Problem A
Beast Bullies
Time Limit: 2 Second(s)

There are some animals with distinct strength levels getting water from a watering hole. The more animals that are at the watering hole the less water each animal will get. For this reason some of the animals will try to force the others to leave.

Periodically the weakest animal in the group may be attacked by some of the stronger animals. Other animals may come to the aid of the weakest animal. The weakest animal will leave if the sum of the strength levels of the attackers is strictly greater than the sum of the strength levels of the defenders. The other defenders, if any, will remain. This may repeat any number of times.

As an up-and-coming zoologist, you are interested in determining if animals are logical thinkers. You have a lot of data, so you will write a program to help check if, in each situation, the animals behaved optimally. Each animal wants to eliminate as many other animals as possible without being eliminated itself.

Given the strength level of each animal and assuming each animal always makes the best decision for their best interests, determine how many animals are guaranteed to remain after all aggression has been resolved.

Input

The first line of input contains a single integer n ($1 \leq n \leq 5 \times 10^5$), which is the number of animals.

Each of the next n lines contains a single integer s ($1 \leq s \leq 10^9$). These are the strengths of the animals. No two strengths will be the same.

Output

Output a single integer, which is the number of animals guaranteed to remain.

Sample Input 1	Sample Output 1
4 3 4 8 9	3

Sample Input 2	Sample Output 2
8 20 1 2 13 9 5 8 61	1

Problem B
Birthday Gift
Time Limit: 2 Second(s)

It's your Venusian friend's birthday. You don't remember their exact age, but you are sure it had to be no more than 10^{18} years. You will give them a decimal number (without leading zeros) for their birthday. You want the number of digits to be equal to their age. To make the number more interesting you will ensure that no adjacent pairs of digits will be identical.

Their exact day of birth is represented as an integer in the range 0 to 224 (since Venus has 225 days in a year). To make their gift more personal you want the given number to have the same remainder as their birthday when divided by 225.

There are potentially a lot of possible gifts that you could give. You may decide to give more than one gift. Determine the number of possible gifts modulo $10^9 + 7$.

Input

The single line of input contains two space separated integers a ($1 \leq a \leq 10^{18}$) and b ($0 \leq b < 225$), where a is the age of your friend and b is the birthdate of your friend.

Output

Output a single integer, which is the number of interesting personalized numbers you could give. Since this number may be quite large, output it modulo $10^9 + 7$.

Sample Input 1	Sample Output 1
`12345 200`	`323756255`

Sample Input 2	Sample Output 2
`100 87`	`896364174`

Sample Input 3	Sample Output 3
`100 35`	`785970618`

Sample Input 4	Sample Output 4
`5000 5`	`176058968`

Sample Input 5	Sample Output 5
`888888 88`	`906317283`

Sample Input 6	Sample Output 6
`9999999 99`	`133442170`

Sample Input 7	Sample Output 7
`101010101010 127`	`893501348`

Problem C
Class Field Trip
Time Limit: 1 Second(s)

Instructors ann and ben firmly believe that everything should be alphabetical, especially names. (They also don't believe in capital letters!) They take alphabetical order to such an extreme that they even order the letters of their names when listed together. For example, the last time they co-taught they listed the instructors as `abennn`! They are taking their classes on a class field trip, so they need to merge their lists of students. Fortunately, they each have their student lists in order, so merging the lists should be quick. If ann has `john` and `amy` in her class and ben has `jack` and `jill` in his class, their two lists would be `ahjmnoy` and `acijjkll`. The field trip list would be `aachijjjkllmnoy`.

Given ann's and ben's lists, produce the field trip list.

Input

Input consists of exactly two lines. Each line contains a string of from 1 to 100 lower case letters. These are ann's and ben's lists.

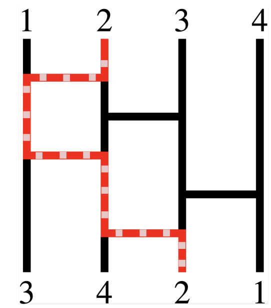
Output

Output a single string of lower case letters, which represents the merging of ann's and ben's lists.

Sample Input 1	Sample Output 1
`ahjmnoy` `acijjkll`	`aachijjjkllmnoy`

This page is intentionally left blank.

Problem D
Ghost Leg
Time Limit: 1 Second(s)



A *Ghost Leg* is a method for representing permutations.

A Ghost Leg can be represented as a set of vertical lines, each line corresponding to one element of the set being permuted. Horizontal lines (like the rungs of a ladder) connect adjacent vertical lines in such a way that no two rungs share an endpoint. An example is shown in the illustration to the right.

To determine the resulting position of any element under a Ghost Leg permutation, begin at the top of the vertical line corresponding to that element. Trace a path down the line until encountering the first rung touching that line. Follow that rung to the adjacent vertical line. It does not matter whether the rung goes to the left or to the right, just follow it. Continue downward, following rungs, until reaching the bottom. This gives the final position of that element under the permutation.

In the example, element 2 ends up third in the permutation. The path is shown with red dotted lines. Likewise, element 1 ends up fourth, element 3 ends up first, and element 4 ends up second.

Given a description of a Ghost Leg, determine the result of the permutation. In other words, apply the specified permutation, and output the elements in their permuted order.

Input

The first line of input contains two positive integers n ($1 \le n \le 100$) and m ($0 \le m \le 1{,}000$), where n is the number of elements being permuted, and m is the number of rungs. The vertical lines (elements) are identified from left to right by the integers 1 through n.

Each of the next m lines contains a single integer a ($1 \le a < n$), which indicates that there is a rung between vertical lines a and $a + 1$. The rungs are listed in order from top to bottom of the Ghost Leg. Clearly, no two rungs can be at the same height.

Output

Output n lines, where each line contains a single integer. This is the resulting permutation. The first line is the element that ends up first in the permutation, the second is the element that ends up second, and so on.

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

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

Problem E
MazeMan
Time Limit: 1 Second(s)

You now work for a video game company - every programmer's dream! You are working on a multiplayer game where players cooperate to enter a maze and try to consume all of the "dots" as quickly as possible. Each player enters the maze at a different entrance. The mazes are randomly generated, so the minimum number of players needed to consume all of the dots can vary, and some dots may not be reachable at all.

You are working in the Quality Control department, analyzing the randomly generated mazes. For analysis, the mazes are represented in text. An `X` is a wall that cannot be crossed. Letters `A`-`W` are entrances. Players can only move up, down, left and right. Players can move though spaces and dots; moving over a dot eats it. If two doors are adjacent, players cannot move from one to the other. For example:

```
XXXXXXXAXXXXXXXBXXXX
X.. ..X.X...... ...X
X.XXX...X.X.XXXXXX.X
X.X.XXXXX.X.X....X.X
X.X... ...X.X.XX.X.X
X.X.X.XXXXXXX.XX.X.X
X.X.X.X...X...X....X
X.X.X.XXXXXXX.XXXX.X
X...X.X X.. ..X..X.X
XXXXXXXDXXXXXXXXCXXX
```

All of the reachable dots can be reached from entrances `A` and `C` (or, equivalently, `B` and `C`). There are three dots that cannot be reached.

Calculate the minimum number of players necessary to eat all the reachable dots, and how many dots are not reachable because they are walled off.

Input

This first line of input contains two integers n and m ($3 \leq n, m \leq 100$), where n is the number of rows in the maze representation, and m is the number of columns.

Each of the next n lines contains a string of length exactly m, consisting only of the capital letters `A` through `X`, space, or period. This is the maze. The borders of the maze (rows 1 and n, columns 1 and m) are guaranteed to consist only of capital letters `A` through `X`. There are no entrances (`A`-`W`) in the middle of the maze.

Output

Output a line with two space-separated integers, the first of which is the minimum number of entrances necessary to enter in order to eat all of the dots (which may be 0 if no dots are reachable), and the second of which is the number of dots which cannot be reached.

Sample Input 1

```
10 20
XXXXXXXAXXXXXXXBXXXX
X.. ..X.X...... ...X
X.XXX...X.X.XXXXXX.X
X.X.XXXXX.X.X....X.X
X.X... ...X.X.XX.X.X
X.X.X.XXXXXXX.XX.X.X
X.X.X.X...X...X....X
X.X.X.XXXXXXX.XXXX.X
X...X.X X.. ..X..X.X
XXXXXXXDXXXXXXXXCXXX
```

Sample Output 1

```
2 3
```

Sample Input 2

```
3 5
XDRVX
X.X.X
XXXXX
```

Sample Output 2

```
2 0
```

Sample Input 3

```
3 5
NAQXX
X X.X
XXXXX
```

Sample Output 3

```
0 1
```

Problem F
Metronome
Time Limit: 1 Second(s)

A *Metronome* is a mechanical device used by musicians for keeping time. It is a very clever device, based on a spring, an inverted pendulum, and an escapement gear. Milo is learning to play the glockenspiel, and has purchased a metronome to help him keep time. Milo has noticed that for every complete turn (one revolution) of the key, the metronome will give four ticks. Milo wants the metronome to stop at the end of each song that he tries to play.

For a given song, how many revolutions must he wind the key?

Input

The single line of input contains a single integer n ($1 \leq n \leq 10^5$), which is the length of the song in ticks.

Output

Output a single real number, which is the number of revolutions Milo must turn the metronome's key so that it stops precisely at the end of the song. This number must be accurate to two decimal places.

Sample Input 1	Sample Output 1
`16`	`4.0`

Sample Input 2	Sample Output 2
`99`	`24.75`

This page is intentionally left blank.

Problem G
Movie Night
Time Limit: 1 Second(s)

You're trying to organize an outing to a movie with a group of friends. This is made complicated by the fact that your friends will decide whether to come based on who else is coming. If you give one of your friends a call, you know you're going to have a conversation like this:

> **You**: "Hi Fred, want to come with us to the 7:30 showing of Coding Horror from the Accidentally Quadratic Lagoon?"
>
> **Fred**: "Well, I dunno... is Francine coming?"

In fact, for each one of your friends *X*, there is exactly one other friend *Y* such that *X* will come only if *Y* also comes. Of course, you must invite a subset of your friends such that everyone invited knows who else is invited and will be willing to come, and no one uninvited will need to come.

The question is, how many such subsets of your friends are there? You don't want to go to the movies alone, so every set must have at least one friend.

Input

The first line of input contains an integer n ($2 \leq n \leq 10^5$). This is the number of friends you might invite to the movie. You identify your friends by a number from 1 to n.

Each of the next n lines contains a single integer y ($1 \leq y \leq n$). This indicates that friend x will only go to the movie if friend y also goes to the movie, where x is 1 for the first y value, 2 for the second y value, and so on. No one will be their own friend! (i.e., $x \neq y$)

Output

Output a single integer, which is the number of distinct nonempty subsets of your friends you could invite such that everyone you invite will be willing to come. Since this number may be quite large, output it modulo $10^9 + 7$.

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

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

Problem H
Platform Placing
Time Limit: 1 Second(s)

The city of Atlantis is making an above-water section. They are doing so by building floating platforms that are anchored at their centers to foundation points that lie in a straight line at the bottom of the ocean. Each platform is a fixed-width rectangle aligned to the foundation line; specifically, a platform of length ℓ anchored to a foundation point at position x along the line occupies the interval $[x - \frac{\ell}{2}, x + \frac{\ell}{2}]$ along the line on the ocean surface. The platforms can be of different lengths, but have both a minimum and a maximum length. Gaps are allowed between consecutive platforms, and platforms are allowed to exactly touch, but they may not overlap. Each foundation point must be attached to exactly one platform.

Help the Atlanteans maximize their above-water section: given the locations of the foundation points and the minimum and maximum allowed platform lengths, determine the maximum possible sum of platform lengths.

Input

The first line of input contains three space separated integers n ($1 \leq n \leq 10^5$), s and k ($1 \leq s \leq k \leq 10^5$), where n is the number of foundation points, s is the smallest platform length possible, and k is the largest platform length possible.

Each of the next n lines contains a single integer x ($1 \leq x \leq 10^9$), representing the location of a foundation point. No two foundation points will be the same.

Output

Output a single integer, which is the maximum possible sum of platform lengths, or -1 if it isn't possible to place the platforms.

Sample Input 1	Sample Output 1
4 1 4 1 6 8 10	11

Sample Input 2

```
5 1 6
6
7
8
3
5
```

Sample Output 2

```
7
```

Sample Input 3

```
2 5 10
1
2
```

Sample Output 3

```
-1
```

Problem I
Problem Pool
Time Limit: 1 Second(s)

ICPC is having multiple regionals on the same day, and these regionals are all pulling problems from the same problem pool. To make sure regionals are distinct there is a rule that each regional problem set must have at least some number of problems different from each other regional problem set, pairwise. For example, if there are three regionals (let's call them **A**, **B** and **C**) and the number of required distinct problems is four, then **A**'s problem set must contain at least four problems that are not in **B**'s problem set. **A**'s problem set must also contain at least four problems that are not in **C**'s problem set, but they might not be the same four problems.

All regions want to put as many problems as they can in their problem set. What's the largest number of problems so that every region can have that many problems in their set, given the number of problems in the pool to pick from, the number of regionals, and the number of problems each regional problem set must have distinct from each other regional problem set? Every problem set must have at least one problem.

Input

The single line of input contains three integers n, r and d ($1 \leq n, r, d \leq 10^9$), where n is the number of problems in the pool, r is the number of regionals, and d is the number of problems for each regional problem set that must be distinct from each other regional problem set, pairwise.

Output

Output a single integer, which is the maximum number of problems that every region can have in their problem set when drawing problems from the pool, subject to the distinct problems rule. Output -1 if it isn't possible to form r problem sets subject to the rule, all of which have at least one problem.

Sample Input 1	Sample Output 1
`5 2 2`	`3`

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

Sample Input 3	Sample Output 3
2 2 2	-1

Problem J
Room Evacuation
Time Limit: 7 Second(s)

You are now the fire marshal. It is not a fun job to have. You have a layout of a room in the building as a 2D grid. There are known locations that people will occupy, there are known locations that people cannot walk into or out of, and there are known locations that are exits. You know that two or more people cannot occupy the same cell of the 2D grid at the same time. You know how quickly everyone needs to evacuate the room in seconds. Asssume that the occupants can only move in one of the four cardinal directions (i.e., North, South, East, or West), and can make one move per second. You can assume that although only one person can stand in the exit at a time, a person in the exit is safe, and of course anyone past the exit is safe.

Given the layout of the room and the desired time to evacuate, determine how many people can get out safely.

Input

The first line of input contains three integers, n, m ($1 \leq n, m \leq 20$) and t ($1 \leq t \leq 200$), where n and m are the height and width of the floor plan, and t is the time allowed to evacuate.

Each of the next n lines contains a string of length exactly m, consisting only of the characters "`P`", "`E`", "`#`", and/or "`.`", where:

- "`P`" is a person
- "`E`" is an exit
- "`#`" is a blocked area that people cannot enter or pass through
- "`.`" is an open area that people can enter and pass through

Output

Output a single integer, which is the number of occupants of the room can be safely evacuated in the allotted time.

Sample Input 1	Sample Output 1

<pre>4 5 3
.....
..P#.
..PPE
..P.E</pre>
	<pre>4</pre>

Sample Input 2	Sample Output 2

<pre>3 3 5
...
P#P
P#E</pre>
	<pre>2</pre>

Problem K
Smallest Calculated Value
Time Limit: 1 Second(s)

Given three integers and the arithmetic operators $(+, -, *, /)$, determine the smallest non-negative integer possible without changing the order of the initial integers. Order of the operators is without their normal precedence, just left-to-right. Note: integer division is fine only if the remainder is zero. For example, $9/3$ is okay, $10/3$ is not.

For example:

For 2, 3 and 5, the answer is 0 $(2 + 3 - 5)$
For 9, 9 and 9, the answer is 0 $(9 - 9 \,/\, 9)$
For 5, 7 and 3, the answer is 1 $(5 - 7 + 3)$

Given three integers, determine the smallest non-negative result that can be computed by placement of the given operators between the numbers. The operators may only be placed between numbers, not in front (they are not unary operators). An operator must be placed between each pair of numbers, numbers cannot be concatenated.

Input

The single line of input contains three integers, all in the range from 1 to $1{,}000$.

Output

Output a single integer, which is the smallest non-negative value possible applying the arithmetic operators.

Sample Input 1	Sample Output 1
2 3 5	0

Sample Input 2	Sample Output 2
9 9 9	0

Sample Input 3	Sample Output 3
5 7 3	1

Problem L
Spidey Distance
Time Limit: 1 Second(s)

Consider a 2D plane of points with integer coordinates. The *Taxi-cab Distance* between two points is a measure of the shortest path between the points where movement is restricted to only horizontal and vertical paths, similar to following the lines on traditional graph paper. Each point connects to exactly four others, and each is one unit away.

The *Spidey Distance* is a similar measure of the shortest path, but allows movement horizontally, vertically, or along diagonal paths. Horizontal and vertical points are separated by one unit, but diagonal points are 1.5 units away. Thus, each point connects to eight other points, four of which are one unit away and four of which are 1.5 units away.

Write a program to compute the fraction of the number of points reachable via a given taxi-cab distance from a point, within an area defined by a spidey distance around that point.

Input

The single line of input contains two integers t and s ($0 \leq t, s \leq 10^6$), where t is a taxi-cab distance, and s is a spidey distance.

Output

Output a fraction in the form n/d, which is the fraction of points within the given spidey distance that can also be reached within the given taxi-cab distance. Output this fraction reduced to lowest form. If $d = 1$ only output n. Do not output any spaces around the $/$.

Sample Input 1	Sample Output 1
`4 4`	`41/49`

Sample Input 2	Sample Output 2
`6 6`	`17/21`

Sample Input 3	Sample Output 3
`3 7`	`25/141`

Sample Input 4	Sample Output 4
7 3	1

Problem M
Toll Roads
Time Limit: 2 Second(s)

Your state has a number of cities, and the cities are connected by roads. Unfortunately, all of the roads are toll roads!

You now run the local chapter of AAA (American Automobile Association), and people are constantly asking you about the tolls. In particular, they've been asking about individual tolls on any single road on a path between two cities. Odd, but that's what they've been asking!

Given a description of the cities in your state and the roads that connect them, and a series of queries consisting of two separate cities, for each query determine two things:

- First, the smallest value such that there is a route between the two cities where no road has a toll greater than that value.
- Second, the number of cities reachable from your starting city using no road with a toll greater than that first value.

Input

The first line of input contains three integers n ($2 \leq n \leq 2 \times 10^5$), m ($1 \leq m \leq 2 \times 10^5$) and q ($1 \leq q \leq 2 \times 10^5$), where n is the number of cities, m is the number of roads, and q is the number of queries. The cities are each identified by a number 1 through n.

Each of the next m lines contains three integers u, v ($1 \leq u, v \leq n, u \neq v$) and t ($0 \leq t \leq 2 \times 10^5$), which represents a road between cities u and v with toll t. The roads are two-way, and the toll is the same in either direction. It is guaranteed that there is a path between any two cities, and that there is at most one road between any two cities.

Each of the next q lines contains two integers a and b ($1 \leq a, b \leq n, a \neq b$). This represents a query about a path from a to b.

Output

Output q lines. Each line is an answer to a query, in the order that they appear. Output two space-separated integers, w and k, on each line, where w is the smallest amount such that there is a route from a to b with no toll greater that w, and k is the number of cities reachable from a using no road with a toll greater than w.

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