Winter Contest 2024

January 27th

Problems

A Alphabetical Athletes *(very easy)*

B Bright Beacons

C Chess Challenge

D Devious Dates

E Euroexpress

F Football Figurines

G Genius Gamer

H Haggling over Hours

I Impossible Install

J Jog in the Fog

K Keeping Keys *(easy)*

L Lookup Table Tennis

M Montage Matrix *(easy)*

This page is intentionally left (almost) blank.

Problem A: Alphabetical Athletes

Time limit: 1 second

Anna is organizing the accommodation of the athletes of the German delegation at the Olympic Games. While going over the list of names, she suddenly realizes that some of the names have their letters sorted alphabetically. She wonders if this is simply a funny coincidence or if there are more words in the German language where this is the case.

A word is said to be sorted alphabetically if the letters occur in the same order in the word as in the alphabet, or if they appear in the reverse order as in the alphabet. For simplicity, Anna does not distinguish between upper- and lowercase letters.

Input

The input consists of:

- One line with a string s ($1 \leq |s| \leq 60$), a German word. The word only consists of letters '`a`' to '`z`' and '`A`' to '`Z`'. All letters are given in lowercase, except for the first one which is either upper- or lowercase.

Output

Output "`yes`" if the word is sorted alphabetically, and "`no`" otherwise.

Sample Input 1	Sample Output 1
`bekloppt`	`yes`

Sample Input 2	Sample Output 2
`Ente`	`no`

Sample Input 3	Sample Output 3
`Rommee`	`yes`

This page is intentionally left (almost) blank.

Problem B: Bright Beacons

Time limit: 1 second

In honour of the Olympic ethos, an archer is faced with a revered task: to light a series of bonfires leading to the Olympic Torch. The archer must establish a network of bonfires, across a grid of hills with heights between 0 and 9, such that they may light the Olympic Torch in the south-east corner of the grid. Bound by ancient tradition, the archer may only shoot in straight lines that follow the cardinal directions of north-south or east-west. You must find the smallest number of additional bonfires needed to create an unbroken chain of bonfires between the starting point and the Olympic Torch.

The rules for establishing visibility between bonfires are:

- Bonfires can be lit when they are within a direct line of sight of another bonfire that is already burning (even if archers may shoot in a parabola, assume that if they cannot see the bonfire, they cannot hit it).
- A mountain is considered to block the line of sight between two bonfires if its peak lies strictly above the straight line between the two peaks of the bonfires in question. When determining if there is a line of sight, assume that each grid cell's terrain height is a peak of zero width in the middle of the cell.
- An archer can only shoot north, west, east or south in a single shot. Shooting in a combination of these directions is not allowed due to the Olympic traditions.

For instance, consider an example with terrain heights in a line like "`124`":

- A bonfire placed on terrain height 1 can see a bonfire placed on terrain height 4 because the terrain with height 2 is not tall enough to obstruct the view.
- However, if the terrain heights were "`134`", a bonfire on height 1 would not see a bonfire on height 4 due to the peak of height 3 blocking the line of sight.
- In cases such as "`222`" or "`123`", bonfires at the first and last point can see each other since the intermediate heights are not obstructing the direct view.

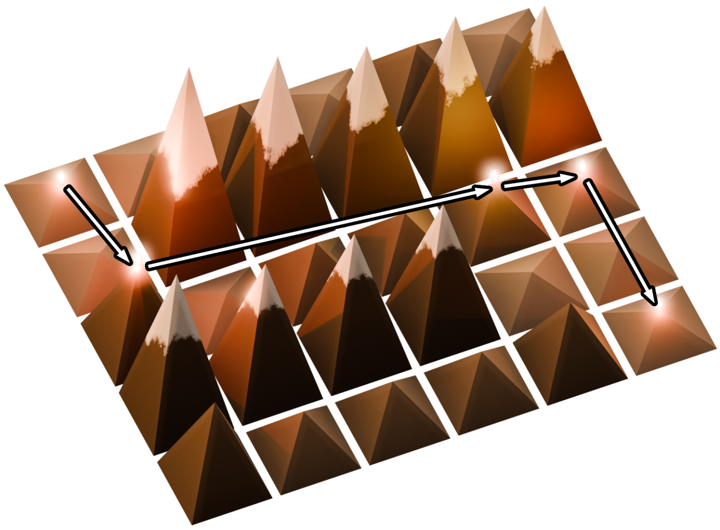


Figure B.1: Path of arrows and lit bonfires in Sample 1.

What is the minimum number of additional bonfires you need to place in order to enable signalling from the bonfire at the north-west to the Olympic Torch at the south-east? Assume that the initial bonfire and the Olympic Torch have already been placed.

Input

The input consists of:

- One line with two integers r and c ($2 \leq r, c \leq 100$), the number of rows and the number of columns in the terrain, respectively.
- Then r lines follow, each with c digits. The digits signify the height h ($0 \leq h \leq 9$) of each mountain.

Output

Output the minimum number of additional bonfires needed in order to enable signalling from the bonfire at the north-west corner to the Olympic Torch at the south-east corner.

Sample Input 1	Sample Output 1
5 6 112121 198776 412221 766611 422231	3

Sample Input 2	Sample Output 2
5 5 25688 59998 69996 89995 88653	5

Sample Input 3	Sample Output 3
2 18 011222222322222110 011222222322222110	6

Problem C: Chess Challenge

Time limit: 2 seconds

Claire loves the Olympic Games, and watches them every day when they are on. Unfortunately, this year, her absolute favourite event starts in the middle of the night. She usually likes playing chess against her brother during the breaks between different events, but since he has long gone to bed, this is not an option tonight. Getting more tired by every second, Claire struggles hard to stay awake. To not fall asleep entirely, she decides to challenge herself with some weird chess puzzles she finds online.

The chess board consists of only one row with n squares, where a square is either empty or contains a rook. Additionally, each rook is assigned a number of remaining moves. In each move, Claire may choose an arbitrary rook with more than zero remaining moves, and capture an adjacent rook with it. Two rooks are adjacent if there are no other rooks between them. The captured rook is taken out of the game, and the capturing rook has one move fewer left. The puzzle is solved if there is only one rook left in the end. Note that in each move, there must be a rook that is captured.

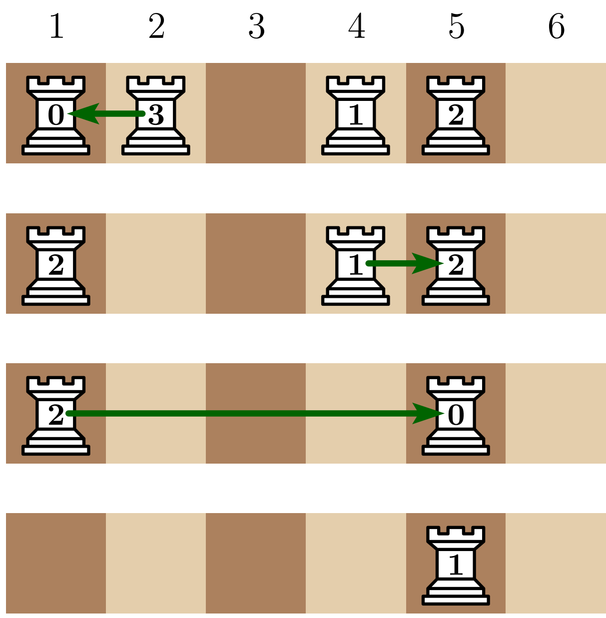


Figure C.1: Illustration of Sample 1. The puzzle is solvable with three moves. The state of the board is shown after each move of a possible solution.

Is it possible to solve the given puzzle? And if yes, which moves should Claire make?

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^5$), the size of the chess board.
- One line with n integers k ($-1 \leq k \leq 10^5$) which describe the state of each square on the board. The ith integer is -1 if the ith square is empty, otherwise, it denotes the number of remaining moves of the rook on square i.

The board contains at least one rook.

Output

Output "`impossible`" if the chess puzzle has no solution. Otherwise, output "`possible`", followed by an integer m, the number of moves of a solution. Output the solution by outputting m pairs of integers a and b ($1 \leq a, b \leq n$, $a \neq b$), meaning that the rook on square a captures the rook on square b. If there are multiple valid solutions, you may output any one of them.

Sample Input 1	Sample Output 1
6 0 3 -1 1 2 -1	possible 3 2 1 4 5 1 5

Sample Input 2	Sample Output 2
6 0 -1 0 2 1 0	impossible

Problem D: Devious Dates

Time limit: 1 second

For this year's Olympic Games, the organizers plan two spectacular joint live TV shows, each showcasing and celebrating all the disciplines. The first of these shows will air a days after Day 1 of the Games, and the second follows m days after the first.

Doreen is responsible for determining the event days of the disciplines. Notably, subsequent events of the same discipline must be the same number of days apart so that the athletes can benefit from a structured schedule. Of course, all disciplines must have an event scheduled on the days on which there is a joint TV show. The day of the second joint TV show is also the last day of the Olympic Games, so there are no event days of any discipline after it.

Naturally, Doreen wants to make the schedule simple for spectators and athletes to understand. Thus, the joint TV shows should be exactly on those days on which every discipline has an event scheduled – neither on more nor on fewer days. For the same reason, no two disciplines may have the same schedule.

Doreen has some difficulty coming up with a schedule for the disciplines that fits all these constraints and has asked you to assist her. An exemplary schedule for three disciplines is given in Figure D.1.

Day 1	—	Archery	—
Day 2	Road cycling	Archery	—
Day 3	—	Archery	—
Day 4	Joint live TV show		
Day 5	—	Archery	—
Day 6	Road cycling	Archery	—
Day 7	—	Archery	—
Day 8	Road cycling	Archery	—
Day 9	—	Archery	Football
Day 10	Road cycling	Archery	—
Day 11	—	Archery	—
Day 12	Road cycling	Archery	—
Day 13	—	Archery	—
Day 14	Joint live TV show		

Figure D.1: Illustration of Sample Output 3. Note that on Days 4 and 14 Road cycling, Archery and Football all have an event scheduled.

Input

The input consists of:

- One line with three integers a, m, k ($0 \leq a < m \leq 10^{12}$, $1 \leq k \leq 10^6$), the number of days from Day 1 of the Olympic Games to the first joint TV show, the number of days between the first and second joint TV show, and the number of disciplines.

Output

If it is impossible to schedule the disciplines as needed, output "`impossible`".

Otherwise, output "`possible`" followed by k schedules of disciplines. For each discipline, output the number of days from Day 1 of the Olympic Games to the first event day of this discipline (potentially a joint TV show), followed by the number of days between subsequent event days of this discipline.

Given the number of days between subsequent event days of a discipline, pick the earliest possible first event day. If there are multiple valid solutions, you may output any one of them.

Sample Input 1

```
7 15 5
```

Sample Output 1

```
impossible
```

Sample Input 2

```
21 105 4
```

Sample Output 2

```
possible
21 35
0 7
0 21
6 15
```

Sample Input 3

```
3 10 3
```

Sample Output 3

```
possible
1 2
0 1
3 5
```

Sample Input 4

```
7 15 1
```

Sample Output 4

```
possible
7 15
```

Sample Note: Two schedules are different if one has an event day the other schedule does not have. Therefore, in Sample 1, there are only four different possible schedules, namely "`7 15`", "`2 5`", "`1 3`" and "`0 1`". Consequently, the answer is "`impossible`".

Problem E: Euroexpress

Time limit: 2 seconds

You want to travel to the Olympic Games this year and already decided to take the Eurostar to Paris to be more environmentally friendly. Your next decision is to pick a suitcase for your journey. Upon reading the terms and conditions, you noticed that there is no clear size limit for the suitcase. Instead, they provide various two-dimensional constraints, and your suitcase is compliant if it fits in a box where each side matches one of the aforementioned constraints.

Suitcase size check. Photo by Kenzel2

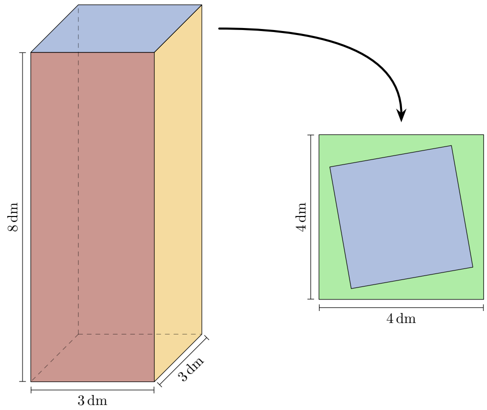


Figure E.1: Illustration of Sample 2. A suitcase with dimensions 3dm $\times$ 8dm $\times$ 3dm fits in a box where each side has either dimension 3dm $\times$ 8dm or 4dm $\times$ 4dm, i.e. complies either with constraint 3 or 1 of the input.

Since you need to buy a new suitcase anyway, you wonder how much volume could a suitcase have and still be compliant?

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 2 \cdot 10^5$), the number of constraints.
- n lines, each containing two integers a and b ($1 \leq a, b \leq 10^6$), the dimensions of the constraint in dm.

Output

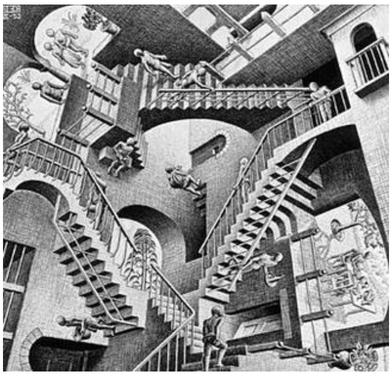
Output a single integer, the maximum volume of a suitcase that you can carry with you in dm^3.

Sample Input 1	Sample Output 1
3 2 31 5 5 13 3	125

Sample Input 2	Sample Output 2
4 4 4 2 15 8 3 20 1	72

Problem F: Football Figurines

Time limit: 2 seconds



M.C. Escher, Relativity

Frank is an avid football fan and of course attends the final game live in the Olympic football stadium. It is a huge building, and Frank is impressed by the maze of staircases connecting the n floors.

Frank notices that from each floor there are exactly two staircases going up, one to the floor directly above and one to the floor two levels up (skipping the floor directly above). On each staircase, a volunteer is distributing little figurines to the spectators, giving one figurine to everyone climbing up this staircase.

There may be many different routes between any two floors, where a route consists of a sequence of staircases. Note that, because of the mass of spectators heading to their seats, Frank can only take staircases going up. Frank is wondering how many figurines he can collect when walking up again and again but taking a different route each time. He considers two routes to be different if they differ in at least one staircase.

Given several queries, each consisting of a pair of two floors, find the maximum number of figurines Frank can collect when climbing up all the different routes between those floors.

Input

The input consists of:

- One line containing two integers n and q ($1 \leq n \leq 10^6$, $1 \leq q \leq 5 \cdot 10^4$), the number of floors and the number of queries, respectively.
- q lines, each containing two integers s and t ($1 \leq s \leq t \leq n$), the numbers of two floors.

The n floors are numbered from 1 to n.

Output

For each query, output the total number of figurines Frank can collect when climbing up all different routes between the two floors. Since this number can be very large, output it modulo $10^9 + 7$.

Sample Input 1	Sample Output 1
5 4 1 1 2 3 1 3 1 5	0 1 3 15

Sample Input 2	Sample Output 2
1000000 1 1 1000000	884826434

This page is intentionally left (almost) blank.

Problem G: Genius Gamer

Time limit: 1 second

The unique solution to Sample Input 4.

Having had a very interesting day watching this year's Road Cycling Finals, you and your friends want to enjoy the evening in the lobby of your hotel. Unfortunately, nobody brought any games to pass the time. You hope that the hotel provides some games and indeed that's the case. In a rack besides many others, you find a copy of the game Rummikub and you decide with your friends to give it a try.

This game is played with tiles where each tile has a number written in one of four colours. In the hotel's version, there are no jokers and every two tiles either differ in their colour or numerical value. Each player has a set of hidden tiles and there are tiles in a common area. The goal of each player is to place all their tiles into the common area.

In the common area, tiles have to be combined to form groups or runs. A group is composed of at least 3 tiles that all have the same numerical value, but different colours. A run is a combination of at least 3 tiles of the same colour, that have consecutive numerical values. Note that in your version of the game, 1 does not follow 13. So the tiles 12, 13 and 1 in red would not form a valid run.

In each turn players can place their hidden tiles into the common area and combine them with existing groups and runs. To make it more interesting, they can even rearrange the common area freely – without removing any tiles of course. The only restriction is that after the move, the common area should be organized into groups and runs again.

It is your turn now. You ask yourself whether you can place all your hidden tiles into the common area this turn. Write a computer program to help you.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 52$), the number of tiles at your disposal.
- n lines, each containing a string c and an integer v ($c \in \{$"`Red`", "`Yellow`", "`Blue`", "`Black`"$\}$, $1 \leq v \leq 13$), indicating that you have the tile with value v and colour c at your disposal.

No combination of colour and value appears more than once in the input.

Output

Output "`possible`" if you can place all your tiles into the common area in this round and "`impossible`" otherwise.

Sample Input 1	Sample Output 1

`3`
`Red 1`
`Black 1`
`Yellow 1`
	`possible`

Sample Input 2

```
8
Red 3
Blue 3
Red 1
Red 2
Black 3
Red 4
Red 5
Yellow 3
```

Sample Output 2

```
possible
```

Sample Input 3

```
7
Red 3
Blue 3
Red 1
Red 2
Black 3
Red 4
Red 5
```

Sample Output 3

```
impossible
```

Sample Input 4

```
21
Red 1
Red 2
Red 3
Red 4
Red 8
Blue 3
Blue 4
Blue 5
Blue 7
Yellow 3
Yellow 4
Yellow 5
Yellow 6
Yellow 7
Yellow 8
Black 3
Black 4
Black 5
Black 6
Black 7
Black 8
```

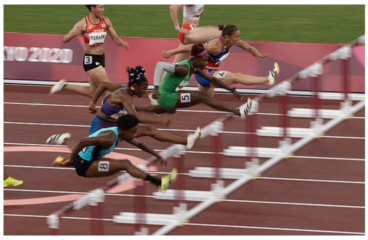
Sample Output 4

```
possible
```

Sample Note: The unique solution to Sample Input 4 is depicted in the image of the story.

Problem H: Haggling over Hours

Time limit: 5 seconds



100m hurdles at the 2020 Summer Olympics,
By Bob Ramsak on Wikimedia Commons

Hannah is responsible for organizing the time slots at which the Athletics competitions at the Olympic Games are held. She has finally found one that adheres to the constraints imposed by the Olympic committee and the other sports. Regrettably, someone else is responsible for assigning athletes to the time slots. After talking to a few athletes, she realizes that a bad assignment of athletes to time slots could be unfortunate for some of them.

Therefore, she wants to reduce the maximum number of time slots an athlete could possibly be assigned to by at least one. She wants to do this by removing some of the time slots as this is the easiest way to adjust her plan. Of course, she wants to remove as few time slots as possible. Note that an athlete can only be assigned to several time slots if there is a break of at least one hour in between consecutive ones.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^3$), the number of time slots.
- n lines, each containing two integers a and b ($0 \leq a < b \leq 10^9$), the start and end time of a time slot in hours.

It is guaranteed that no two time slots are identical.

Output

Output the minimum number of time slots which have to be removed such that the maximum number of time slots an athlete could be assigned to decreases by at least one.

Sample Input 1	Sample Output 1
5 0 2 1 4 3 6 5 8 7 9	1

Sample Input 2	Sample Output 2
6 0 2 1 3 4 6 5 7 8 10 9 11	2

Sample Input 3	Sample Output 3
5 0 2 3 6 1 2 1 4 5 6	2

Problem I: Impossible Install

Time limit: 4 seconds

Irina was hired as an unpaid intern to handle IT at the Olympic Games. Fortunately, there are a bunch of Python projects that already handle everything. They just have to be installed.

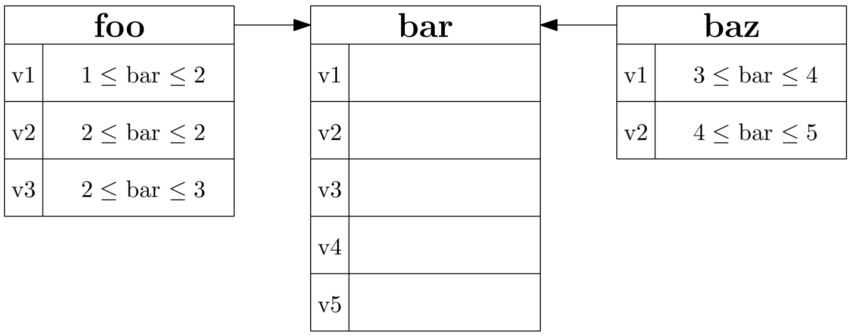


Figure I.1: Illustration of the Sample Input 1. The projects are called foo, bar, and baz here.

Garry, the intern who was tasked with the setup before, quit unexpectedly but left some notes. “What is going on?”, “This is insane!” and “QUIT WHILE YOU CAN!!!”. However, some older notes seem to be more helpful and give an idea about the setup. There are software projects and dependencies between them that form a directed acyclic graph without multi-edges. Software projects have multiple versions. A version of a project specifies a lower and an upper bound on the version of each dependency. The bounds on the dependencies are weakly increasing over the versions of a project.

Help Irina pick a version for each project such that all dependencies are satisfied.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^5$), the number of projects. The projects are numbered from 1 to n.
- n sections, the pth of which describes project p:
  - One line with two integers v_p and d_p ($1 \leq v_p \leq 10^9$, $0 \leq d_p < n$), the number of versions and the number of dependencies of project p.
  - d_p dependencies of project p, each consisting of three lines:
    - One line with an integer q ($1 \leq q \leq n$, $q \neq p$), indicating that project p depends on project q.
    - One line with v_p integers $\ell_1, \dots, \ell_{v_p}$ ($1 \leq \ell_i \leq v_q$, $\ell_i \leq \ell_{i+1}$ for each i), where ℓ_i is the minimum version of project q that is required by version i of project p.
    - One line with v_p integers $r_1, \dots, r_{v_p}$ ($1 \leq r_i \leq v_q$, $r_i \leq r_{i+1}$, $\ell_i \leq r_i$ for each i), where r_i is the maximum version of project q that is supported by version i of project p.

It is guaranteed that $\sum_{p=1}^{n} v_p \cdot d_p \leq 10^6$ and that the dependency graph has no cycles. The dependencies of each project are distinct projects.

Output

If there is no way to install everything, then output a line with "`impossible`". Otherwise, output "`possible`" followed by n integers giving a version of each project such that all dependencies are satisfied.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1

```
3
3 1
2
1 2 2
2 2 3
5 0
2 1
2
3 4
4 5
```

Sample Output 1

```
possible
3 3 1
```

Sample Input 2

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

Sample Output 2

```
impossible
```

Sample Input 3

```
2
1 1
2
400000000
600000000
1000000000 0
```

Sample Output 3

```
possible
1 400000000
```

Problem J: Jog in the Fog

Time limit: 1 second

On the night before the big event, your fellow distance runner Jesse is running a few final laps on the practice track near the Olympic village. You want to join them; however, it is so foggy that you can not even see one metre ahead.

You know the running route of Jesse, which they will repeat in loops. Using an optimal strategy, how long will you need on average to reach Jesse?

Both Jesse and you move in steps on a 2D grid, one grid cell at a time. You both move either up, down, left, or right a full cell. It takes one second to move one cell. Jesse will always be moving along their fixed route in a loop, and this route will visit no cell multiple times, except after they start looping their route. When you start your attempt, Jesse could be on any step of their route, uniformly at random. You may wait in a cell for 1 second instead of moving. If you and Jesse are in neighbouring cells and move towards each other, you will meet after 0.5 seconds.

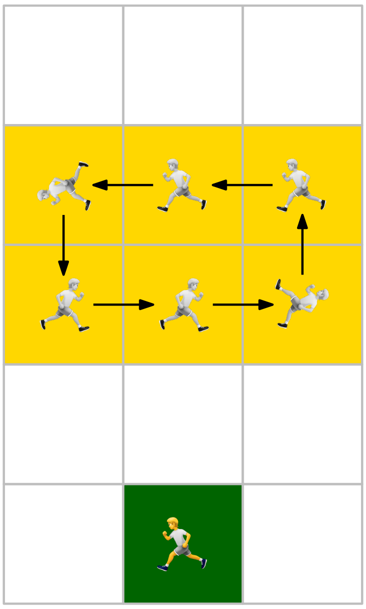


Figure J.1: Illustration of Sample Input 2, where the optimal expected time is 3.25 seconds.

Input

The input consists of:

- One line with three integers x, y, and n ($0 \leq x \leq 10^9$, $0 \leq y \leq 10^9$, $2 \leq n \leq 10^5$), describing your initial position (x, y) and the length n of Jesse's looping route.
- n lines, the ith of which contains two integers x_i and y_i ($0 \leq x_i \leq 10^9$, $0 \leq y_i \leq 10^9$), describing the ith position of Jesse's route.

All neighbouring steps of Jesse are guaranteed to be of distance exactly 1, so Jesse will always be moving one step up, down, left, or right. This guarantee also holds for the last and first position, allowing Jesse to loop their route. Additionally, no two positions in this route are the same. See Figure J.1 for an example.

Output

Output the expected time in seconds to reach Jesse assuming an optimal strategy. Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1

```
2 1 2
4 4
4 5
```

Sample Output 1

```
5.25
```

Sample Input 2

```
1 0 6
0 2
1 2
2 2
2 3
1 3
0 3
```

Sample Output 2

```
3.25
```

Problem K: Keeping Keys

Time limit: 1 second

Sample 5 printed using the eco-friendly printer.

At the Olympic Media Centre, an eco-friendly printer is provided to print schedules and results. In order to reduce the ink consumption, it's meant to charge 1 cent for each lowercase letter and 2 cents for uppercase letters. But you've found a glitch! It seems to not charge per letter, but per key pressed on the attached keyboard. As a skilled typist, you realize that you can hold a key to type any number of the corresponding character for a single cent. Furthermore, you realize that the cost for capital letters is implemented by using the cost of pressing the character's key and the cost of pressing the shift key. Finally, a space character can be typed by pressing the space bar, and this can be done even while holding shift (pressing the space bar costs 1 cent as well).

What is the minimum cost in cents you will be charged by the printer system if you use the printer as efficiently as possible? You can hold and release keys in any combination you want.

Input

The input consists of:

- One line with a string s ($1 \leq |s| \leq 1000$), consisting solely of the characters '`a`' to '`z`' and '`A`' to '`Z`' as well as spaces. The line neither starts nor ends with a space (but the input ends with a line break, which does not need to be paid).

Output

Output the minimum cost in cents needed to print the text.

Sample Input 1	Sample Output 1
`Hello`	`5`

Sample Input 2	Sample Output 2
`AAAAAAAAAA`	`2`

Sample Input 3	Sample Output 3
`Buy Llamas`	`11`

Sample Input 4	Sample Output 4
`A    AaA`	`5`

Sample Input 5

```
The RUNNERS set new world records
```

Sample Output 5

```
34
```

This page is intentionally left (almost) blank.

Problem L: Lookup Table Tennis
Time limit: 1 second

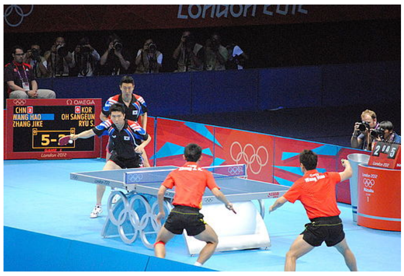


Table Tennis at the 2012 Summer Olympics.

For this year's Olympic tournament, the International Table Tennis Federation (ITTF) is pioneering a new ball tracking technology, intended to provide the live broadcast with additional data and insights. In the software, the playing area surrounding the table is overlaid with a three-dimensional grid of dimensions $n \times n \times n$. At any time, the position of the table tennis ball is assumed to be at an integer location within this grid, that is, at a position (x, y, z) where $0 \leq x, y, z \leq n$ are integers.

The system was created using the latest and greatest AI driven technology, which unfortunately means that none of the engineers behind it understand the details of its inner workings. In particular, it is not even possible to directly obtain the location of the table tennis ball from the computer! Instead, the only way to interact with the system is to ask queries consisting of a position and a distance, to which the system will respond whether the table tennis ball is at most the given distance away from the given position. In other words, each query is in the shape of a ball, and the system will tell you whether the table tennis ball is within the query ball.

The engineers now plan to use multiple of these queries to determine the location of the table tennis ball, and have asked you to assist them in doing so. Note that because the ball tracking needs to be done in real time you should never use more than 5000 queries to locate the table tennis ball.

Interaction

This is an interactive problem. Your submission will be run against an *interactor*, which reads from the standard output of your submission and writes to the standard input of your submission. This interaction needs to follow a specific protocol:

The interactor first sends one line with an integer n ($1 \leq n \leq 10^6$), the size of the grid. The table tennis ball is at some point (x_0, y_0, z_0) with integer coordinates $0 \leq x_0, y_0, z_0 \leq n$. For simplicity, its size is considered to be negligible.

Then, you need to start asking queries consisting of four integers "$x\ \ y\ \ z\ \ s$", where x, y and z ($0 \leq x, y, z \leq n$) are the centre coordinates and s ($0 \leq s \leq 3 \cdot n^2$) is the *square* of the radius of your query ball. The interactor will respond with 1 if (x_0, y_0, z_0) is within the query ball (including the boundary) and 0 otherwise. In other words, the answer will be 1 if and only if $(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 \leq s$.

When you ask the query "$x_0\ \ y_0\ \ z_0\ \ 0$", the interactor will respond 1 and the interaction ends. Your program must then exit.

You may send at most 5000 queries; any more than that results in a "Wrong Answer" verdict.

Interaction is not adaptive; the location (x_0, y_0, z_0) is fixed at the start and does not change.

After every request you should *flush* the standard output to ensure that the request is sent to the interactor. For example, you can use `fflush(stdout)` in C++, `System.out.flush()` in Java, `sys.stdout.flush()` in Python, and `hFlush stdout` in Haskell.

A testing tool is provided to help you develop your solution.

Read	Sample Interaction 1	Write
`5`		
		`0 0 2 25`
`1`		
		`0 0 2 24`
`0`		
		`2 3 2 1`
`0`		
		`2 3 2 3`
`1`		
		`3 4 2 1`
`1`		
		`3 4 2 0`
`1`		

Read	Sample Interaction 2	Write
`1`		
		`0 0 0 1`
`1`		
		`1 1 1 1`
`0`		
		`1 1 1 2`
`1`		
		`1 0 0 0`
`0`		
		`0 1 0 0`
`1`		

Problem M: Montage Matrix

Time limit: 2 seconds

Argentinian Field Hockey Silver Medal winners 2020.
By Secretaría de Deportes on Wikimedia

You have been hired by the organizing committee of the Olympic Games to take pictures of the event. Of course, you focus on the most important task: taking pictures of the winners!

Unfortunately, you quickly realize that for some competitions, it is quite complicated to take the perfect shot. Especially troubling are team sports like field hockey with a large number of players, both on the field and later on the photo.

The main challenge with so many people is to fit everyone on the photo! After all, so many people do not fit in the same row without anybody being cut off the photo. Therefore, each sports team must form multiple rows with at most w people per row.

Of course, for a perfect picture, all players must be visible on the photo and not be blocked by someone standing in front of them. This means that for each person on the photo, only people of strictly smaller height may stand in front of them.

However, you are unsure whether this is even possible for all teams taking part in the competition. Determine whether a given team can be correctly arranged for the photo.

Input

The input consists of:

- One line with two integers n and w ($1 \leq n \leq 2 \cdot 10^5$, $1 \leq w \leq n$), where n is the number of people in the team and w is the maximum number of people in one row.
- One line with n integers $h_1, \ldots, h_n$ ($0 \leq h_i \leq 10^9$ for each i), where h_i is the height of person i.

Output

If it is possible to arrange all people for the photo so that everyone is visible, then output "yes". Otherwise, output "no".

Sample Input 1	Sample Output 1
5 3 180 160 200 190 150	yes

Sample Input 2	Sample Output 2
3 1 150 150 140	no

This page is intentionally left (almost) blank.