IDI Open Programming Contest March 16th, 2024

Problem Set

A Climbing Capacity
B Deceitful Dice (Easy)
C Exponential Eggs (Easy)
D Friend Finding
E Möbius Music
F Puzzling Poison Protection
G Restaurant Rinser
H Sausage Slicing
I Terminal Tools

Jury and Problem Writers

Jean Niklas L'orange (Head Judge)
Tobias Meyer Andersen

Test Solvers

Johan Sokrates Wind

Rules

TL;DR: Teams of up to three persons try to solve as many problems as possible from a set, without external help.

Before the contest begins, you are allowed to log in on your assigned computer, and log in on the submission system. You may do nothing else with the computer (such as starting to write code). You may not touch the problem set before the contest has started.

Contestants are *only* allowed to communicate with members of their own team, and the organisers of the contest. You may not surf the web (except for allowed content), read e-mail, chat on Slack, or similar things. The only network traffic you may generate is from submitting problem solutions, and access to content specified by the local organisers.

- What you may bring to the contest floor:
  - Any written material (Books, manuals, handwritten notes, printed notes, etc).
  - Pens, pencils, blank paper, stapler and other useful non-electronic office equipment.
  - NO material in electronic form (CDs, USB pen and so on).
  - NO electronic devices (Cellular phone, PDA and so on).
- What you may use during the contest:
  - What you brought to the contest floor (see above).
  - Your assigned (single) computer.
  - The specified system for submitting solutions: `https://idio24.kattis.com`
  - Printers designated by the organiser.
  - The external documentation for your language of choice. This is documented in the tutorial for your language at `https://support.kattis.com`
  - All compilers and IDEs pre-installed on your assigned computer
  - Non-programmable tools which are a natural part of the working environment (such as `diff` and `less`).

The problem set consists of a number of problems (usually 8-12). The problem set will be in English, and given to the participating teams when the contest begins. For each of these problems, you are to write a program supported by the Kattis system. The jury guarantees that each problem is solvable in C, C++, Java and Python 3. No guarantees for other languages are given due to the large number of allowed languages, however the jury guarantees that for every language there is at least one problem solvable in that language. It has always been the case that several of the problems were solvable in all available languages, but there is no guarantee of this.

Your programs should read from standard in and write to standard out. See `https://support.kattis.com` for a tutorial and the compiler options for your language of choice.

The team that solves the most problems correctly wins. If two teams solve the same number of problems, the one with the lowest total time wins. If two top teams end up with the same number of problems solved and the same total time, then the team with the lowest time on a single problem is ranked higher. If two teams solve the same number of problems, with the same total time, and the same time on all problems, it is a draw. The time for a given problem is the time from the beginning of the contest to the time when the first correct solution was submitted, plus 20 minutes for each incorrect submission of that problem. The total time is the sum of the times for all solved problems, meaning you will not get extra time for a problem you never submit a correct solution to.

If you think some problem is ambiguous or underspecified, you may ask the judges for a clarification request through the Kattis system. The most likely response is "No comment, read problem statement", indicating that the answer can be deduced by carefully reading the problem statement or by checking the sample test cases given in the problem, or that the answer to the question is simply irrelevant to solving the problem.

Input Validation

In general we are lenient with small formatting errors in the output, in particular whitespace errors within reason. But not printing any spaces at all (e.g. missing the space in the string "`1 2`" so that it becomes "`12`") is typically not accepted. The safest way to get accepted is to follow the output format exactly.

For problems with floating point output, we only require that your output is correct up to some error tolerance. For example, if the problem requires the output to be within either absolute or relative error of 10^{-4}, this means that

- If the correct answer is 0.05, any answer between 0.0499 and .0501 will be accepted.
- If the correct answer is 500, any answer between 499.95 and 500.05 will be accepted.

Any reasonable format for floating point numbers is acceptable. For instance, "`17.000000`", "`0.17e2`", and "`17`" are all acceptable ways of formatting the number 17. For the definition of reasonable, please use your common sense.

Tips

- Tear the problem set apart and share the problems among you.
- Problems are **not** sorted by difficulty.
- Try solving the easy problems first. Two problems in this set are tagged with "(Easy)" to help point you in the right direction.
- If your solution fails on a problem, you can print your program and debug it on paper while you let someone else work on a different problem on the computer.
- All problems are guaranteed to be solvable in C, C++, Java and Python 3
- Look at the scoreboard if you are unsure which problem to work on next.

Climbing Capacity

Problem ID: climbingcapacity

Agnar loves ascending mountains. He has planned a route that consists of N legs that must be completed in order. He starts each day fully rested, but each leg makes him more tired that day. Agnar tolerates a certain level of tiredness from all the legs he completes on a certain day. The interesting thing is that the level of tiredness he experiences in a day is the product of how tiresome every individual leg completed that day is! Given M, the number of days Agnar has to complete the hike, figure out the highest level of tiredness Agnar will be if he tries to minimize how tired he is on the most tiresome day, if he strategically chooses how many consecutive legs to complete each day.

Input

The first line contains the number N, which describes the number of legs the complete hike consists of, and the number M, which is the number of days Agnar can use to complete the hike. The next line contains the N integers T_1 to T_N, which is the level of tiredness associated with each leg.

Output

The tiredness level from the most tiresome day can grow very large, therefore it is sufficient to output the decimal number E, which is the number such that 10^E is equal to the tiredness level. Your answer must have an absolute or relative error of at most 10^{-6}.

Limits

- $1 \leq N \leq 100\,000$
- $1 \leq M \leq N$
- $1 \leq T_i \leq 100$

Sample Input 1	Sample Output 1
4 2 2 3 2 2	0.77815125

Deceitful Dice
Problem ID: deceitfuldice

Hallvard the high-roller has produced a set of dice that will be used in his new, and exceptionally unfair casino. The set of dice has K faces, and Hallvard knows the probability of each face facing upwards. To make sense of what odds to give the gamblers in his casino, Hallvard must organize the dice according to their expected value.

Given a set of N K-sided dice, print out the dices ordered by which dice rolls the highest value on average.

Input

The first line contains two integers, N, and K, which describes the number of dice, and the number of faces each dice has. Then follow N lines of data, where the ith line describes the ith dice. Each line consists of the K integers C_1 to C_K. C_j is the percentage of dice rolls that will have the value of j. You may assume that all of the percentages for each dice add up to 100.

Output

N numbers, where the first number is the dice that produces the highest value on average, and the second one produces the second highest value on average and so on. If two dice produce the same average, their order should be the same as in the input.

Limits

- $1 < N, K \leq 100$
- $0 \leq C_i \leq 100$

Sample Input 1

```
3 4
48 48 2 2
2 48 48 2
2 2 48 48
```

Sample Output 1

```
3 2 1
```

Sample Input 2

```
3 6
15 15 15 15 15 25
25 25 0 0 25 25
20 20 10 10 20 20
```

Sample Output 2

```
1 2 3
```

Exponential Eggs

Problem ID: exponentialeggs

In Elmspring, they celebrate New Year's Eve with a very local tradition: Eating an egg. Every member of the village receives an egg, and only when the clock strikes 12, you are allowed to eat it.

Of course, using normal chicken eggs would be too easy, so they have opted to give everyone an egg from the legendary roc. Because Elmspring needs so many eggs from such a rare bird, they have decided to tame and breed rocs to ensure every villager will get an egg on New Year's Eve.

The mayor Eldra Willowbrook wants to ensure they have a healthy surplus, and has decided that they will hatch every egg they don't use during the festivities. For example, if they get 100 eggs and only need 77, then they will hatch the remaining 23 eggs into rocs.

Now, the roc is not a legendary bird just because it is big: It is also immortal, and every female bird will lay an egg just before New Year's Eve. Of course, not every hatched roc will be female, only about f of all hatched eggs will be.

Eldra wants to estimate the number of rocs they will have after N years. Since there's uncertainty involved, she's fine with an approximation, and has decided to use the formula

$$R_n = \lfloor (1+f)R_{n-1} \rfloor - V$$

where R_n is the number of estimated rocs in year n, V is the number of villagers, and $\lfloor n \rfloor$ is the floor of n (n rounded down to the closest integer).

Unfortunately, she doesn't have a computer available, so she cannot easily compute this for herself. Therefore she has turned to you, as you are the only villager in Elmspring with a computer.

Input

The input consists of one line with three integers N, R_0 and V, and one real number f.

N is the number of years to compute for, R_0 is the current population of rocs, V is the number of villagers, and f is the approximate proportion of rocs that are female.

Output

Assuming the number of villagers is constant for the next N years, output R_N.

Limits

- $0 < N \leq 20$
- $20 \leq R_0 \leq 1000$
- $0 < V < \lfloor R_0 \times f \rfloor$
- $0.00 < f < 1.00$
- All real numbers will have exactly two digits after the decimal point.

Sample Explanation

In the first sample, R_1 is computed as follows:

$$R_1 = \lfloor (1+f)R_0 \rfloor - V = \lfloor (1+0.23)30 \rfloor - 5 = 31$$

With R_1, we can compute R_2:

$$R_2 = \lfloor (1+f)R_1 \rfloor - V = \lfloor (1+0.23)31 \rfloor - 5 = 33$$

Sample Input 1	Sample Output 1
2 30 5 0.23	33

Sample Input 2	Sample Output 2
10 250 100 0.50	3068

Friend Finding

Problem ID: friendfinding

Dilawar, Haavard and Kjerand are best friends, and they are moving to the same neighborhood. The neighborhood consists of N intersections, and M roads connecting them. It takes the same amount of time to walk a road in both directions. The friends want to choose the position of their new homes according to a very strict criteria. They want to live in a way that minimizes the time it takes for all three to meet at an intersection. Their new homes must be placed on top of an intersection, and they will live at three distinct intersections.

Input

The first line contains N and M, the number of intersecitons, and roads respectively. Then follows M lines with the values u_i, v_i, and t_i, which describe the starting position of the i'th road, the intersection at which it ends, and the time it takes to travel from one end to the other.

Output

Output the number T, which states how long time it has to take for them too meet at an intersection, given that they place their new homes optimally.

Limits

- $3 \leq N, M \leq 100000$
- $1 \leq t_i \leq 10000$
- $0 \leq u, v \leq N - 1$

Sample Input 1	Sample Output 1
4 4 0 1 2 0 2 3 0 3 4 2 3 3	3

Möbius Music

Problem ID: mobiusmusic

Erlend the entertainer has just composed another piece that will impress the world. The piece takes the idea of cyclic form to the next level, the song is constructed to be a perfect loop, where the end of the song beautifully connects to the beginning.

To make a number out of this on the next concert, Erlend has written his notes on a Möbius strip. A Möbius strip is a geometric shape with only one face, meaning that the piece when written on the strip is only a single line of notes, but its end and start connect. This was thought to be a brilliant idea, but just before the concert Erlend realised his mistake, he cannot tell where the piece begins! Luckily you remember all the notes and can use that to help him find where it begins. Time is running out, so tell him where the first note of the song is, relative to where Erlend is currently looking on the strip.

Input

The first line contains N, the number of notes in the entire piece. Then follows a line with the music piece from start to finish, it is represented as N integers m_1 to m_N, where each integer represents a specific note. Then follows a second line, which has the same format as the music piece on the previous line, except it is from the perspective of Erlend. This means that the first integer on the line describes the note he is currently looking at.

Output

To help Erlend quickly prepare for the concert, you should print the smallest number of notes he has to move his sheet either to the left or the right to correctly align the sheet with the start of the piece. Output "`left`" or "`right`" followed by the number of notes Erlend has to move his gaze to be looking at the first note of the piece. Tiebreaks favor looking at a note to the right. If he is already looking at the right place, print "`you are ready!`".

Limits

- $1 \leq N \leq 100\,000$
- $1 \leq m_i \leq 1000$
- It is always possible to align Erlend's sheet to the start of the music piece.

Sample Input 1

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

Sample Output 1

```
right 2
```

Sample Input 2

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

Sample Output 2

```
left 1
```

Sample Input 3

```
2
10 100
10 100
```

Sample Output 3

```
you are ready!
```

Puzzling Poison Protection

Problem ID: puzzlingpoisonprotection

Illustration by Géo Dorival

Ivar is playing an RPG, and has gotten quite far into the game. By now he has gathered up a lot of different equipment: some nice gauntlets, a couple of boots, some rings, some chainmails, and even some nice hats.

Even though he has gotten all of this nice loot, he didn't know that the equipment may protect against poison: A piece of equipment can completely protect against all damage from a poison attack. It was only when he talked to his friend Ingve the other day that he found out, and now he's interested in knowing what kind of immunity his gear will provide.

It's not weird that Ivar didn't know about this though: The game won't actually tell you that a piece of equipment provides poison protection. The only way to find out is to equip it and write down if you got hurt or not by a poison attack.

The problem, of course, is that whenever an attack gets completely blocked, it could be any (or even multiple) of the equipped items that blocked the attack. So even though Ivar's interested in knowing what equipment protects against poison, he's not *that* interested in figuring out the exact details.

For that reason, he'll continue on with the story as usual. But before every fight, he'll randomise the equipment he wears and record the outcome of any poison attack.

Ivar's now recorded a lot of encounters, and wonder if you can help him deduce which pieces of equipment protect against poison.

Input

The first line contains six integers. It starts with the integer R, the number of records. Then five integers E_1, E_2, E_3, E_4, E_5 follow, denoting the number of items of type 1 (gauntlets), 2 (boots), 3 (rings), 4 (chainmails) and 5 (hats) that Ivar has in his inventory.

Next follows R lines, one for each poison attack recorded. The record starts with a character b_i, followed by five integers $e_{1,i}$, $e_{2,i}$, $e_{3,i}$, $e_{4,i}$, $e_{5,i}$. b_i is "`y`" if the attack was blocked, and "`n`" if it was not. Then follows the five items equipped when he was attacked.

Output

For each item type i, output a E_i long character string.

The character $C_{i,j}$ should be "`y`" if item $e_{i,j}$ protects against poison, "`n`" if it doesn't, and "`?`" if more information is needed to know if the item protects against poison.

Limits

- $0 \leq R \leq 1\,000\,000$
- $0 < E_1, E_2, E_3, E_4, E_5 < 50$
- $0 \leq e_{i,j} < E_i$
- No record will contradict another record.

Sample Input 1	Sample Output 1
2 1 1 1 2 3 n 0 0 0 1 0 y 0 0 0 0 0	n n n yn n??

Sample Input 2	Sample Output 2
2 2 2 1 1 1 n 0 0 0 0 0 y 1 1 0 0 0	n? n? n n n

Restaurant Rinser

Problem ID: restaurantrinser

Sanna the sanitiser is working the closing shift at the nearby fast food shop, and has the responsibility of cleaning all the tables. She wants to get home early and has used her knowledge of electronics and cybernetics to construct a robot that can clean tables for her! Unfortunately, she has not yet had time to program the planning algorithm which would make it go directly to the dirty tables, so for now it wanders around aimlessly cleaning everything it encounters.

The robot can move around the restuarant, which is a grid of tables. For every time step, the robot moves to one of the neighboring tables and cleans it. Since there is no planning involved, it moves to a random neighboring table! It can move in all four directions, but it will avoid walking into walls. Sanna is almost done cleaning the tables, and turns on the robot when there is exactly one dirty table left, hoping that it will clean it during the night. Help her figure out how what fraction of the possible paths the robot could take would end up cleaning the last dirty table. To be clear, we are asking about what fraction of all possible the robot could take would leave the restaurant clean, not the probability of it happening.

Input

The first line of input contains T, the number of timesteps the robot will complete during the night, and L, the length and width of the grid of tables. The next L lines each contain a string of L characters. The characters can be either "`C`", meaning that the table is clean, "`D`", meaning that the table is dirty, or "`S`", meaning it is the starting position of the robot. There will be exactly one "`S`" and one "`D`" in the input.

Output

Output the fraction of possible paths the robot could take that would end up cleaning the last dirty table. Since the numerator and denominator can be very large the answer should be given modulo $1\,000\,000\,007$. This means the answer should be equivalent to $\frac{A}{B}$ in modular arithmetic terms, where A is the total number of valid paths that clean up the last table, and B is the total number of valid paths. Since we are working modulo $1\,000\,000\,007$, the answer should be $A \cdot B^{-1}$, where B^{-1} is the multiplicative inverse of B in this modulus. The test data guarantees that this multiplicative inverse exists.

Limits

- $1 \leq T \leq 1000$
- $2 \leq L \leq 200$

Sample Input 1

```
5 3
CCD
CCC
SCC
```

Sample Output 1

```
843750006
```

Sample Input 2

```
2 4
DCCS
CCCC
CCCC
CCCC
```

Sample Output 2

```
0
```

Sausage Slicing

Problem ID: sausageslicing

Sausages are produced in long chains. Bjørnar the butcher has A customers in a group of customers that value the nutritional contents of a sausage, and B customers that value the taste. Every member in the same group has attributed a single number to each individual sausage which is how much someone in that group values it. When considering a set of sausages, a customer will value it the same as the sum of the value of each individual sausage. Bjørnar will use the minimal amount of slices such that the original string of sausages will be divided into precisely $A + B$ contiguous segments, one for each customer.

The customers will think the butcher is giving some customers special treatment if he does not divide the sausages fairly. The $A + B$ customers think the division is fair if they each receive sausages with a value at least $\frac{1}{A+B}$ of the total value of the sausages based on their own evaluations. Is it possible for the sausages to be distributed fairly using the minimal number of chops?

Input

The first line contains N, the number of sausages, A, the number of people in the first group, and B, the number of people in the second group. The second line contains N integers a_i, which is how much each person in the first group values the ith sausage. The third line contains N integers b_i, which is how much each person in the second group values the ith sausage.

Output

If the sausages can be divided fairly print "`yes`", otherwise print "`no`".

Limits

- $1 < N \leq 100$
- $0 < A, B \leq 25$
- $0 \leq a_i, b_i \leq 1000$

Sample Input 1

```
5 2 2
25 25 25 0 25
0 50 0 50 0
```

Sample Output 1

```
yes
```

Sample Input 2

```
5 2 2
25 24 24 3 24
0 50 0 50 0
```

Sample Output 2

```
no
```

Terminal Tools
Problem ID: terminaltools

Wanting to be more like Tore the Terminal Terminator you want to create the most magnificant command line tool ever. The greatest command line interface (CLI) must clearly be the one with the most options available. An option is simply a feature that can be activated when the program is run, by providing the name of the option. You are also a person that cares about efficiency, and want every feature to have an abbreviation that only consists of a single letter.

This can cause problems! If you have an option to run the tool in "help" mode, you might pick the abbreviation "h", but if you now want to add a feature to for instance hide some of the output, you cannot use the letter "h" to represent the word "hidden" because "h" is already in use! It is still possible to pick synonyms of the word "hidden" with other one-character abbreviations that are meaningful. This strategy leaves room for no more than 26 features! This is a hard limit by the english alphabet if we are to use lowercase letters.

When you create the most glorious CLI ever you simply cannot be limited by the 26 letters in the english alphabet, and instead choose to utilize an extended ascii characterset with 256 different letters in it. You have N different features in your program that the user should be able to refer to with a single character. You have also already come up with logical synonyms and characters that can represent a certain feature. Figure out if you can map the features to single characters such that a character at most represents a single feature.

Input

The first line contains N, the number of features, and M, the number of synonym-character pairs you have come up with in total for all the features. Then follows M lines. Each of those lines contains two integers, F_i describes which feature the synonym is associated, and U_i is the id of the ascii character character that can may represent the feature F_i.

Output

If it is possible to represent all the features with a single character then output "`possible`", if collisions are unavoidable your dreams are ruined and you should output "`impossible`". Note that the output is case sensitive.

Limits

- $1 \leq N \leq 256$
- $1 \leq M \leq N \times 256$
- $1 \leq F_i \leq N$
- $1 \leq U_i \leq 256$

Sample Input 1

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

Sample Output 1

```
possible
```

Sample Input 2

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

Sample Output 2

```
impossible
```