KTH Challenge

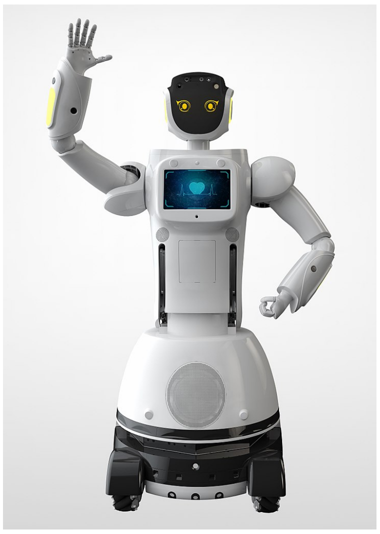
KTH Challenge 2020

April 25, 2020

Problems

Do not open before the contest has started.

Problem A
AI Jeopardy


Picture by QIHAN Technology

The robot revolution is finally here, albeit not quite in the highly explosive way envisioned in various science fiction books and movies. It seems that, perhaps due to a small typo in the AI source code, the robots are not taking our lives but instead taking our livelihoods. One early job market fatality in this revolution was the (somewhat niche) occupation as jeopardy player: already in 2011 the *Watson* computer defeated two legendary but inferior human jeopardy champions.

Nowadays, more and more of Jeopardy's *viewers* are AIs themselves and as such the show is considering having categories on topics that are more popular with this new number-crunching viewer base. Focus group testing has revealed that AIs are particularly fond of the "Binomial Coefficients" category. The premise of this category is that the answer that contestants get is some positive integer X, and the contestants must respond with a question of the form "What is n choose k?" (and this is a correct response if the binomial coefficient n choose k equals X).

Write an AI to play this new Jeopardy category. If there are several different possible solutions for n and k, the AI should choose the most elegant solution, having the smallest value of n, and of those with the smallest n it should choose the one with the smallest value of k.

Input

Input consists of a single integer X ($1 \leq X \leq 10^{100}$).

Output

Output two non-negative integers n and k such that the binomial coefficient n choose k equals X, with ties between multiple solutions broken as explained above.

Sample Input 1	Sample Output 1
`10`	`5 2`

Sample Input 2	Sample Output 2
`2020`	`2020 1`

Sample Input 3	Sample Output 3
`1`	`0 0`

This page is intentionally left (almost) blank.

Problem B
Bling

Picture by Francis Chung, cc by

Trapped at home in quarantine, Johan tries to keep madness at bay and fend off isolation by playing Critter Junction, a social simulation video game. One of the main aspects of the game is collecting and harvesting various types of resources, in order to gain Bling, the currency of the game. Johan specializes in running the fruit% category of the game, in which the goal is to obtain the maximum amount of Bling in 40 days using only fruits and no other resources.

Each fruit can be sold for 100 Bling, or planted to become a fruit tree (but not both). Every three days, starting on the third day after it was planted, a fruit tree yields three new fruits.

There are also some more exotic fruits that can be bought from the neighboring village. Once per day, the player can travel to the neighboring village and pay 400 Bling to buy a single exotic fruit which you can then plant or sell already on the same day. Analogously to normal fruits, these exotic fruits can be planted into exotic fruit trees which yield three exotic fruits every three days. Each exotic fruit can be sold for 500 Bling.

Any number of fruits/exotic fruits can be harvested, sold and planted during a day (subject to availability of course, e.g. it is not possible to sell more fruits than you actually have), but at most a single exotic fruit can be bought. These activities can be done in any order, so it is for instance possible within the same day to first harvest a few fruits (exotic or not), then sell those fruits for Bling, then use that Bling to buy an exotic fruit, and then either plant or sell that exotic fruit.

Given the current state of Johan's fruit farm, what is the maximum amount of Bling he can achieve in the remaining time?

Input

The input consists of a single line containing six integers d, b, f, t_0, t_1 and t_2 ($1 \leq d \leq 40$, $0 \leq b \leq 500$, and $0 \leq f, t_0, t_1, t_2 \leq 100$), where:

- d is the number of remaining days Johan has
- b is the current amount of Bling Johan has.
- f is the current number of fruits Johan has.
- t_i is the number of fruit trees Johan has that will yield crop i days from today (for $0 \leq i \leq 2$).

Johan currently does not have any exotic fruits or exotic fruit trees.

Output

Output a single integer: the maximum amount of Bling Johan can have after playing d days.

Sample Input 1

```
4 0 1 0 0 0
```

Sample Output 1

```
300
```

Sample Input 2

```
5 0 1 0 1 0
```

Sample Output 2

```
1900
```

Sample Input 3

```
6 0 1 1 0 0
```

Sample Output 3

```
2300
```

Sample Input 4

```
10 399 0 0 0 0
```

Sample Output 4

```
399
```

Sample Input 5

```
1 400 0 0 0 0
```

Sample Output 5

```
500
```

Problem C
Friendly Fire

Public domain

You have been captured by an evil organization which seeks to exploit your algorithm skills to improve their weapons. The first task they forced you to work on was to program guided torpedoes. Naturally, your goal is to instead make the torpedoes miss every ship whenever possible.

A typical scenario where the torpedoes will be used is a 2D plane, where the torpedo is fired from $(0, 0)$ towards m ships in the positive y-direction. Every ship has the form of a line segment *parallel to the x-axis*, with integer coordinate endpoints. The torpedo is shaped like a single point, and every second it will travel from (x, y) to either $(x - 1, y + 1)$, $(x, y + 1)$, or $(x + 1, y + 1)$. In other words, it will always go one unit forward, but your program decides how much sideways it should go. If the torpedo hits one of the ships (including one of the endpoints of a ship) it will explode, destroying the ship. On the other hand, if the torpedo stays in one piece for n seconds, its fuel will run out and it will harmlessly sink to the ocean floor.

Write a program which, given the position of the m ships, finds out whether it is possible to avoid them all, and if so, outputs instructions to do it.

Input

The first line of input contains two integers n and m ($2 \leq n \leq 500\,000$ and $0 \leq m \leq 200\,000$), the number of seconds until the torpedo runs out of fuel, and the number of ships.

Then follow m lines, each containing three integers x_1, x_2, y ($-n \leq x_1 \leq x_2 \leq n$ and $1 \leq y < n$), indicating a ship with endpoints (x_1, y) and (x_2, y).

You may assume that no pair of ships touch each other.

Output

If it is possible to dodge all the ships, output a string of length n containing the characters $-$, 0, and $+$. This string represents how the torpedo should turn in each of the n time steps. For example, if the first character is $+$, then the torpedo will start by going from $(0, 0)$ to $(1, 1)$. If there are multiple solutions, output any one of them. If it is impossible to avoid all the ships, output "`impossible`".

Sample Input 1	Sample Output 1

5 6
-3 -2 3
-2 -2 4
2 3 3
-1 1 2
0 1 4
2 5 1
	--+0-

Sample Input 2

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

Sample Output 2

```
0+-
```

Sample Input 3

```
3 2
1 2 1
-2 1 2
```

Sample Output 3

```
impossible
```

Problem D
Gaggle

Picture by Antranias

At the new start-up company Gaggle, we have rejected the oppressive corporate structures of old, with all of their managers and subordinates and hierarchies and so on. Instead we have embraced a free and open corporate culture in which all employees (called Gagglers) are in charge of themselves and allowed to roam free.

Rather than having managers overseeing the work, the main method used to coordinate work at Gaggle is a mentor system: each Gaggler designates some other Gaggler as their mentor, with whom they discuss their ongoing projects. This mentor relation may or may not be symmetric (in other words you may or may not be the mentor of your mentor) but you can never be the mentor of yourself.

Initially, all Gagglers were able to pick anyone they liked as their mentor, but after a while it was discovered that this lead to two problems:

1. Some people were more popular than others and had too many choosing them as their mentor, causing them not to have time to do their actual work.
2. Some flocks of Gagglers ended up isolated from the rest of the company (e.g., if Gagglers A and B are each other's mentors and they are not the mentor of anyone else), causing failure of these flocks to coordinate with the rest of the company.

In order to remedy these two flaws, it was (collectively) decided that:

1. Every Gaggler must be the mentor of exactly one other Gaggler, and
2. Assuming every Gaggler only communicates with their mentor and their mentee, it must still be possible for any information that any Gaggler has to reach any other Gaggler.

In order to reward lower-numbered (more senior) Gagglers while introducing this new policy, it was decided that lower-numbered Gagglers should get to keep their current mentor if possible, and if they have to change, their new mentor should be as low-numbered (more senior, and therefore more experienced) as possible.

Concretely, consider two possible new assignments of mentors, and suppose the lowest-numbered Gaggler where these assignments differ is Gaggler number i. Then if one of the two assignments assigns Gaggler i the same mentor as they originally had, we prefer that assignment. Otherwise, if Gaggler i gets a new mentor in both of the two assignments, then we prefer the assignment where the number of the new mentor of Gaggler i is smaller.

For example, consider Sample Input 2 below. One possible new assignment of mentors would be to simply change so that Gaggler 1 becomes mentored by Gaggler 2. However, in the best assignment, shown in Sample Output 2, we let Gaggler 1 keep their current mentor and instead change the mentors of both Gagglers 2 and 3.

Input

The first line of input contains a single integer n ($2 \leq n \leq 500\,000$), the number of Gagglers. Then follows a line containing n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq n$ and $a_i \neq i$ for each i) where a_i is the current mentor of Gaggler i (the Gagglers are numbered from 1 to n).

Output

Then output a line with the new assignment $b_1, \ldots, b_n$ of mentors, in the same format as in the input. The new list should be a valid assignment according to the new requirements, and be the best according to the tie-breaking rule described above.

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

Sample Input 2	Sample Output 2
3 3 3 1	3 1 2

Problem E
Pitch Performance

After a recent disaster at the Easter party karaoke, you are working on improving your singing. To gauge how well you are doing, you would like to measure how much the pitch and timing of your singing differs from the target melody you were trying to perform.

We model the melody in a simplified manner as a piecewise-constant function f, where at time x the melody has pitch $f(x)$. In other words from time 0 up to some time x_1, $f(x)$ is some constant value y_1, and then at time x_1 it changes to some other value y_2 and remains at that value until some time $x_2 > x_1$, and so on.

Your voice, on the other hand, is of a more wavering nature, and you may generally not be able to hold an exact constant pitch for any period of time, sometimes breaking off into an unwelcome falsetto and sometimes croaking on those low tones. The pitch of your voice can be modeled in a highly simplistic way as a piecewise-quadratic function g. In other words from time 0 up to x_1 (not necessarily the same x_1 as for the function f), your pitch $g(x)$ agrees with some quadratic polynomial, and then from time x_1 to x_2 with some other quadratic polynomial, and so on.

The difference between your performance g and the target melody f is the area between these two functions. See Figure E.1 for an example. Given the melody f and your performance g, compute their difference.

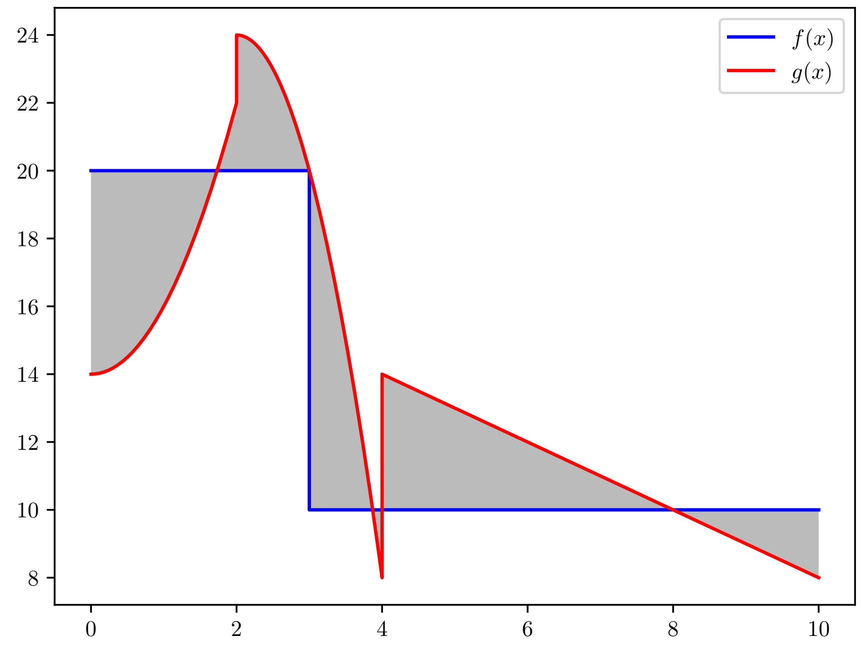


Figure E.1: Illustration of Sample Input 1. The difference between f and g is the area of the shaded region in the figure.

Input

The first line of input contains an integer n ($1 \leq n \leq 500$), the number of pieces in the target melody function f. Then follow n lines describing f. The i'th such line contains two integers x_i and y_i ($x_{i-1} < x_i \leq 10^4$ and $0 \leq y_i \leq 10^4$). For all x in the half-open interval $[x_{i-1}, x_i)$, the value of $f(x)$ equals y_i. For the first interval we define $x_0 = 0$.

Then follows a line containing an integer m ($1 \leq m \leq 500$), the number of pieces in the function g describing your performance. The next m lines contain the description of g. The i'th such line contains four integers x'_i, a_i, b_i and c_i ($x'_{i-1} < x'_i \leq 10^4$ and $-10^7 \leq a_i, b_i, c_i \leq 10^7$). For all x in the half-open interval $[x'_{i-1}, x'_i)$, the value of $g(x)$ equals $a_i x^2 + b_i x + c_i$. For the first interval we define $x'_0 = 0$.

You may assume that $0 \leq g(x) \leq 10^4$ for all $x'_0 \leq x \leq x'_m$ and that the two functions end at the same time (i.e., $x_n = x'_m$).

Output

Output the difference between f and g. Your output should be correct to within an absolute or relative error of at most 10^{-6}.

Sample Input 1

```
2
3 20
10 10
3
2 2 0 14
4 -4 16 8
10 0 -1 18
```

Sample Output 1

```
24.7785420704411
```

Sample Input 2

```
1
20 50
1
20 1 -20 100
```

Sample Output 2

```
609.47570824873
```

Problem F
Proofs

You are teaching discrete math. You have done your best to teach your students about axioms and inference rules, proofs and theorems. Sometimes the students write beautiful proofs that Fermat would be proud of but sometimes, also like Fermat, their proofs are not quite right. You are getting a little tired of hunting through some of these so-called "proofs" for the magic tricks that let them prove $1 = 2$ and had the great idea to write a computer program to speed things up!

Because this is the first class in proof-based mathematics, you have started your students off with a simple proof system. All proof lines consist of a list of assumptions, an arrow, and a conclusion. If there are no assumptions, the conclusion is an axiom. A line of the proof is valid if and only if all assumptions were conclusions of previous lines. Sometimes the students derive a conclusion more than once just to be extra sure it is true, and that is perfectly all right!

Input

The first line of input consists of an integer $1 \leq n \leq 400\,000$, the number of lines in the "proof". Then follow the n lines of the "proof". Each line has $0 \leq a \leq 5$ assumptions, followed by an arrow (the string "->"), followed by one conclusion. All assumptions and conclusions consist of $1 \leq c \leq 5$ uppercase alphabetic characters. The assumptions, arrow, and conclusion are all separated by single spaces.

Output

If every line is correct output "`correct`". Otherwise, output the number of the first line with an error (line numbers start at 1).

Sample Input 1

```
3
-> ALICE
-> BOB
ALICE BOB -> CARL
```

Sample Output 1

```
correct
```

Sample Input 2

```
1
A -> B
```

Sample Output 2

```
1
```

This page is intentionally left (almost) blank.

Problem G
Triple Texting

Julia enjoys talking to her grandma, playing with legos, and inventing two-player card games where she has a winning strategy. Recently however, she has not been able to talk to her grandma in person because of some kind of “pandemonium”. Instead, they have resorted to texting, which is a very slow process since grandma types very slowly and often mistypes letters. To make matters worse, grandma has started to write every word three times so that Julia can correct her mistypes. For example, if grandma wants to write the word “hello”, she will instead write “hellohellohello”. If she mistypes one of those letters, it might instead be sent as “hellohrllohello’.’

Your task is to write a program that given a message sent by grandma, where possibly one letter has been changed to some other letter, finds the original word.

Input

The input consists of one string s containing lower case English letters ($3 \leq |s| \leq 99$). This is the message sent by grandma. It is guaranteed that this string is the result of a word being written three times, where possibly one letter was changed to some other letter.

Output

Output one string t, the original word.

Sample Input 1	Sample Output 1
`hellohrllohello`	`hello`

Sample Input 2	Sample Output 2
`hejhejhej`	`hej`

This page is intentionally left (almost) blank.

Problem H
Winning the Vote

In the country of Elecuador, a very strange voting system is used. When it is time for the election, each one of the n citizens will arrive in some order to the voting station. There are only two parties to vote for, conveniently named 1 and 2. When arriving to the voting station, a person will vote for one of the parties, unless they are a *teller*. The tellers do not vote, instead they count how many votes each of the two parties has at the time the teller arrives, and if one of the parties has more votes than the other then that party receives one point (if the two parties have the same number of votes, neither of them receives a point). The party with the most points at the end wins. If both parties end up with the same number of points, chaos ensues.

As the president of Elecuador representing party 1, you are worried that the coming election will be the end of your reign. Fortunately, you have a plan to stop this from happening. Being the president, you know who everyone in the country will vote for, who the tellers are, and in what order everyone will arrive to the voting station. By making the right phone calls, you can also affect when the tellers arrive. In one move, it is possible to swap a teller with an adjacent person in the list of arrivals to the voting station. Note that it is not possible to swap two adjacent non-tellers. What is the minimum number of swaps necessary to ensure that party 1 wins?

Input

The input starts with a line containing an integer n n ($1 \leq n \leq 5\,000$), the number of citizens in Elecuador. Then follows a line containing a string s of length n, consisting of the characters 0, 1, and 2. This string represents the citizens in the order they arrive to the voting station. If the i'th character s_i is 1 or 2, it means that the i'th citizen will vote for party 1 or 2, respectively. If s_i is 0, it means that the i'th citizen is a teller.

Output

If it is possible to ensure victory, output one integer, the minimum number of swaps necessary. Otherwise, output "`impossible`".

Sample Input 1	Sample Output 1

`8`
`12210020`
	`4`

Sample Input 2	Sample Output 2

`4`
`1111`
	`impossible`

Sample Input 3	Sample Output 3

`11`
`00211222220`
	`5`

This page is intentionally left (almost) blank.