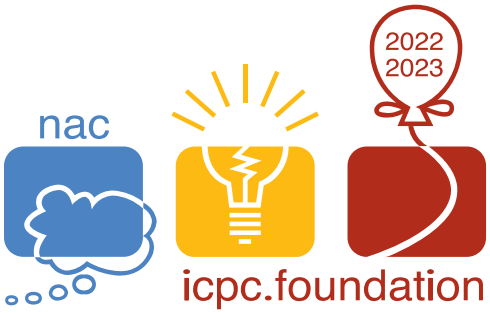
icpc international collegiate programming contest

ICPC North America Contests

Rocky Mountain Regional Contest

Official Problem Set



icpc global sponsor
programming tools

upsilon pi epsilon
honor society

FEBRUARY 25, 2023

Contest Problems

A: Blueberry Waffle
B: Profitable Trip
C: Champernowne Count
D: Triangle Containment
E: Color Tubes
F: Food Processor
G: Greedy Increasing Subsequences
H: Branch Manager
I : I Could Have Won
J : Sun and Moon
K: Everything Is A Nail
L: Family Visits

This contest contains 12 problems over 28 pages. Good luck.

For problems that state "*Your answer should have a relative error of less than* 10^{-9}", your answer, x, will be compared to the correct answer, y. If $\frac{|x-y|}{|y|} < 10^{-9}$, then your answer will be considered correct.

For problems that state "*Your answer should have an absolute error of less than* 10^{-9}", your answer, x, will be compared to the correct answer, y. If $|x - y| < 10^{-9}$, then your answer will be considered correct.

Definition 1

For problems that ask for a result modulo m:
If the correct answer to the problem is the integer b, then you should display the unique value a such that:

- $0 \leq a < m$

  and

- $(a - b)$ is a multiple of m.

Definition 2

A string $s_1 s_2 \cdots s_n$ is lexicographically smaller than $t_1 t_2 \cdots t_\ell$ if

- there exists $k \leq \min(n, \ell)$ such that $s_i = t_i$ for all $1 \leq i < k$ and $s_k < t_k$

  or

- $s_i = t_i$ for all $1 \leq i \leq \min(n, \ell)$ and $n < \ell$.

Definition 3

- Uppercase letters are the uppercase English letters $(A, B, \ldots, Z)$.
- Lowercase letters are the lowercase English letters $(a, b, \ldots, z)$.

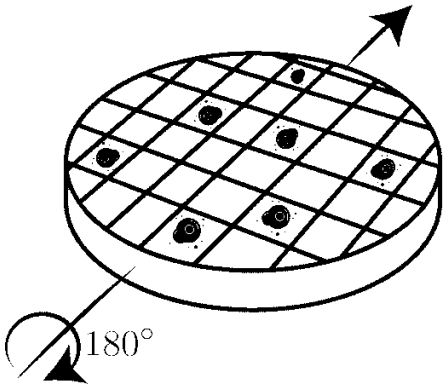
This page is intentionally left blank.

Problem A
Blueberry Waffle
Time limit: 1 second

You are using a waffle maker machine to make a delicious blueberry waffle. One side of your waffle is covered in blueberries, while the other side is plain. Initially, the cooking pan of the waffle maker lies horizontally. Once started, the cooking pan will rotate at a constant speed for a fixed duration, then stop. The cooking time is set so that when the waffle maker stops, the cooking pan will not be in a vertical position.

If the cooking pan is not horizontal after this time, the waffle maker will return to a horizontal position via the smallest rotation possible. Therefore, the waffle maker will rotate less than 90 degrees, either forward or backward, until the cooking pan is horizontal again.

The pan rotates at a rate of 180 degrees every r seconds, and stops after f seconds. You don't want to take out your waffle with its blueberry side down. Therefore you'd like to figure out whether the blueberry side of the waffle is up or down after the cooking pan returns to a horizontal position.



Input

The single line of input contains two integers r and f ($1 \leq r, f \leq 10^4$). The pan rotates at a rate of 180 degrees every r seconds, and stops after f seconds. It is guaranteed that after f seconds the cooking pan is not at a vertical position.

Output

Output a single line with a single string, which is `up` if the blueberry side of the waffle is up, or `down` otherwise.

Sample Input 1	Sample Output 1
10 20	up

Sample Input 2	Sample Output 2
10 34	down

Sample Input 3	Sample Output 3
10 47	down

Problem B
Profitable Trip
Time limit: 7 seconds

You are planning a road trip. You have carefully mapped out all potential waypoints that you may stop at. From each waypoint, there are roads that allow you to drive to other waypoints, but roads can only be used in one direction. In order to alleviate traffic jams, the road planners have decided to require tolls on the more popular roads. The less popular roads may even pay you to drive on them.

As a result, it may be possible to make a profit from the road trip. But there is a catch—your wallet can only hold a maximum of w dollars more than what you have started with. You are allowed to get paid even when your wallet is full, but you will not be able to keep more than w dollars over what you started with. You may assume that whenever you need to pay a toll, you will have money to pay.

What is the maximum profit you can make on your trip?

Input

The first line of input contains three integers n, m, and w ($1 \leq n, m \leq 2\,000$, $1 \leq w \leq 100$), where n is the number of waypoints, m is the number of roads, and w is the additional capacity of your wallet. The waypoints are numbered 1 through n. The first waypoint (1) is the start of your road trip, and the last waypoint (n) is the destination.

Each of the next m lines contains three integers u, v, and t ($1 \leq u, v \leq n$, $-100 \leq t \leq 100$), indicating that there is a road from waypoint u to waypoint v, with a gain of t dollars. If $t > 0$, then you are paid t dollars to use the road. If $t < 0$, you must pay a toll of $|t|$ dollars to use the road, resulting in a change of t to your profit. It is guaranteed that $u \neq v$, and there is at most one road from u to v (but there can also be a road from v to u).

You may assume that it is possible to reach waypoint n from waypoint 1.

Output

Output a single integer, which is the maximum profit you can make during the road trip. If there is a loss, output the loss as a negative number.

Sample Input 1	Sample Output 1
4 4 9 1 2 5 1 3 -2 2 4 1 3 4 10	8

Sample Input 2

```
4 4 7
1 2 5
1 3 -2
2 4 1
3 4 10
```

Sample Output 2

```
7
```

Sample Input 3

```
3 3 5
1 3 -10
3 2 2
2 3 -1
```

Sample Output 3

```
4
```

Problem C
Champernowne Count
Time limit: 1 second

The nth Champernowne word is obtained by writing down the first n positive integers and concatenating them together. For example, the 10th Champernowne word is "12345678910".

Given two positive integers n and k, count how many of the first n Champernowne words are divisible by k.

Input

The single line of input contains two integers, n ($1 \leq n \leq 10^5$) and k ($1 \leq k \leq 10^9$).

Output

Output a single integer, which is a count of the first n Champernowne words divisible by k.

Sample Input 1	Sample Output 1
4 2	2

Sample Input 2	Sample Output 2
100 7	14

Sample Input 3	Sample Output 3
314 159	4

Sample Input 4	Sample Output 4
100000 999809848	1

This page is intentionally left blank.

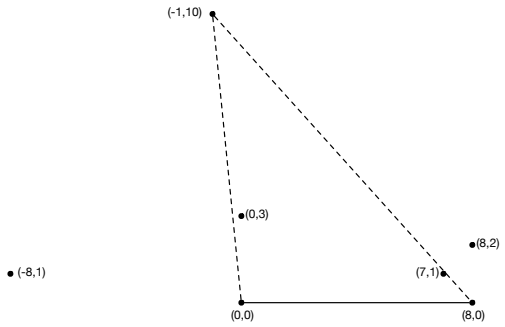
Problem D
Triangle Containment
Time limit: 4 seconds

You recently discovered there is treasure buried on your farm land. A **lot** of treasure! You quickly decide to put a fence around the land.

Alas, you have but a single fence post! You will have to drive to town to get more fencing material. But you can't just leave the land as open as it is, so you decide to create a makeshift fence to protect some of the treasure while you are gone. You will place the post in the ground and run some wire in a straight line between two sides of your barn wall and the fence post to section off a triangular area. Also, the ground is very hard: only places that were dug up to bury a treasure are soft enough for you to quickly place the fence post.

To figure out the best option, you first calculate the following. For each of the treasures in your field, if you were to place the fence post at that treasure and complete the fence as described, then what is the total value of all treasures that would be enclosed by the fence? Note that the treasure under the post you place is not considered enclosed by the fence (it might not be safe since someone could dig around the post).

Sample Input 1 is illustrated below. The triangle that includes the point $(-1, 10)$ encloses exactly two other treasure points which have total value $4 + 8 = 12$.



Input

The first line of input contains two integers n ($1 \leq n \leq 10^5$) and x ($1 \leq x \leq 10^9$), where n is the number of treasure points and x fixes the two corners of the barn wall at locations $(0, 0)$ and $(x, 0)$.

Each of the next n lines contains three integers x, y, and v ($-10^9 \leq x \leq 10^9$, $1 \leq y \leq 10^9$, and $1 \leq v \leq 10^9$) giving the location (x, y) and value v of one of the treasure points. All of these points are distinct. It is also guaranteed that for each point, the triangle formed with the barn wall will not contain any other treasure point on its boundary.

Output

Output n lines, one for each treasure point in the order of the input. For each point output a single integer, which is the total value of all points in the interior of the triangle that point forms with the barn wall. Note that the value of the point itself should be excluded from this sum.

Sample Input 1	Sample Output 1
5 8 -8 1 1 -1 10 2 0 3 4 7 1 8 8 2 16	0 12 0 0 8

Sample Input 2	Sample Output 2
6 6 0 1 1 2 3 10 2 5 100 3 1 1000 3 5 10000 4 5 100000	0 1000 1010 0 1010 1000

Problem E
Color Tubes
Time limit: 1 second

There is a new puzzle generating buzz on social media—Color Tubes. The rules are relatively simple: you are given $n+1$ tubes filled with $3n$ colored balls. Each tube can hold at most 3 balls, and each color appears on exactly 3 balls (so there are n colors).

Using a series of moves, you are supposed to reach a Color Tubes state—each tube should either hold balls of a single color or it should be empty.

The only move allowed is to take the top ball from one tube and place it into a different tube that has room for it (i.e. holds at most two balls before the move).

You want to write a program to solve this puzzle for you. Initially, you are not interested in an optimal solution, but you want your program to be good enough to solve any puzzle configuration using at most $20n$ moves.

Input

The first line of input contains a single integer n ($1 \leq n \leq 1\,000$), which is the number of colors.

Each of the next $n+1$ lines contains three integers b, m and t ($0 \leq b, m, t \leq n$), which are the descriptions of each tube, where b is the color of the ball on the bottom, m is the color of the ball in the middle, and t is the color of the ball on the top.

The tubes are numbered from 1 to $n+1$ and are listed in order. The colors are numbered from 1 to n. The number 0 describes an empty space. It is guaranteed that no empty space will be below a colored ball.

Output

On the first line output an integer m, the number of moves that your program will use to solve the puzzle. Remember, m has to be at most $20n$.

On the next m lines, output two space-separated integers u and v that describe a move ($1 \leq u, v \leq n+1$). In each move, you are taking the uppermost ball out of tube u and placing it in tube v, where it will fall until it hits the uppermost ball already in that tube, or the bottom of the tube if the tube is empty.

Your solution will be deemed incorrect if it uses more than $20n$ moves, or any of the moves are not allowed, or the final configuration is not a Color Tubes state.

Sample Input 1

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

Sample Output 1

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

Sample Input 2

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

Sample Output 2

```
0
```

Problem F
Food Processor
Time limit: 2 seconds

You have a food processor with a variety of blades that can be attached to it, as well as some food you would like to process into smaller pieces.

The food processor can have one blade attached at any time. Each blade processes food by reducing its average piece size at a particular exponential rate, but it also has a maximum average piece size requirement; if the average piece size of the food is too big for the blade, the food processor will get stuck. Given a starting average food piece size, a target average piece size, and a set of blades for your food processor, determine the minimum amount of processing time needed to process your food into the target average piece size.

Note that we only care about the time spent actively processing food; we do not track time spent switching out blades or loading/unloading the food processor.

Input

The first line of input contains three integers s, t, and n ($1 \leq t < s \leq 10^6, 1 \leq n \leq 10^5$), where s is the starting average piece size, t is the target average piece size, and n is the number of blades.

Each of the next n lines contains two integers m and h ($1 \leq m, h \leq 10^6$). These are the blades, where m is the maximum average piece size of the blade and h is the number of seconds the blade needs to halve the average piece size.

Output

Output a single number, which is the minimum amount of time in seconds needed to process the food to the target average piece size. If it is not possible to reach the target, output -1. Your answer should have a *relative* error of at most 10^{-5}.

Sample Input 1	Sample Output 1
10 1 2 10 10 4 5	23.219281

Sample Input 2	Sample Output 2
10000 9999 1 10000 1	1.4427671804501932E-4

This page is intentionally left blank.

Problem G
Greedy Increasing Subsequences
Time limit: 5 seconds

Jimmy's homework is to find a long increasing subsequence of a given sequence $a_1, a_2, \ldots, a_n$. But the sequence is really long! Jimmy doesn't know how to do this effectively.

So Jimmy takes a greedy approach. He begins by picking the first number in the sequence. Then he repeats the following rule until it no longer applies: pick the next number in the sequence that is bigger than the number he just picked.

More precisely, Jimmy picks the subsequence $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ where:

- $i_1 = 1$
- For each $1 \leq j < k$, i_{j+1} is the smallest index greater than i_j such that $a_{i_j} < a_{i_{j+1}}$
- $a_{i_k} \geq a_\ell$ for every $\ell > i_k$

Jimmy realizes that this may not produce a very long subsequence. So to help him find other subsequences, he removes $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ from the given sequence and finds another increasing subsequence using his greedy algorithm on the remaining sequence. He repeats this until he has used up all numbers from the original sequence.

But even this is starting to sound exhausting for Jimmy, so he asks you to help him by finding all of the sequences that would be formed by repeatedly applying the above greedy procedure and removing the resulting subsequence until the given sequence is empty.

Input

The first line of input contains a single integer n ($1 \leq n \leq 2 \times 10^5$) indicating the length of the original sequence.

The second line of input contains n integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq 10^9$).

Output

The first line of output contains the number s of sequences that are produced. The next s lines contain the sequences, the ith such line containing the increasing subsequence that is formed in the ith application of the greedy algorithm.

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

Sample Input 2	Sample Output 2
7 8 6 7 5 3 0 9	5 8 9 6 7 5 3 0

Problem H
Branch Manager
Time limit: 5 seconds

You are managing a transportation network of one-way roads between cities. People travel through the transportation network one by one in order all starting from the same city, and each person waits for the person before them to stop moving before starting. The people follow a simple algorithm until they reach their destination: they will look at all the outgoing roads from the current city, and choose the one that leads to the city with the smallest label. A person will stop when they either reach their destination, or reach a city with no outgoing roads. If at any point someone fails to reach their destination, the rest of the people still waiting in line will leave.

Before each person enters the transportation network, you can permanently close down any subset of roads to guarantee they reach their destination. The roads that you choose to close down will not be available for future people.

There are n cities, labeled from 1 to n. There are $n-1$ directed roads, and each road will always be from a lower labeled city to a higher labeled one. The network will form a rooted tree with city 1 as the root. There are m people that want to travel through the network. Each person starts from city 1, and has a specific destination city d in mind. These people will line up in the given order. What is the maximum number of people you can route correctly to their destination if you close roads optimally?

Input

The first line of input contains two integers n and m ($2 \leq n, m \leq 2 \times 10^5$), where n is the number of cities and m is the number of people.

Each of the next $n-1$ lines contains two integers a and b ($1 \leq a < b \leq n$), denoting a directed road from city a to b. These roads will describe a rooted tree with city 1 as the root.

Each of the next m lines contains a single integer d ($2 \leq d \leq n$), denoting the destination city of the next person in line.

Output

Output a single integer, which is the maximum number of people you can route to the correct destination.

Sample Input 1

```
8 5
1 2
4 8
4 6
1 4
2 5
4 7
2 3
5
2
6
4
8
```

Sample Output 1

```
5
```

Sample Input 2

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

Sample Output 2

```
1
```

Problem I
I Could Have Won
Time limit: 1 second

"We will be closing in about 5 minutes. Thank you for visiting the ICPC gym today."

With this announcement, Alice and Bob stopped playing their rock-paper-scissors marathon in the middle of the 10th game. Each player scores a point if their throw beats the other player's throw. Each game was played by the *first-to-11* rule, meaning that whoever scores 11 points first wins the game. Today, Bob narrowly defeated Alice by a single game; he scored 11 points first in five games, while Alice only scored 11 points first in four games.

After carefully inspecting how each game was played, however, Alice realized that she could have won more games than Bob if they played under slightly different rules, such as *first-to-5* or *first-to-8*, instead of the regular *first-to-11*.

Given the sequence of points scored by Alice and Bob, determine all values of k such that Alice would have won more games than Bob under the *first-to-k* rule.

Both Alice and Bob start with zero points at the beginning of a game. As soon as one player reaches k points, that player wins the game, and a new game starts. Alice wins a game if she scores k points before Bob does. Neither player wins the game if it's interrupted by the gym closing before either player reaches k points.

Input

The single line of input consists of a string of uppercase letters "A" or "B", denoting who scored each point from the beginning of the rock-paper-scissors marathon. The length of the string is between 1 and 2 000 letters, inclusive. "A" means Alice scored the point, "B" means Bob scored the point.

Output

On the first line, output the number of positive integers k for which a *first-to-k* rule would have made Alice win more games than Bob. If this number isn't zero, on the next line output all such values of k in increasing order, separated by spaces.

Sample Input 1	Sample Output 1
BBAAABABBAAABB	3 3 6 7

Sample Input 2	Sample Output 2
AABBBAAB	2 2 4

This page is intentionally left blank.

Problem J
Sun and Moon
Time limit: 1 second

You recently missed an eclipse and are waiting for the next one! To see any eclipse from your home, the sun and the moon must be in alignment at specific positions. You know how many years ago the sun was in the right position, and how many years it takes for it to get back to that position. You know the same for the moon. When will you see the next eclipse?

Input

The input consists of two lines.

The first line contains two integers, d_s and y_s $(0 \leq d_s < y_s \leq 50)$, where d_s is how many years ago the sun was in the right position, and y_s is how many years it takes for the sun to be back in that position.

The second line contains two integers, d_m and y_m $(0 \leq d_m < y_m \leq 50)$, where d_m is how many years ago the moon was in the right position, and y_m is how many years it takes for the moon to be back in that position.

Output

Output a single integer, the number of years until the next eclipse. The data will be set in such a way that there is not an eclipse happening right now and there will be an eclipse within the next $5\,000$ years.

Sample Input 1	Sample Output 1
3 10 1 2	7

This page is intentionally left blank.

Problem K
Everything Is A Nail
Time limit: 5 seconds

As an employee of the questionable Iffy immense Colossal Pinnacle Construction (ICPC) company building a very tall skyscraper, you have a number of tasks to complete high above the ground in a specific order. You can always choose to skip a task, but you fear that doing so too many times might cause some catastrophic failure of the building. You cannot revisit or complete a task once it has been skipped.

Each task is a nail, a screw, or a bolt. You have three tools: a hammer (works on nails), a screwdriver (works on screws), and a wrench (works on bolts). When you start a new task you can choose to switch your tool out by dropping it (hopefully no one was below you at the time), but when you do so you permanently lose the dropped tool.

Given the list of tasks in the order they should be completed, determine the maximum number of tasks that can be completed. You may choose to use any tool as the initial tool.

Input

The first line of input contains an integer n ($1 \leq n \leq 3 \times 10^5$), which is the number of tasks you need to complete.

Each of the next n lines contains a single integer t ($0 \leq t \leq 2$). These are the tasks, in order. Each task is one of 0 (nail), 1 (screw), or 2 (bolt).

Output

Output a single integer, which is the maximum number of tasks that can be completed.

Sample Input 1	Sample Output 1
10 1 1 1 0 0 0 0 2 2 2	10

Sample Input 2	Sample Output 2
10 0 1 2 0 1 2 0 1 2 0	5

Problem L
Family Visits
Time limit: 2 seconds

You are a college student living on your own. However, your doting family still likes to visit you, and they often stop by to check on your room at night before going to dinner. Your family will be worried if they find a mess in your room. Therefore you make an effort to ensure that they never see a mess in your room when visiting at night. You have some free time each afternoon that allows you to clean up, but the amount of free time varies each day due to prior commitments.

Luckily, your schedule is planned out well. You know exactly how big of a mess you will make each morning, how much mess you can clean each afternoon, and on which nights your family will stop by. Since you are lazy, you want to spend as few afternoons as possible cleaning such that your family will always see a room without any mess. You may assume that your room starts completely clean, and any mess that is not cleaned remains until it is cleaned.

Input

The first line of input contains two integers, n and d ($1 \leq d \leq n \leq 1\,000$), where n is the number of days in your schedule and d is the number of days your family will visit.

Each of the next n lines contains two integers m and c ($0 \leq m, c \leq 1\,000$). For each day, in order, m is the amount of mess you make in the morning, and c is the amount you can clean in the afternoon.

Each of the next d lines contains a single integer v ($1 \leq v \leq n$). These are the days on which your family will visit, and they are listed in strictly increasing order.

Output

Output the smallest number of afternoons you have to spend cleaning to ensure your family will never see a mess. If it is not possible, output -1.

Sample Input 1	Sample Output 1
6 2 1 2 2 1 1 4 3 2 3 6 2 3 3 6	3

Sample Input 2	Sample Output 2
10 5 12 10 0 2 7 1 1 8 3 4 3 4 2 3 1 2 10 1 7 5 2 4 5 6 8	7