icpc.foundation
icpc International Collegiate Programming Contest
TWO SIGMA north america sponsor
JET BRAINS programming tools sponsor

Official Problem Set

icpc regionals 2018

PLEASE!

DO
NOT
TOUCH
ANYTHING

UNTIL
TOLD TO DO SO!

2018 ICPC
North Central North America
Regional Contest

This page is intentionally left blank.

Problems

A Pokegene
B Maximum Subarrays
C Rational Ratio
D Travel the Skies
E Euler's Number
F Lipschitz Constant
G Tima goes to Xentopia
H New Salaries
I Other Side
J Kaleidoscopic Palindromes

This page is intentionally left blank.

Problem A
Pokegene

Professor Oak, the friendly neighbourhood Pokémon Professor, has been getting a lot of questions about Pokémon ancestors lately.

One of the things a lot of Pokémon trainers are curious about is finding common ancestors of Pokémon in their Pokédex. In particular, each Pokémon Trainer wants to know about the number of Pokémon that are ancestors of *exactly* L of their Pokémon. The value of L, too, is decided by the trainer asking the question.

In Pokémon world, given the genome strings of Pokémon A and Pokémon B, Pokémon A is considered an ancestor of Pokémon B if and only if the genome string of Pokémon A is a prefix of Pokémon B's genome string. Also, all non-empty genome strings correspond to a real Pokémon in the Pokémon world.

Given Professor Oak's database of Pokémon genomes, help him answer the trainers' queries.

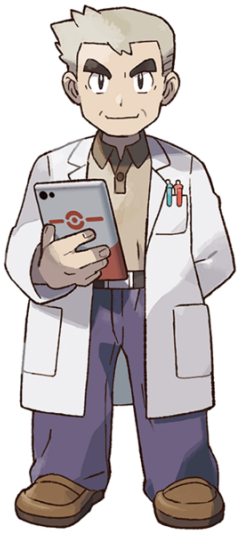


Image by Let's Go, Pikachu! and Let's Go, Eevee!.

Input

The first line of input will contain the integer N, ($1 \leq N \leq 200\,000$), the number of Pokémon in Professor Oak's database, and the integer Q, ($1 \leq Q \leq 200\,000$), the number of Pokémon trainers that have questions for Professor Oak. The following N lines contain a string, each denoting a Pokémon genome string from Professor Oak's database. All genome strings are *distinct* and consist of lowercase English letters.

Then $2 \cdot Q$ lines follow. Two lines describe each Pokémon trainer's query. The first line of each query contains two integers: K, ($1 \leq K \leq N$), the number of Pokémon they own, and L, ($1 \leq L \leq K$), the value described in the statement. The second line contains K distinct space-separated integers. An integer x in the list of K integers indicates that this trainer owns the xth Pokémon from Professor Oak's database, where the Pokémon Professor Oak owns are numbered from 1 to N.

The sum of the lengths of the genome strings in input does not exceed $200\,000$ characters and the sum of K over all queries does not exceed $1\,000\,000$.

Output

For each Pokémon trainer's query, print the number of Pokémon that are ancestors of exactly L of the K Pokémon they own. Assume that all non-empty genome strings correspond to a Pokémon. Note that L is defined in the query, and does *not* necessarily remain the same for all queries.

Example

There are five Pokémon in Professor Oak's database: "nib", "abcd", "abee", "abced", and "nit". There are two trainers with queries.

The first Pokémon trainer owns Pokémon with genome strings "nib", "abcd", "nit", and "abee", and would like to know how many ancestors there are of exactly two of her Pokémon. Pokémon with genome strings "a" and "ab" are ancestors of "abcd" and "abee", and Pokémon with genome strings "n" and "ni" are ancestors of "nib" and "nit", therefore the answer is 4.

The second Pokémon trainer owns Pokémon with genome strings "abcd", "abee", and "abced", and would also

like to know how many ancestors there are of exactly two of her Pokémon. Only the Pokémon with genome string "abc" is an ancestor of exactly 2 of the owned Pokémon, "abcd" and "abced", therefore the answer is 1.

Sample Input 1	Sample Output 1
5 2 nib abcd abee abced nit 4 2 1 2 5 3 3 2 2 3 4	4 1

Problem B
Maximum Subarrays

You may have heard of the "maximum subarray problem" from your university's undergraduate algorithms course. The problem goes like this: You are given an array A of n integers, and the task is to find a contiguous subarray of A whose sum is maximum. For example, in the array below:

```
A := [-2, 3, 5, -7, 8, 13, -20, 14, 1],
```

the solution is the subarray from the second index (the number 3) to the sixth index (the number 13), with total sum 22. The problem can be solved via divide and conquer, in $O(n \log n)$ time, or dynamic programming, in $O(n)$ time.

Being a strong student in algorithms yourself, you are of course familiar with both approaches, so we won't bother explaining them here. However, your classmate Steve, a *not-so-strong* student in algorithms, has been bragging to you about the maximum subarray problem. He claims that not only can he solve the maximum subarray problem in linear time, he can even solve what he calls the "k-maximum subarray problem," ALSO in linear time! What is the "k-Maximum subarray problem," you ask? In a condescending tone, Steve explains: It is the natural generalization of the maximum subarray problem. Given an array A of n integers, you must find k disjoint, contiguous subarrays of A so that their total sum is maximum.

You're not at all confident that Steve knows how to solve the "k-maximum subarray problem" in linear time, let alone that there even exists a linear time solution. You have gone to every algorithms lecture, however, a near-infinite improvement on Steve's attendance, and so you believe if anyone is to be the authority on the problem, it probably should be you.

You decide to write a program to solve the "k-maximum subarray problem." You can use this to verify Steve's likely-incorrect algorithm. To simplify things, you make the program just output the value of an optimal solution. You can modify it later if you want the solution itself. Furthermore, you decide your algorithm is sufficiently efficient as long as its running time is bounded by a small polynomial in n and k. You think coming up with an actually-correct linear time solution would probably take you awhile, and you've decided you only want to spend, say, a maximum of five hours on your code.

Input

The input will consist of two lines. On the first line are the integers n and k, ($1 \leq k \leq n \leq 5\,000$). On the second line are n integers, representing the array A. The integers in array A will be between -10^9 and 10^9, inclusive.

Output

Output the maximum possible total sum of k disjoint contiguous subarrays of array A. Although the subarrays are required to be disjoint, a subarray may end at index i and another subarray start at index $i+1$. No subarray is allowed to be empty.

Sample Input 1	Sample Output 1
9 1 -2 3 5 -7 8 13 -20 14 1	22

Sample Input 2	Sample Output 2
11 3 23 -10 12 -2 -33 13 -5 55 23 -8 13	126

This page is intentionally left blank.

Problem C
Rational Ratio

Every (positive) rational number can be expressed as a ratio of two (positive) integers. However, in decimal form, rational numbers often have an infinitely repeating pattern, e.g., $1/7 = 0.142857142857142857...$ A convenient way of writing this repeating pattern is to put a bar over the first occurrence of the repeating part, so $1/7$ would be written:

$$0.\overline{142857}.$$

Given a rational number consisting of a series of digits, followed by a decimal point, followed by more digits, and then a number indicating how many of the rightmost digits repeat (i.e., the number of digits under the bar), your task is to find the ratio of two integers, in the most reduced form, that represent the same rational number. For example, for the input "0.142857 6" you should find $1/7$.

Input

The input will be a single line with two numbers separated by one space. The first number will consist of 1 to 3 digits (0–9), followed by a decimal point, followed by 1 to 11 digits (0–9), representing the decimal form of the number, possibly with leading zeros. The second number will be a positive integer indicating how many of the rightmost digits of the preceding number repeat. The first number will always be greater than 0. The second number will never be less than 1 nor larger than the number of digits to the right of the decimal point.

Output

Print the corresponding fraction in its most reduced form, that is, the fraction with the smallest possible integer values in the numerator and denominator.

Sample Input 1	Sample Output 1
0.142857 6	1/7

Sample Input 2	Sample Output 2
1.6 1	5/3

Sample Input 3	Sample Output 3
123.456 2	61111/495

This page is intentionally left blank.

Problem D
Travel the Skies


Wikimedia, cc-by-sa

One day your boss explains his new planning scheme for your company, Fly Away Air. Rather than having customers book tickets between destinations, they'll say when they want to leave and from where, and your company will take care of the rest. That means you get to generate their flight schedules and destinations for them! He has an eye on the books though and wants to make sure all your flights are fully booked. He tasks you with the job of determining if a set of flight plans in a given flight window is financially optimal in that regard.

You assure him that he put his trust—and his pocket book—in the hands of the right employee. Your job is to plan flight schedules for customers so that you fill each of the flights scheduled in the given flight window. To ensure you don't lose customers due to air sickness, you decide that each customer can only take one flight a day. Further, since you're sure that all of your customers are gracious folks, you decide to help your boss out and let them fly on any day on or after their suggested departure date. Finally, to simplify things, you do not worry about ensuring each customer return to their original departure airport, though this is allowed to be scheduled. If needed, they can book their own return flights!

Input

On the first line, you're given three integers: k, ($2 \leq k \leq 12$), the number of airports; n, ($1 \leq n \leq 8$), the number of days of the flight departure window; and m, ($1 \leq m \leq k \cdot (k-1) \cdot n$); the number of flights in the window. Then, m lines follow with four integers each: u, ($1 \leq u \leq k$), the flight's starting location; v, ($1 \leq v \leq k$, $u \neq v$), the flight's ending destination; d, ($1 \leq d \leq n$), the day on which the flight flies in the window; and z, ($1 \leq z \leq 30\,000$), the capacity of the flight. It is guaranteed there will be at most one flight in each direction between any two airports on a given day. Following are kn lines with three integers each: a, ($1 \leq a \leq k$), the airport; b, ($1 \leq b \leq n$), the day; and c, ($1 \leq c \leq 30\,000$), the number of customers wishing to begin their travels by leaving their homes to their local airport a on day b (notably, this does not include those who may be arriving from other flights, which is for you to decide). Each airport-day pair will appear exactly once.

Output

Output only the word `optimal` if all m flights can be filled to capacity, else output only the word `suboptimal` if it is not possible to fill all m flights. It is not necessary that each customer arriving at an airport ever be booked on a flight.

Example

Consider two airports: Chicago and Minneapolis, across two days. There are two flights: one from Chicago to Minneapolis on day 1, with capacity 30, and one from Minneapolis to Chicago on day 2, with capacity 50. On day 1, 10 people arrive at the airport in Minneapolis from their homes and 30 people arrive at the airport from their homes in Chicago. On day 2, 10 people arrive at both the Minneapolis and Chicago airports from their homes. We can fill all flights as follows. The first flight, on day 1, can take the 30 people that arrived in Chicago that day to Minneapolis.

The second flight, on day 2, can take the 30 people that arrived in Minneapolis from Chicago in the previous day's flight, plus it can take the 10 people that arrived at the Minneapolis airport on day 1 and the additional 10 people that arrived at the Minneapolis airport on day 2. Thus this flight plan is `optimal`.

Sample Input 1	Sample Output 1
2 2 2 1 2 1 30 2 1 2 50 2 1 10 1 1 30 1 2 10 2 2 10	optimal

Sample Input 2	Sample Output 2
2 1 1 1 2 1 2 1 1 1 2 1 1	suboptimal

Problem E
Euler's Number

Euler's number (you may know it better as just e) has a special place in mathematics. You may have encountered e in calculus or economics (for computing compound interest), or perhaps as the base of the natural logarithm, $\ln x$, on your calculator.

While e can be calculated as a limit, there is a good approximation that can be made using discrete mathematics. The formula for e is:

$$\begin{aligned} e &= \sum_{i=0}^{n} \frac{1}{i!} \\ &= \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \cdots \end{aligned}$$

Note that $0! = 1$. Now as n approaches ∞, the series converges to e. When n is any positive constant, the formula serves as an approximation of the actual value of e. (For example, at $n = 10$ the approximation is already accurate to 7 decimals.)

You will be given a single input, a value of n, and your job is to compute the approximation of e for that value of n.

Input

A single integer n, ranging from 0 to 10 000.

Output

A single real number – the approximation of e computed by the formula with the given n. All output must be accurate to an absolute or relative error of at most 10^{-12}.

Sample Input 1	Sample Output 1
3	2.6666666666666665

Sample Input 2	Sample Output 2
15	2.718281828458995

This page is intentionally left blank.

Problem F
Lipschitz Constant

Today you are doing your calculus homework, and you are tasked with finding a Lipschitz constant for a function *f(x)*, which is defined for N integer numbers x and produces real values. Formally, the Lipschitz constant for a function f is the smallest real number L such that for any x and y with *f(x)* and *f(y)* defined we have:

$$|f(x) - f(y)| \leq L \cdot |x - y|.$$

Input

The first line contains N – the number of points for which f is defined. The next N lines each contain an integer x and a real number z, which mean that $f(x) = z$. Input satisfies the following constraints:

- $2 \leq N \leq 200\,000$.
- All x and z are in the range $-10^9 \leq x, z \leq 10^9$.
- All x in the input are distinct.

Output

Print one number – the Lipschitz constant. The result will be considered correct if it is within an absolute error of 10^{-4} from the jury's answer.

Sample Input 1

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

Sample Output 1

```
2
```

Sample Input 2

```
2
1 4
2 2
```

Sample Output 2

```
2
```

Sample Input 3

```
4
-10 6.342
-7 3
46 18.1
2 -34
```

Sample Output 3

```
4.111111111
```

This page is intentionally left blank.

Problem G
Tima goes to Xentopia


Image by WikiImages.

The city of Xentopia has a well-connected railway network. The city has N junctions numbered from 1 to N. There are M pairs of railway tracks that exist between the junctions. Trains can travel in both directions on each track. Each railway track is labelled either red, blue, or white in colour.

Tima, a tourist in the city, wants to travel from junction S to junction T in the minimum possible time. She has a map of the railway network that she can use to achieve this goal.

Tima, being rather eccentric, has an interesting constraint for her travel: She wants to travel via exactly k_1 red tracks, exactly k_2 blue tracks, and any number of white tracks, in any order. She is fine with using a railway track more than once.

Can you tell the minimum time Tima will take to go from S to T, ensuring that her constraint is not violated?

Input

The first line contains four space-separated integers: N, ($1 \leq N \leq 450$); M, ($1 \leq M \leq 1\,100$); k_1; and k_2, ($0 \leq k_1, k_2 \leq 800$, $k_1 \cdot k_2 \leq 800$). Following are M lines. Each line contains four space-separated integers: U V X C, denoting that a track exists between junction U and junction V, ($1 \leq U, V \leq N, U \neq V$); the train covers this track in X seconds, ($0 \leq X \leq 10^9$); and the track is labelled colour C, ($0 \leq C \leq 2$). A white track is denoted by $C = 0$, a red track is denoted by $C = 1$, and a blue track is denoted by $C = 2$.

The last line contains two space-separated integers S, ($1 \leq S \leq N$), and T, ($1 \leq T \leq N$), the source and destination of Tima's journey, respectively. Note: S may be equal to T.

Output

Print a single integer denoting the total time Tima would take. If it is not possible for Tima to reach her destination using exactly k_1 red tracks, k_2 blue tracks, and any number of white tracks, output `-1`.

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

Sample Input 2	Sample Output 2
4 3 200 1 1 2 1 1 2 3 1 0 2 4 1 2 1 3	-1

Problem H
New Salaries

Oh no! As a result of recent elections the "Random Laws" party took control of the government. This is going to have bad consequences for Mr. Bourgeois' company, which just approved the way new salaries will be calculated. The company has N workers and the salary for worker i is going to be determined as a number drawn uniformly from the range $[L_i, R_i]$. Since the company already figured out which workers are the most efficient ones, for each i in $[2, N]$, we know that $L_{i-1} \leq L_i$ and $R_{i-1} \leq R_i$, but note that as a result of chance, worker $i-1$ might still end up with a larger salary than worker i.

The new government introduced a law, where any worker who got a smaller salary than a coworker can sue the company for the amount of their difference. What's even more atrocious is that they can do it for every worker who got a larger salary. So if there were three employees: Alice, Bob, and Charlie, who got salaries of 1, 3, and 7 coins respectively, then employee Bob can sue with regards to Charlie for 4 coins, while Alice can sue for 2 coins because of Bob and for 6 coins because of Charlie. The total amount of damages the company will have to pay is 12.

While the exact salary amounts are not known yet, Mr. Bourgeois would like to find out the expected amount of damages that his company will have to pay. Since the answer can be very big, output the answer divided by N^2.

Input

The first line contains N, ($1 \leq N \leq 100\,000$). The next N lines each contain two real numbers L_i and R_i ($1 \leq L_i \leq R_i \leq 10^6$). All real numbers in the input have at most 6 digits after the decimal point.

Output

Output one number: expected payment divided by N^2. Your answer will be considered correct if its absolute or relative error is less than 10^{-4}.

Sample Input 1

```
2
1.2 10.2
2.2 15.2
```

Sample Output 1

```
1.114672365
```

Sample Input 2

```
3
1 3
3 5
100 120
```

Sample Output 2

```
24
```

This page is intentionally left blank.

Problem I
Other Side

John Doe wants to transport his possessions from one bank of Lake Michigan to the other. His possessions consist of W wolves, S sheep, and C cabbages. The transportation will be carried out using a boat that can hold up to K of these items at the same time. During each step, John can take some items from one bank and transfer them to the other bank. Unfortunately, when left unsupervised, wolves will eat sheep and sheep will eat cabbages (but wolves don't eat cabbages). John doesn't want to lose his possessions, so he has to devise a scheme such that this doesn't happen. With John present, any combination of items is allowed (both on the bank and in the boat). This is also true during the loading process. Since John isn't very good at solving problems like this, he asks you to help him.

Indiana Dunes State Park. Image by pixabay.

Input

Input contains a single line with four integers: W, S, C, K. The input satisfies the following constraints:

$$0 \le W, S, C, K \le 10^6,$$

$$1 \le max(W, S, C).$$

Output

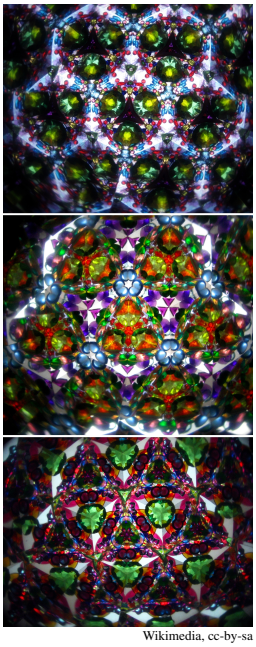
If it's possible to perform the transportation without the loss of items, print `YES`, otherwise print `NO`.

Sample Input 1	Sample Output 1
1 1 1 1	YES

Sample Input 2	Sample Output 2
2 2 0 1	NO

This page is intentionally left blank.

Problem J
Kaleidoscopic Palindromes


Wikimedia, cc-by-sa

Nicholas Neverson was a student at Northlings Neverland Academy. As with any daydreaming student, Nicholas was playing around with a Kaleidoscope one day instead of paying attention to the teacher. Since this was math class, his daydreams quickly turned to palindromic numbers. A palindromic number is any number which reads the same forwards and backwards.

He describes his vision to you at lunch: numbers which are palindromic in several bases at once. Nicholas wonders how many such numbers exist. You decide you can quickly code up a program that given a range and a number k, outputs the number of numbers palindromic in all bases j, $2 \leq j \leq k$, in that range.

Input

Input consists of three space-separated integers: a, b, and k. The input satisfies the following constraints:

$$0 \leq a \leq b \leq 2\,000\,000,$$

$$2 \leq k \leq 100\,000.$$

Output

Output the quantity of numbers between a and b inclusive which are palindromes in every base j, for $2 \leq j \leq k$.

Sample Input 1	Sample Output 1
1 356 2	36

Sample Input 2	Sample Output 2
18 118 13	0

This page is intentionally left blank.