Problem A
Betting

The popular streaming platform switch.tv just unveiled their newest feature: switch betting. Streamers can now get their viewers to bet on two different options using switch points (patented).

Each viewer bets some number of switch points for one of the two options. The total amount of switch points bet by everyone is called the prize pool. The streamer will choose one of the options as the winner and the prize pool is split (not necessarily equally) between all the viewers who bet on that option; the more you bet on the option, the more of the prize pool you receive. In particular, if you contributed $p\%$ of all the bets for one of the options and that option wins, then you receive $p\%$ of the total prize pool.

The switch.tv team has come to you to compute what the switch point payout is for each viewer if their selected option wins. To do this, they ask you to find the switch-payout-ratio for each of the two options. Since the payout to each viewer is proportional to the number of switch points they put into the bet, the switch team will be able to use this ratio to determine each viewer's winnings.

For example, suppose a streamer created a switch bet where three viewers participated. Two viewers bet 10 and 30 switch points on option one and the last viewer bets 10 switch points on option two. We can see that option one has 80% of the bets and option two has 20% of the bets.

If option one wins, then the two viewers who bet on that receive 12.5 and 37.5 switch points, respectively, which means that the switch-payout-ratio is $1{:}1.25$ for option one. If option two wins, then the single viewer who bets on that receives 50 switch points, which means that the switch-payout-ratio is $1{:}5$.

Given the percentage of total bets for option one, help switch.tv by computing the switch-payout-ratio for the two options.

Input

The input consists of one integer a ($0 < a < 100$), which is the percentage of switch points bet on option one.

Output

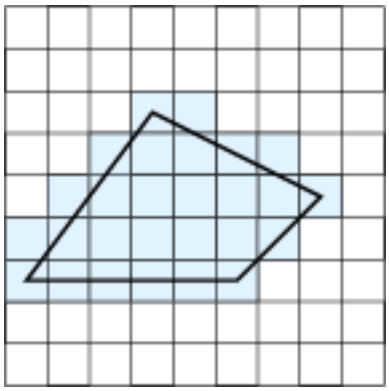
For each option (option one, then option two), display the number `x` such that `1 : x` is the switch-payout-ratio for that option. Your answer should have an absolute or relative error of at most 10^{-3}.

Sample Input 1	Sample Output 1
80	1.2500000000 5.0000000000

Sample Input 2	Sample Output 2
15	6.6666666667 1.1764705882

This page is intentionally left blank.

Problem B
Antialiasing



To reduce aliasing effects, computer graphics systems render a polygon by setting the brightness of each pixel proportional to the area of the pixel inside the polygon. If a pixel is completely inside the polygon, that pixel is set to the brightest intensity. If only half of the pixel is inside the polygon, then the pixel is set halfway between darkest and brightest.

Given a convex polygon and a pixel location, determine the fraction of the pixel's area that is inside the polygon. Each pixel is square, and is indexed by its row and column coordinates r and c. If a vertex of the polygon is located at (r, c), then the vertex is located at the center of the pixel at row r and column c. Rows are numbered from 0 starting from the top row, and columns are numbered from 0 starting from the leftmost column.

Input

The first line of input specifies two integers N ($3 \leq N \leq 100$), which is the number of vertices in the convex polygon, and Q ($1 \leq Q \leq 1\,000$), which is the number of queries. The next N lines each contains two integers r and c giving the coordinates of the polygon in counterclockwise order. The next Q lines each contains two integers r and c indicating the coordinates of the pixel we are interested in. It is guaranteed that the area of the polygon is positive. All coordinates satisfy $0 \leq r \leq 1\,000$ and $0 \leq c \leq 1\,000$.

Output

For each query, display a line indicating the fraction of the pixel's area inside the polygon. The fraction should be in lowest terms.

Sample Input 1	Sample Output 1
4 4 1 1 4 1 4 4 1 4 3 3 10 10 1 3 1 4	1/1 0/1 1/2 1/4

Sample Input 2	Sample Output 2
3 4 1 1 11 11 1 21 1 1 11 11 21 1 4 4	1/8 1/4 0/1 1/2

Problem C
Social Distancing

It's time for a social distancing party! A group of friends are sitting around a circular table where some seats are filled and some seats are empty. In particular, to maintain social distancing protocols, no two people are sitting directly beside each other.

They want to expand the party and include more friends, but no one is willing to move out of their current seat. Given the current table seating, determine the maximum number of additional people that can be seated such that there is still at least one empty seat between all pairs of people seated.

Input

The first line of input contains two integers S ($3 \leq S \leq 1\,000$), which is the number of seats at the table, and N ($1 \leq N \leq S/2$), which is the number of people that are already seated at the table.

Note that the seats of the table are numbered $1, 2, \ldots, S$ in a circular fashion: for each $1 \leq i < S$, seats numbered i and $i + 1$ are directly beside each other. Seats S and 1 are also directly beside each other.

The second line contains N integers $a_1, a_2, \ldots, a_N$ ($1 \leq a_1 < a_2 < \cdots < a_N \leq S$), which indicates that seat number a_i is currently occupied. No two occupied seats are directly beside each other.

Output

Display the maximum number of additional friends that can be seated at the table such that there is still at least one empty seat between all pairs of people seated.

Sample Input 1

```
9 2
2 6
```

Sample Output 1

```
2
```

Sample Input 2

```
10 3
1 4 7
```

Sample Output 2

```
1
```

Sample Input 3

```
6 2
2 5
```

Sample Output 3

```
0
```

Sample Input 4

```
100 5
7 14 47 78 99
```

Sample Output 4

```
43
```

Sample Input 5	Sample Output 5
6 1 3	2

Problem D
Pawn Shop

You run a tight ship at the pawn shop. You arrange certain items in the window to be displayed to the street. You sometimes display the same type of item multiple times. For simplicity, we think of the items on display as a sequence of values where the value represents the type of item being shown.

For example, your display could be this sequence:

$$1\ 2\ 6\ 2\ 7\ 9\ 8\ 5$$

After coming back from your latest vacation, you find that your staff has completely rearranged the display by moving items around. Yikes! For example, the display above could be rearranged to:

$$2\ 6\ 1\ 2\ 9\ 7\ 5\ 8$$

You fear this could be the cause of confusion and may scare off repeat customers. But you don't have time to move items back to their original positions.

As a compromise, you will put dividers up in the window to partition the displayed items into groups of consecutive items. Each group should be a rearrangement of the types of items that were in those positions in your preferred arrangement.

More precisely, let $a_1, \ldots, a_N$ denote the first sequence and $b_1, \ldots, b_N$ denote the second sequence. You may place dividers around $i, i+1, \ldots, j$ if $b_i, b_{i+1}, \ldots, b_j$ is a rearrangement of $a_i, a_{i+1}, \ldots, a_j$. You do not need to put a divider at the beginning or end of the sequence. Note, if $a_i = b_i$ then a group may be formed using just this single index i.

With the sequences above, you could place dividers # at three positions as indicated here:

$$2\ 6\ 1\ \#\ 2\ \#\ 7\ 9\ \#\ 5\ 8$$

It is not possible to divide the sequence into more than four groups that have this property.

Given the two sequences, determine how to partition the new sequence into the maximum possible number of groups.

Input

The first line of input contains a single integer N ($1 \leq N \leq 300\,000$), which is the length of the two sequences.

The next line contains N integers $a_1, \ldots, a_N$ ($1 \leq a_i \leq 10^9$), which is the original sequence.

The next line contains N integers $b_1, \ldots, b_N$ ($1 \leq b_i \leq 10^9$), which is the rearranged sequence. The values $b_1, \ldots, b_N$ are a rearrangement of the values $a_1, \ldots, a_N$.

Output

Display the rearranged sequence with a valid and maximum placing of dividers (#).

If there are multiple possible solutions, display any of them.

Sample Input 1

```
8
1 2 6 2 7 9 8 5
2 6 1 2 9 7 5 8
```

Sample Output 1

```
2 6 1 # 2 # 9 7 # 5 8
```

Sample Input 2

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

Sample Output 2

```
2 1 1 # 1
```

Sample Input 3

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

Sample Output 3

```
2 5 1
```

Problem E
Election Paradox

In Oddland, the leader of the country is determined by a democratic election. The country is divided into an odd number of regions, in which each region has an odd number of voters.

There are two (an even number!) political parties in Oddland, and the winning party is the one that wins the most number of regions. A party wins a region if it receives more votes than the other party in that region.

Under this system, it is possible that the losing party receives more votes than the winning party. For example, if there are three regions with 11, 3, and 3 people, respectively, then a party could receive 8, 1, and 1 votes and lose the election. In this case, the losing party received the majority of the votes in the total population.

Determine the largest number of votes a party can receive and still lose the election.

Input

The first line of input contains an odd integer N ($3 \leq N \leq 999$), which is the number of regions in Oddland.

The next line contains N odd integers p_i ($1 \leq p_i \leq 999$), which are the populations of the N cities.

Output

Display the largest number of votes a party can receive and still lose the election.

Sample Input 1	Sample Output 1
3 11 3 3	13

This page is intentionally left blank.

Problem F
Protect the Pollen!

The Flariana flowers and the bumblebees form one of the nicest partnerships in the rainforest. In spring, several flowers bloom and start producing pollen. Special vines form a network of bridges between the flowers. Using the vines, there is exactly one way to get from each flower to any other flower.

Every flower has a family of bees on it. This family protects all of the vines that touch that flower. This means that every vine is protected by two families. The family on flower k consists of s_k bees and has a pollination power of p_k.

One day, a bee scout announced that there is a new flower patch over the hill and they need a group of bees to help pollinate it.

As the bee queen, you must select a set of families to send on the mission. For every vine, at least one of the two families currently protecting it must stay behind so the vine remains protected. All bees in the selected families must go. You are willing to send at most S bees on the mission in total.

Determine the largest total pollination power that you can send on the mission.

Input

The first line contains the integer N ($1 \leq N \leq 300$), which is the number of flowers, and S ($1 \leq S \leq 300$), which is the maximum number of bees you can send on the mission. The flowers are numbered 1 to N.

The next N lines describe the families. Each of these lines contains two integers s_k ($1 \leq s_k \leq 300$), which is the number of bees in this family, and p_k ($1 \leq p_k \leq 100$), which is the pollination power of this family.

The last $N - 1$ lines describe the vines. Each of these lines contains two distinct integers u ($1 \leq u \leq N$) and v ($1 \leq v \leq N$), indicating that there is a vine between flowers u and v.

Output

Display the largest total pollination power that you can send on the mission.

Sample Input 1	Sample Output 1
5 10 2 1 2 2 2 4 2 8 2 16 1 2 2 3 3 4 4 5	21

Sample Input 2	Sample Output 2
7 10 1 7 2 4 5 18 2 3 3 12 9 20 2 8 1 2 1 3 2 4 2 5 3 6 3 7	33

Problem G
Loot Chest

Your favorite online game has a prize system. After each match you win, there is a $P\%$ chance you will receive a prize. This value P changes over time:

- every time you lose a match, P increases by Δ_L,
- every time you win a match but fail to receive a prize, P increases by Δ_W, and
- every time you win a match and receive a prize, P is set to 0.

Whenever P is increased, it is capped at 100. That is, if P is to be increased by Δ, P increases to $\min(P + \Delta, 100)$.

The developers just revealed one of the prizes this season is a silverback gorilla suit with star-shaped sunglasses. You want it! Each prize has a $G\%$ chance of being this gorilla suit.

You start with $P = 0\%$. You have an $L\%$ chance of losing each match you play. Given Δ_L, Δ_W, G, and L, compute the expected number of matches you will have to play until you obtain the silverback gorilla suit.

For example, in the first sample case you win every match you play and are guaranteed to receive a prize every 2 matches. Each prize has a 50% chance of being the gorilla suit. So you expect to obtain the gorilla suit after receiving 2 prizes. Thus, in expectation, it takes 4 matches to obtain the gorilla suit.

Input

The input consists of a single line containing four integers Δ_L ($1 \leq \Delta_L \leq 100$), Δ_W ($1 \leq \Delta_W \leq 100$), G ($1 \leq G \leq 100$), and L ($0 \leq L \leq 99$), which are described above.

Output

Display the expected number of matches you will play before you obtain the gorilla suit. Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
1 100 50 0	4

Sample Input 2	Sample Output 2
50 50 100 25	2.8333333333333333

Sample Input 3	Sample Output 3
1 100 10 0	20

Sample Input 4	Sample Output 4
2 3 10 80	197.00570671995630567

This page is intentionally left blank.

Problem H
RSA Mistake

An *RSA* number is a positive integer n that is the product of two distinct primes. For example, $10 = 2 \cdot 5$ and $77 = 7 \cdot 11$ are RSA numbers whereas $7 = 7, 9 = 3 \cdot 3$, and $105 = 3 \cdot 5 \cdot 7$ are not.

You are teaching a course that covers RSA cryptography. For one assignment problem, you asked students to generate RSA numbers. They were to submit two positive integers A, B. Ideally, these would be distinct prime numbers. But some students submitted incorrect solutions. If they were not distinct primes, partial credit can be earned if $A \cdot B$ is not an integer multiple of k^2 for any integer $k \geq 2$. If there is an integer $k \geq 2$ such that k^2 divides $A \cdot B$, then the student receives no credit.

For a pair of positive integers submitted by a student for the assignment, determine if they should receive full credit, partial credit, or no credit for this submission.

Note: In the sixth sample case below, the number $545\,528\,636\,581 \cdot 876\,571\,629\,707$ is divisible by $1\,000\,003^2$ and in the seventh sample case below, the number $431\,348\,146\,441 \cdot 3$ is divisible by $656\,771^2$.

Input

The input consists of a single line containing two integers A ($2 \leq A \leq 10^{12}$) and B ($2 \leq B \leq 10^{12}$), which are the two submitted numbers.

Output

Display if the student should receive `full credit`, `partial credit`, or `no credit` for the submitted numbers.

Sample Input 1	Sample Output 1
13 23	full credit

Sample Input 2	Sample Output 2
35 6	partial credit

Sample Input 3	Sample Output 3
4 5	no credit

Sample Input 4	Sample Output 4
17 17	no credit

Sample Input 5	Sample Output 5
15 21	no credit

Sample Input 6	Sample Output 6
545528636581 876571629707	no credit

Sample Input 7	Sample Output 7
431348146441 3	no credit

Problem I
Slide Count

In your programming class, you are given an assignment to analyze an integer array using a sliding window algorithm. Specifically, given N integers $w_1, \ldots, w_N$ and some constant C, the sliding window algorithm maintains start and end indices s and e such that

- initially $s = e = 1$;
- as long as $s \leq N$:
  - if $e + 1 > N$, then increment s;
  - else if $w_s + \cdots + w_{e+1} > C$, then increment s;
  - else increment e.

During the execution of this algorithm, each distinct pair of indices (s, e) defines a window. An element w_i belongs to the window defined by (s, e) if $s \leq i \leq e$. Notice that if $s > e$, the window is empty.

Consider the first sample input below. The windows appearing during the execution of the algorithm are defined by $(1, 1)$, $(1, 2)$, $(1, 3)$, $(2, 3)$, $(3, 3)$, $(3, 4)$, $(4, 4)$, $(5, 4)$, $(5, 5)$, and $(6, 5)$.

For each element w_i, determine how many different windows it belongs to during the execution of the sliding window algorithm.

Input

The first line of input contains two integers N ($1 \leq N \leq 100\,000$), which is the number of elements, and C ($1 \leq C \leq 1\,000\,000$), which is the sliding window constant.

The next line contains N integers $w_1, \ldots, w_N$ ($0 \leq w_i \leq C$).

Output

For each element, in order, display the number of different windows it belongs to during the execution of the algorithm.

Sample Input 1	Sample Output 1
5 3 1 1 1 2 2	3 3 4 2 1

Sample Input 2	Sample Output 2
5 10 1 2 3 4 5	4 4 4 5 2

Problem J
Snowball Fight

Back in my day, we were allowed to have snowball fights during recess. Me and my two friends would split up, build a fort, and stock it with snowballs. When the fighting started, we threw snowballs at each other's forts until there was one left standing. Those were the days.

There are three forts labelled `A`, `B`, and `C` that appear in a circle: with `B` to the left of `A`, `C` to the left of `B`, and `A` to the left of `C`.

The strengths of the forts are represented as nonnegative integers. If the strength of a fort is 0, then it is just rubble and the person in that fort no longer throws snowballs.

The fight proceeds in rounds. Each round, each person in a non-rubble fort picks a target. Their target is the fort with highest strength, apart from their own. If both possible targets have the same strength, the person chooses the fort on their left as the target. The people then simultaneously throw a single snowball at their chosen target. Each snowball reduces the strength of the target fort by 1. This repeats until there is at most one fort that is not reduced to rubble.

Given the initial strengths of the three forts, you are to determine if there is a fort that is not reduced to rubble and, if so, the remaining strength of that fort.

Input

Input contains a single line containing three integers N_A ($1 \leq N_A \leq 10^{18}$), which is the initial strength of fort `A`, N_B ($1 \leq N_B \leq 10^{18}$), which is the initial strength of fort `B`, and N_C ($1 \leq N_C \leq 10^{18}$), which is the initial strength of fort `C`.

Output

If all forts are reduced to rubble, display `Rubble!`. Otherwise, display `A`, `B`, or `C` indicating which fort was left standing followed by the remaining strength of that fort.

Sample Input 1	Sample Output 1
`10 3 1`	`A 3`

Sample Input 2	Sample Output 2
`3 2 1`	`Rubble!`

Sample Input 3	Sample Output 3
`2 3 2`	`C 1`

Sample Input 4	Sample Output 4
`100 101 100`	`A 1`

Sample Input 5	Sample Output 5
100 99 100	Rubble!

Sample Input 6	Sample Output 6
1000 5000 1000	B 1001

Sample Input 7	Sample Output 7
2000 1000 1000	C 1

Sample Input 8	Sample Output 8
1000000000000000 2000000000000000 4000000000000000	B 1

Sample Input 9	Sample Output 9
1000000000000000 2000000000000000 4000000000000001	Rubble!

Problem K
Team Change

Mr. Smith, the physical education teacher at Rocky Mountain High School, likes to take a hands-off approach to teaching gym class. What does that mean? A lot of dodgeball!

For the last few months, the students have been split into the same two teams. But now some students are asking for a change.

Some students want to be assigned to a specific team. Also, certain pairs of students that are currently on opposite teams have become *rivals* and are unwilling to be on each other's team.

Since Mr. Smith does not want to be complacent he has decided to shuffle the teams. He realizes it is not necessarily possible to form new teams satisfying both of the above constraints. To solve this, he will make some students sit the next game out.

Your job is to help Mr. Smith form the new teams that satisfy the above constraints such that the number of students who must sit out is minimum. Note that the teams do not need to have the same number of players nor does a team have to have any players at all.

Input

The first line of input contains two integers N ($1 \leq N \leq 1\,000$), which is the number of students, and R ($0 \leq R \leq 10\,000$), which is the number of rivalries.

The next line contains a string which describes the current teams. The string has length N consisting of characters `A` and `B`. The ith character on this line is the team that the ith student is currently playing for.

The next line contains a string which describes the students' requested teams. The string has length N consisting of characters `A`, `B`, and `?`. The ith character on this line is the desired team of the ith student or `?` if the student has no preference.

The next R lines describe the rivalries. Each line contains two distinct integers i ($1 \leq i \leq N$) and j ($1 \leq j \leq N$), which indicates that student i and student j are rivals. Rival students are on different teams. No rival pair of students will be reported more than once.

Output

Display a valid team formation with the minimum number of students sitting out. If the ith student is to sit out of the game, the ith character should be `X`. Otherwise, the ith character of this line should either be `A` or `B`, indicating which team this player is assigned to.

If there are multiple possible solutions, display any of them.

Sample Input 1

```
5 4
AAABB
??AAA
1 4
2 4
3 4
3 5
```

Sample Output 1

```
BBXAA
```

Sample Input 2

```
2 1
AB
AA
1 2
```

Sample Output 2

```
XA
```

Sample Input 3

```
8 8
ABBABAAA
?B?ABBAB
1 2
4 3
6 5
4 5
1 5
2 8
3 7
2 4
```

Sample Output 3

```
BXBAXBAB
```

Problem L
Ticket Completed?

Many are familiar with the board game Ticket To Ride[1] where players compete to build a railway empire, claiming routes between cities. The game consists of a map of cities and various rail segments each connecting two adjacent cities.

A key way to score points towards winning the game is to complete *Destination Tickets*. Each ticket specifies two distinct cities. A player earns the points that are indicated on the ticket if they have claimed one or more rail segments that form a path connecting the two cities.

There is one ticket for each distinct unordered pair of cities. In our version of the game, each player is randomly given a ticket and they have an equal probability of receiving any ticket. Given a list of rail segments you have already claimed, determine the probability you earn points from the ticket you are given.

Input

The first line of input contains two integers N ($2 \leq N \leq 10^5$), which is the number of cities, and M ($0 \leq M \leq 10^6$), which is the number of rail segments you have claimed.

The next M lines describe your claimed rail segments. Each line contains two distinct integers i ($1 \leq i \leq N$) and j ($1 \leq j \leq N$), which are the cities that this rail segment connects.

Output

Display the probability you earn points from the ticket you are given.

Your answer should have an absolute error of at most 10^{-6}.

[1] Ticket To Ride is copyrighted by Days of Wonder, Inc.

Sample Input 1	Sample Output 1
4 2 1 2 3 4	0.33333333333333333

Sample Input 2	Sample Output 2
5 4 1 5 2 3 2 4 3 4	0.4

Sample Input 3	Sample Output 3
7 5 1 2 2 3 3 4 5 6 6 7	0.42857142857142857143

Problem M
Trade Routes

All roads lead to Rome. In this case, we mean every road in the road network in the Roman Empire can be travelled in only one direction. Each city that is not Rome has exactly one road that leaves it and by following these roads, you will always end up in Rome.

Each city, including Rome itself, may create a trade route to Rome which brings Rome some value. These values are all distinct. Each city does not want to be too congested with traders so they constrain the number of trade routes that the city can be a part of.

We say that a city is part of a trade route if it is contained in the unique path from the city that created the trade route and Rome. A city is a part of its own trade route.

Given the city constraints, determine the maximum value that can be brought to Rome by choosing a subset of cities to create trade routes.

Input

The first line of input contains a single integer N ($2 \leq N \leq 300\,000$), which is the number of cities. The cities are numbered 1 to N. Rome is city number 1.

The next line contains $N-1$ integers $p_2, p_3, \ldots, p_N$ ($1 \leq p_i < i$), where p_i is the city you reach by following the single road out of city i.

The next line contains N integers $b_1, b_2, \ldots, b_N$ ($0 \leq b_i \leq N$), where b_i is the maximum number of trade routes that city i can be a part of.

The next line contains N distinct integers $v_1, v_2, \ldots, v_N$ ($0 \leq v_i \leq 10^9$), which is the value brought to Rome if city i creates a trade route.

Output

First display the maximum possible value Rome can receive from any valid subset of chosen trade routes. Next display T, the number of trade routes created, then display the T cities, in increasing order, that were selected.

If there are multiple possible solutions, display any of them.

Sample Input 1	Sample Output 1
7 1 1 2 2 3 3 2 1 2 1 1 1 1 6 5 3 8 4 7 1	15 2 4 6

Sample Input 2	Sample Output 2
9 1 1 2 3 3 4 4 4 4 4 2 4 1 0 1 1 1 100 30 10 0 50 200 12 15 13	195 4 1 2 5 8

Problem N
Wordle with Friends

Zoe and her friends enjoy playing Wordle[2] together and have decided to work cooperatively to solve the daily puzzle.

Wordle is a game where players get six attempts to guess a hidden 5-letter word. With each word guessed, the system will mark each letter with one of three feedback colors:

1. `Green` - this letter is in the word and occurs at this location.
2. `Yellow` - this letter is in the word, but not at this location.
3. `Gray` - this letter is not in the hidden word (with an exception for duplicate letters, see below).

Note that duplicate letters can be a little tricky. First, `Green` letters are marked. For a single letter, suppose there are X non-`Green` occurrences in the hidden word and Y non-`Green` occurrences in the guess. The *leftmost* $\min(X, Y)$ of the non-`Green` occurrences of this letter will be marked `Yellow` and the rest will be `Gray`.

For example, if the hidden word was `FREED` and a guessed word was `GEESE`, the feedback would show the second `E` (the third letter) in `Green`, and the first and third `E`s (second and fifth letters of `GEESE`) respectively in `Yellow` and `Gray`.

Knowing the list of all guessable words, help Zoe determine which words are still valid given their original guesses.

Input

The first line of input contains two integers N ($1 \leq N \leq 10$), which is the number of guesses Zoe and her friends have made, and W ($1 \leq W \leq 10^4$), which is the number of guessable words.

The next N lines describe the guesses. Each line contains two 5-letter strings g and f. The first string, g, is the guess which consists only of uppercase English letters and is in the list of guessable words. The second string, f, is the feedback. The feedback is composed of the characters `G`, `Y`, and `-`, respectively indicating `Green`, `Yellow`, and `Gray` for the guess.

The last W lines describe the list of distinct guessable words. Each line contains a 5-letter string of uppercase English letters.

[2] Wordle was created by Josh Wardle, and is now owned by the New York Times.

Output

Display all valid words, in the order they appear, from the guessable list of words. There will always be at least one valid word.

Sample Input 1

```
2 5
BERRY -G---
APPLE ---YY
MELON
BERRY
LEMON
LIMES
APPLE
```

Sample Output 1

```
MELON
LEMON
```

Sample Input 2

```
3 5
BERRY -G---
APPLE ---YY
LIMES G-GY-
APPLE
BERRY
LEMON
LIMES
MELON
```

Sample Output 2

```
LEMON
```

Sample Input 3

```
3 5
BLANK --Y--
SIGHS ----G
STORM YGG-Y
ATOMS
BLANK
MOATS
SIGHS
STORM
```

Sample Output 3

```
ATOMS
```

Sample Input 4	Sample Output 4
4 5 FRUIT -G--Y NUTTY --Y-- ROOTS Y--YG SEEDS -YG-G FRUIT NUTTY ROOTS SEEDS TREES	TREES

This page is intentionally left blank.