving her response to each riddle, Faris is allowed to place an x-dollar bet, where x can be any non-negative value she chooses that is less than or equal to the amount of money she currently has (x does **not** have to be an integer). She then gains or loses x dollars depending on whether or not she answers correctly. Note that she immediately learns the correct answer to each riddle after making her guess.

The Sphinx's questions are impossibly difficult, so Faris doesn't know how to solve any of the riddles by themselves. However, she does have a secret advantage: she knows that the Sphinx will never ask Q consecutive riddles with the same answer. For example, if $Q = 3$ and the Sphinx has just asked two questions where the correct answer has been *YES*, the next answer **must** be *NO*. If she uses this fact to her benefit, what is the maximum amount of money that Faris can **guarantee** having after all N questions? In other words, if Faris acts optimally to maximize her profits in the worst case, what is the least amount of money she could end up with? The answer can be represented as $\frac{p}{q}$, where p and q are positive relatively prime integers. Find $p \cdot q^{-1} \pmod{10^9 + 7}$, where q^{-1} denotes the modulo inverse of q with respect to $10^9 + 7$.

Input Format:

The only line contains N and Q ($2 \leq Q \leq N \leq 10^6$).

Output Format:

Print one line with the amount of money guaranteed if Faris plays optimally, taken $\pmod{10^9 + 7}$ in the format described above.

Subtasks:

1. $N \leq 1000$ (50% of system tests)
2. No additional constraints (50% of system tests)

Sample Input:

```
3 3
```

Sample Output:

```
333333337
```

It can be shown that the maximum amount of money Faris can guarantee is $\frac{4}{3}$ dollars, so the answer is $4 \cdot 3^{-1} \equiv 333333337 \pmod{10^9 + 7}$.

§12 Building Atlantis

People have had many theories about the location of Atlantis, including in the Mediterranean, off the coast of Spain, and even under what is now Antarctica.

Aria has travelled back in time to visit the mythical city of Atlantis. When she finds out that Atlantis never existed in the first place, Aria decides to build the city herself! She plans to construct N giant pillars lined up in a row to mark the city's entrance. Each pillar starts out with height 0. To build up these pillars, Aria has programmed M robots numbered $1, \ldots, M$. Robot i is assigned an interval $[l_i, r_i]$ and a production value d_i.

The robots take turns depositing sandstone to increase the height of the pillars, going in the order $1, \ldots, M$. When it is robot i's turn to build the pillars, it finds the shortest pillar in its interval $[l_i, r_i]$ and increases its height by d_i inches (if there is a tie, it chooses the lower-indexed pillar). Once all robots have performed their task, robot 1 takes another turn and the cycle continues over and over again.

Let L_t and H_t be the heights of the shortest and tallest pillars, respectively, after all robots have completed t moves. Aria wonders what will happen if she leaves the robots to work for thousands of years. More specifically, she wants to know the limit of $\frac{L_t}{H_t}$ as $t \to \infty$. Can you help her find this ratio? Your output will be accepted if the absolute error is less than 10^{-6}.

Input Format:

The first line contains N and M ($1 \leq N, M \leq 25000$).

Among the next M lines, the i-th line contains l_i, r_i, and d_i ($1 \leq l_i \leq r_i \leq N; 1 \leq d_i \leq 10^9$), the properties of the i-th robot.

Output Format:

Print one line with the ratio of the shortest pillar height to the tallest pillar height as t approaches infinity.

Sample Input:

```
4 2
1 2 4
2 4 20
```

Sample Output:

```
0.6
```

After both robots have taken 4 moves, the pillars will have heights $16, 40, 20, 20$. After both robots have taken 100 moves, the pillars will have heights $400, 680, 660, 660$. It can be shown that as t increases, $\frac{L_t}{H_t}$ will approach $\frac{3}{5} = 0.6$.