



Winter Camp Contest 2021

NCTU PCCA

ID	Problem Name	Time Limit
A	A Water Problem	1 sec
B	Bongcloud	2 sec
C	Chicken Nuggets	1 sec
D	Discombobulator 3000	1 sec
E	Evenly Distributed	1 sec
F	Fishy Study	5 sec
G	Group-Theoretic Machine	1 sec
H	Halting Problem	2 sec

NCTU PCCA Winter Camp Contest 2021

Contest Information

Contest Rules

Contestants who fail to abide the rules will be disqualified.

1. A team shall consist of at most three (3) contestants.
2. No machine-readable materials (e.g., source codes, templates, etc.) are allowed. However, paper-based materials, such as textbooks, dictionaries, printed notes, etc., are allowed.
3. Contestants are only allowed to contact their teammates during the contest. Contestants shall not discuss with their coach and other teams.
4. Contestants shall only access the internet for downloading the problem description, submitting source codes, requesting problem clarification and checking the scoreboard. Any other type of internet access is prohibited.
5. A team shall not simultaneously use more than one computer to write programs during the contest. Contestant shall not use any other type of electronic devices, except extra monitors and printers.
6. All malicious actions interfering the contest are prohibited.

Scoring and Ranking

1. Contestants who are disqualified do not count towards the rankings.
2. Teams must submit their solutions via DOMjudge. Each submission will be given an ID and a timestamp of its submission time. It is guaranteed that a submission with greater ID is submitted later. The judge system will only response to submissions that is submitted within the contest duration (210 minutes). The response to each run must be one of the following:
 - **Correct:** The judge accepts your code.
 - **Compiler-Error:** Your code cannot be successfully compiled.
 - **TimeLimit:** Your program consumes too much time.
 - **Run-Error:** Your program terminates with an non-zero return code, which often means your program is terminated by the operating system.
 - **Wrong-Answer:** The judge rejects the output of your program.
 - **No-Output:** Your program does not generate any output.



3. Teams are ranked according to the most problems solved. Teams who solve the same number of problems are ranked by least total time. The total time is the sum of the time consumed for each problem solved. The time consumed for a solved problem is the time elapsed from the beginning of the contest to the submittal of the accepted run plus 20 penalty minutes for every rejected run for that problem regardless of submittal time. There is no time consumed for a problem that is not solved.
4. If multiple teams have the same number of problems solved and total time, then the winner goes to the team which first submitted their last accepted solution.



Almost blank page



Problem A

A Water Problem

Time limit: 1 second

Memory limit: 512 megabytes

Problem Description

Alice the shark loves water. In fact, she can't live without water.

Alice has n fish tanks. For $i = 1, 2, 3, \dots, n$, the i -th fish tank has a_i liters of water in it, where a_i is a positive integer. Each fish tank has a "potential value": if it contains x liters of water, then its potential $\mathcal{P}(x)$ is defined as $\mathcal{P}(x) = x \times \mathcal{D}(x)$, where $\mathcal{D}(x)$ is the sum of all digits in x . For example, $\mathcal{P}(123) = 123 \times (1 + 2 + 3) = 738$, and $\mathcal{P}(20000) = 20000 \times (2 + 0 + 0 + 0 + 0) = 40000$.

Please help Alice calculate the potential values of all her fish tanks.

Input Format

The first line contains an integer n . The second line contains n space-separated integers $a_1, a_2, \dots, a_n$.

Output Format

Print n lines. The i -th line should contain the potential of the i -th fish tank.

Technical Specification

- $1 \leq n \leq 10^3$
- $1 \leq a_i \leq 10^9$, for $i = 1, 2, \dots, n$

Sample Input 1

```
4
123 20000 5 998244353
```

Sample Output 1

```
738
40000
25
46917484591
```



Almost blank page

Problem B Bongcloud

Time limit: 2 seconds

Memory limit: 512 megabytes

Problem Description

Bob the shark enjoys playing chess. In fact, he has just won the **Legendary Grandmaster** title.

The sharks do not play chess on a regular chessboard, because they think that's boring. Instead, they have a board with n rows and m columns, where each square is either black or white. For each game of chess, they pick a region on the board as the “play region” of the game. The play region should be a rectangle of positive area, with each side parallel to one of the board's sides, and each vertex at the corner of some square on the board. Two regions are the same if they contain the same set of squares on the board, otherwise they are different.

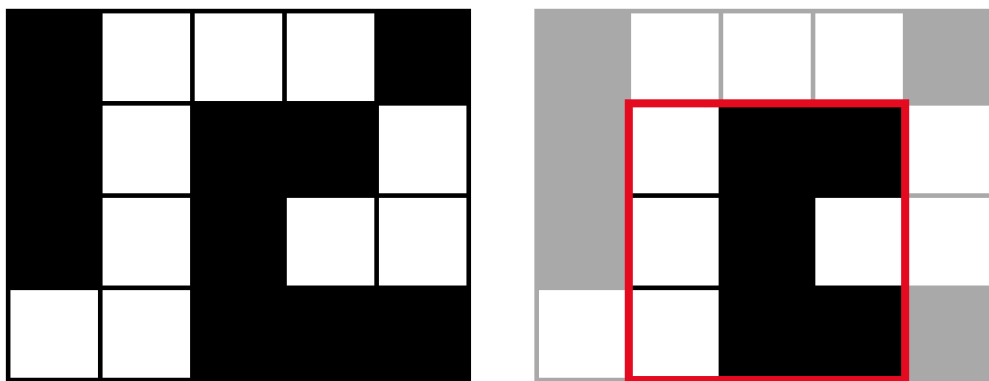


Figure 1: An example of a board with 4 rows and 5 columns. The red rectangle on the right marks a possible play region containing 9 squares.

Bob's favorite chess move is the “Bongcloud Attack”. He uses this move in chess tournaments very often, because it is always a powerful surprise attack that no opponent would expect. One day, he played the Bongcloud as usual, and the opponent took very long to think about the next move. While the opponent was still thinking, the following question came to Bob's mind: How many different play regions on the board are vertically symmetric? A region is called vertically symmetric if it looks the same after reversing its rows. For example, the region shown above is vertically symmetric. Please help Bob calculate the answer with a program.

Input Format

The first line contains two integers n and m . The following n lines describe the sharks' board. Each line contains m characters. The j -th character on the i -th line denotes the color of the square at the i -th row, j -th column: 1 for a white square, or 0 for a black square.

Output Format

Print one integer, the answer to Bob's question.



Technical Specification

- $1 \leq n, m$
- $n \times m \leq 10^6$

Sample Input 1

```
2 2
01
00
```

Sample Output 1

```
7
```

Sample Input 2

```
3 4
0101
0100
0001
```

Sample Output 2

```
42
```



Problem C

Chicken Nuggets

Time limit: 1 second

Memory limit: 512 megabytes

Problem Description

Carl the shark goes to his favorite restaurant for dinner. There is a special event happening today: if he plays and wins a game, the restaurant will offer free chicken nuggets as prize!

The game is set up as follows:

1. First, the restaurant manager arranges n plates in a row. They are numbered from 1 to n .
2. For $i = 2, 3, \dots, n$, the manager chooses an integer a_i such that $1 \leq a_i \leq i - 1$, and connects plate i and plate a_i with a single strand of spaghetti (it is attached to both plates).
3. For $i = 1, 2, \dots, n$, the manager chooses a positive integer c_i , and places c_i chicken nuggets on plate i .

Then, the game is played by one player. The player should do the following:

1. They select a set of plates, and remove all the selected plates. It is possible to select zero or all plates.
2. For each spaghetti strand, if it was attached to a plate that was removed, then the strand is removed as well.
3. For each remaining plate, they count the number of remaining spaghetti strands attached to it. If it is an even number, then the game is lost and the player receives nothing.
4. Otherwise, if the game isn't lost, then the player wins and receives all chicken nuggets on the remaining plates as prize.

Carl has a chance to play the game. Since chicken nuggets are his favorite food, he wishes to get as many chicken nuggets as possible. However, the number of plates is so large that he can't possibly calculate the result of the game. Can you help him out?

Input Format

The first line consists of one integer n . The second line contains $n - 1$ integers $a_2, a_3, \dots, a_n$. The third line contains n integers $c_1, c_2, \dots, c_n$.

Output Format

Print one integer, the maximum amount of chicken nuggets Carl can receive from the game.



Technical Specification

- $2 \leq n \leq 2 \times 10^5$
- $1 \leq a_i \leq i - 1$ for $i = 2, 3, \dots, n$
- $1 \leq c_i \leq 10^4$ for $i = 1, 2, \dots, n$

Sample Input 1

```
3
1 1
3 6 5
```

Sample Output 1

```
9
```

Sample Input 2

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

Sample Output 2

```
26
```



Problem D

Discombobulator 3000

Time limit: 1 second

Memory limit: 512 megabytes

Problem Description

Diana the shark has built a new machine for her project: the Discombobulator 3000. It is capable of solving any interactive programming problem that involves a permutation!

One day, while Diana was swimming peacefully, the machine fell out of her pocket and landed on the ocean floor. Because of this, it broke into two pieces. Diana was not able to repair it immediately, but she knew how the machine works, even if it was broken:

- One piece of the machine contains a sequence $a_0, a_1, \dots, a_{n-1}$ of n integers. The sequence is a permutation of the integers from 1 to n . That is, each integer between 1 and n (inclusive) appears in this sequence exactly once.
- The other piece of the machine contains a sequence $b_0, b_1, \dots, b_{n-1}$ of n integers. The sequence is also a permutation of the integers from 1 to n .
- There is an integer k such that $0 \leq k \leq n - 1$ and the relation $a_i = b_{(i+k) \bmod n}$ holds for $i = 0, 1, \dots, n - 1$.

Unfortunately, Diana only knew the integer n , and had no knowledge about k or the sequences. The only way to access the sequences is via a special button on the machine. When you press the button, the machine will ask you for two integers x and y and reply with the value $\max(a_x, b_y)$. The button cannot be pressed for more than $2n$ times in total, otherwise everything will explode.

Diana was discombobulated by the situation, and now she needs your help. If you can find out the integer k , then she will be able to combine the two pieces back into the Discombobulator 3000. Can you do that by interacting with the machine?

Input Format

This is an interactive problem.

The first line of input contains the integer n .

The input has q more lines if you press the button q times. Each of these lines contains an integer - the number that the machine replies with.

Output Format

Your output should consist of $q + 1$ lines if you press the button q times.

Each time you wish to press the button, print a line in the form “? x y ”, where x and y are the numbers you tell the machine. The machine’s reply will be sent as one line of input (described



above). Don't forget to flush the output stream after printing a line.

The last line should be in the form “! k ”, where k is the integer you have to find.

Technical Specification

- $2 \leq n \leq 200$
- $0 \leq k \leq n - 1$
- Your output must satisfy $0 \leq q \leq 2n$ and $0 \leq x, y \leq n - 1$ in each line.

Sample Input 1

4
1
2
2
4
2

Sample Output 1

? 2 3
? 0 1
? 0 3
? 3 1
? 2 1
! 1

Hint

In the example, the sequences are $a = [2, 3, 1, 4]$ and $b = [4, 2, 3, 1]$. The answer is $k = 1$.

Remember that you don't have to find the two sequences, only the integer k .



Problem E

Evenly Distributed

Time limit: 1 second

Memory limit: 512 megabytes

Problem Description

Emily the shark loves fish. In fact, she can't live without fish.

Emily has n fish tanks. Each fish tank may have some fish in it. For a set of fish tanks, they are “evenly distributed” if they can be partitioned into two groups of fish tanks, such that the total amount of fish in both groups are equal. For example,

- If three fish tanks have 2, 3, and 5 fish in them respectively, then they are evenly distributed, because you can partition them into two groups (the first and second tanks in one group and the third in the other group), so that each group has 5 fish in total.
- If three fish tanks have 2, 2, and 6 fish in them respectively, then they are not evenly distributed, because no such partition exists.

Emily has a total of k fish in her n fish tanks, and she wants to rearrange all fish such that:

- The amount of fish in each tank is a positive integer.
- The total amount of fish is k .
- For every non-empty subset of these fish tanks, the subset is **not** evenly distributed.

Please help Emily find such an arrangement, or determine if it's impossible.

Input Format

The input consists of two integers k and n .

Output Format

If the fish cannot be arranged according to Emily's rules, print the string **No**.

Otherwise, print the string **Yes** on the first line. Then on the second line, print a list of n space-separated integers, denoting the amount of fish in each tank. If there are multiple answers, print any one of them.

Technical Specification

- $1 \leq n \leq k \leq 200$

Sample Input 1

9 3

Sample Output 1

Yes
2 3 4



Sample Input 2

6 3

Sample Output 2

No



Problem F

Fishy Study

Time limit: 5 seconds

Memory limit: 512 megabytes

Problem Description

Frank the shark is conducting a study on the behavior of tropical fish. During the study, he discovered an interesting species of fish, called the Gol fish, which inhabit the underwater terrain in a very bizarre way.

The Gol fish live in an area that can be represented by a square grid with n rows and n columns. Each square can be described as a pair (i, j) , where i and j denotes the number of the row and column it belongs to. Rows and columns are numbered from 0 to $n - 1$. For each of the squares, there are either 0 or 1 fish living in it. Additionally, there is a sea urchin in one of the squares.

Let's define two different squares to be adjacent if they share a common edge, and be neighbors of each other if they share at least a common vertex. In each day, the following happens in order:

1. The sea urchin moves to a square that is adjacent to the one it is currently in.
2. The Gol fish's ecosystem updates itself to a new generation. For each square in the grid:
 - If the sea urchin is in this square, then no fish will live in it in the next generation.
 - Otherwise, if it has a fish in it, and exactly 2 or 3 neighbors have fish in them, then there will be a fish in this square in the next generation.
 - Otherwise, if exactly 3 neighbors have fish in them, then there will be a fish in this square in the next generation.
 - Otherwise, there will be no fishing living in it in the next generation.

The update happens for every square at the same time.

The "state" of all Gol fish can be described with $n \times n$ numbers, where each number denotes the amount of fish living in a square. Frank wants to conduct a experiment on the fish's behavior. Therefore, he adjusted the fish's positions so that the current state is s , and moved the sea urchin to (r, c) . He also wrote down a state t that he thinks is the "ideal" state for his study, and an integer d . Is it possible that the state becomes t in d or less days?

Input Format

The first line consists of two integers n and d . The second line consists of two integers r and c , denoting the sea urchin's location (r, c) .



Then n lines follow, each containing n digits. These lines describe the state s . The j -th digit on the i -th line denotes the number of fish living in the square at (i, j) .

Then n more lines follow, each containing n digits. These lines describe the state t . The j -th digit on the i -th line denotes the number of fish living in the square at (i, j) .

Output Format

If it's possible to reach the state t from s in d or less days, print the string **Yes**. Otherwise, print **No**.

Technical Specification

- $2 \leq n \leq 8$
- $0 \leq d \leq 8$
- Each digit describing s and t is either 0 or 1.

Sample Input 1

```
4 2
2 1
0100
1010
0100
0000
0100
1000
0100
0000
```

Sample Output 1

```
Yes
```

Sample Input 2

```
8 4
7 7
01000000
00100000
11100000
00000000
00000000
00000000
00000000
00000000
00000000
00100000
00010000
01110000
```

Sample Output 2

```
No
```



```

00000000
00000000
00000000
00000001
    
```

Sample Input 3

```

7 8
6 6
0100000
0010000
1110000
0000000
0000000
0000000
0000000
0000000
0000000
0000000
0001000
0011000
0000000
0000000
    
```

Sample Output 3

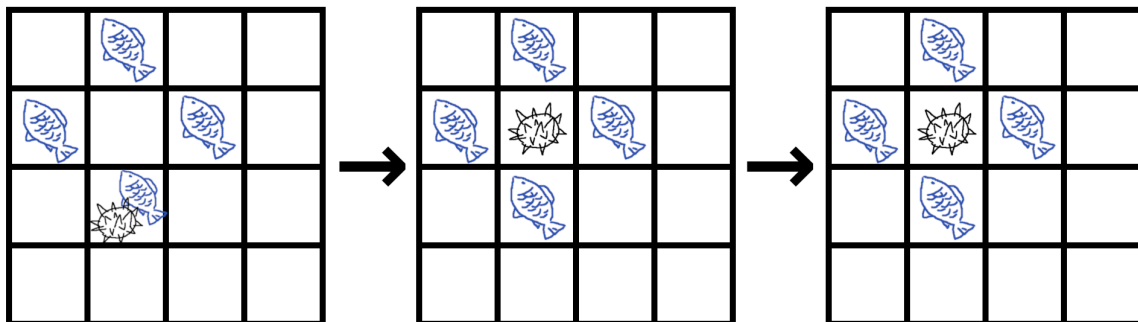
```

Yes
    
```

Hint

In the first example, it is possible to reach state t in 2 days:

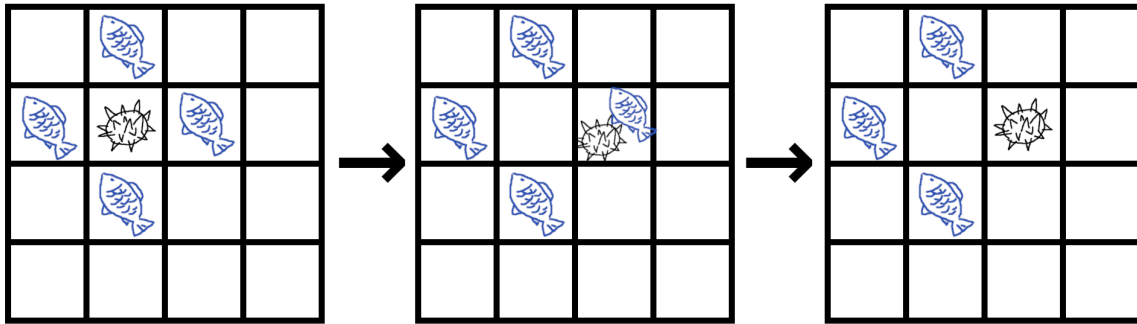
Day 1



 = Gol Fish
  = Sea Urchin



Day 2



 = Gol Fish  = Sea Urchin

Problem G

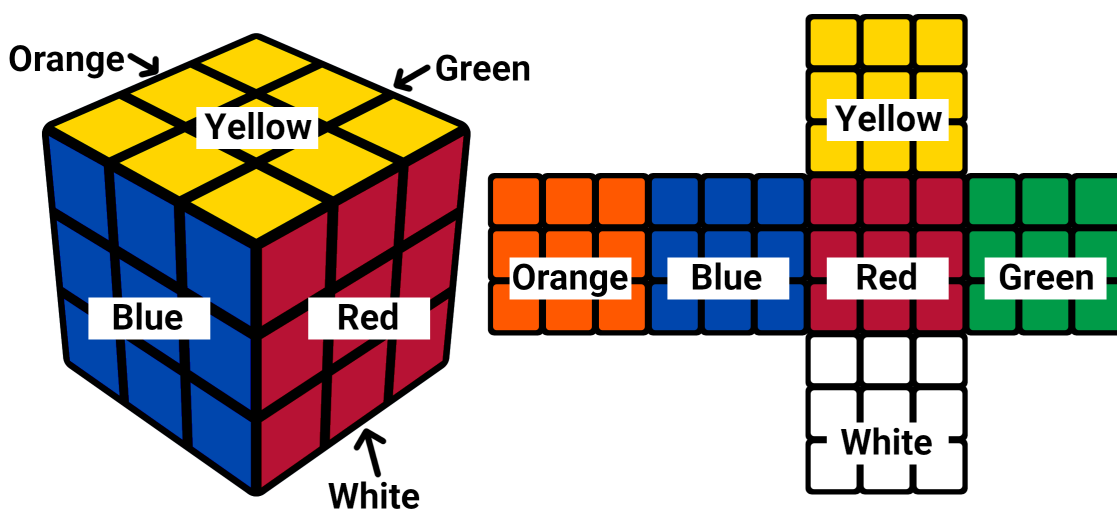
Group-Theoretic Machine

Time limit: 1 second

Memory limit: 512 megabytes

Problem Description

Gary the shark loves to play puzzles. One day, he learned about the most famous and mind-bending puzzle ever existed: the Rubik's Cube. The Rubik's Cube is a cube-shaped puzzle toy. Each of its faces is divided into a 3×3 grid, where each "slice" can be rotated independently. For a solved Rubik's Cube, every side has a unique color, and its color layout is shown below:



Gary has a Rubik's Cube sitting in his toy box, and he's having a hard time solving it. (This is probably due the fact that, according to scientists, most shark species are color blind.) To solve this problem once and for all, he created the powerful Group-Theoretic Machine.

The Group-Theoretic Machine (GTM) is a special machine that can solve any scrambled Rubik's Cube using group-theoretic algorithms. It has six sensors O, R, W, Y, G, B , which correspond to the colors orange, red, white, yellow, green, and blue respectively. Each sensor is located at a fixed position. To increase the stability of the machine, the sensors are designed so that the segments $\overline{OR}$, $\overline{WY}$, and $\overline{GB}$ are parallel to the x -, y -, and z -axes respectively.

For the machine to work, one should first activate it by placing a solved Rubik's Cube inside, such that each sensor touches a face with the corresponding color (touching that face's boundary also counts). Gary has just received a new Rubik's Cube with a side length of d , and he wonders if it can be used to activate the machine. The cube may be oriented or moved freely.

Input Format

The first line contains one integer t denoting the number of test cases. Then t test cases follow.

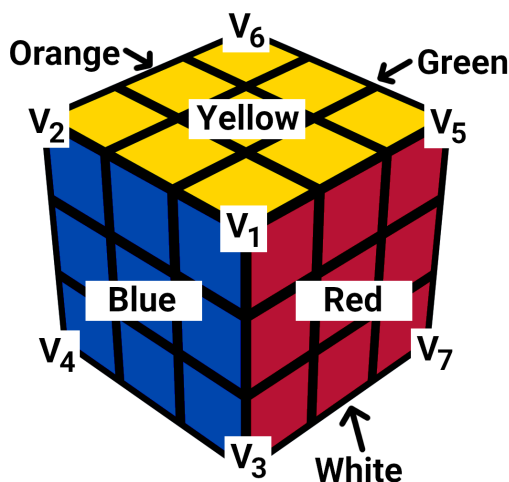
In each test case, there are 7 lines. The first line contains an integer d . The remaining 6 lines

describe the sensors O, R, W, Y, G, B respectively. Each of them contains three integers x, y, z denoting the coordinates of a sensor's location.

Output Format

For each test case, if the machine cannot be activated with Gary's new cube, print a line containing the string NO.

Otherwise, print the string YES. Consider a placement of the Rubik's Cube that can activate the machine. Let $V_1, V_2, \dots, V_8$ denote the Rubik's Cube's vertices in the order shown (V_8 is the remaining vertex at the back so cannot be seen here):



Print 8 more lines. The i -th ($i = 1, 2, \dots, 8$) line should consist of three space-separated real numbers x, y, z describing the coordinates of V_i under said placement. If there are multiple answers, output any one of them.

Technical Specification

- $1 \leq t \leq 10^3$
- $1 \leq d \leq 10^4$
- All coordinates in the input are integers between -10^4 and 10^4 (inclusive).
- No two input points in a test case are equal.
- The line segment $\overline{OR}$ is parallel to the x -axis.
- The line segment $\overline{WY}$ is parallel to the y -axis.
- The line segment $\overline{GB}$ is parallel to the z -axis.
- Your answer will be accepted if the cube you output satisfies:
 - The length of each edge is in the range $(d - 10^{-6}, d + 10^{-6})$.
 - For each interior angle θ of each face, $\cos(\theta)$ in the range $(-10^{-6}, 10^{-6})$.

– For each face, the distance between it and the corresponding sensor is less than 10^{-6} .

- You may print at most 30 characters for each number.
- The checker has at least 256 bits of floating-point precision.

Sample Input 1

```
3
2
-1 0 0
1 0 0
0 -1 0
0 1 0
0 0 -1
0 0 1
5
2 7 8
8 7 8
6 3 8
6 9 8
6 7 4
6 7 10
2
-1 0 0
1 0 0
0 1 0
0 -1 0
0 0 -1
0 0 1
```

Sample Output 1

```
YES
1.000000 1.000000 1.000000
-1.000000 1.000000 1.000000
1.000000 -1.000000 1.000000
-1.000000 -1.000000 1.000000
1.000000 1.000000 -1.000000
-1.000000 1.000000 -1.000000
1.000000 -1.000000 -1.000000
-1.000000 -1.000000 -1.000000
YES
7.666667 8.666667 9.666667
3.500000 10.159407 7.340593
5.340593 4.500000 11.159407
1.173927 5.992740 8.833333
9.159407 6.340593 5.500000
4.992740 7.833333 3.173927
6.833333 2.173927 6.992740
2.666667 3.666667 4.666667
NO
```

Note

The images above are licensed under CC-BY-SA 4.0 and are modified from existing images by other authors. The original files can be found here:

1. 3D Rubik's Cube (CC-BY-SA 4.0):

<https://commons.wikimedia.org/wiki/File:Solved1.svg>

2. Rubik's Cube net (CC0):

https://commons.wikimedia.org/wiki/File:Rubik%27s_cube_colors.svg



Almost blank page



Problem H

Halting Problem

Time limit: 2 seconds

Memory limit: 512 megabytes

Problem Description

Hans the shark lives in an underwater country that is divided into n states, numbered from 1 to n . There are m bidirectional roads in this country, each of which connects two different states. It is possible to travel from any state to any other state by swimming along the roads.

Hans is planning some journeys around the country. He first chooses a set of states in the country, and calls them the “halting states”. Then he defines a “journey” as follows:

1. He chooses any halting state as the start state. It is his initial current state.
2. For each day, he chooses some state such that it is connected with his current state by a road. He then swims along that road, so the chosen state becomes the new current state.
3. After swimming along a road, if the current state is a halting state and is not the start state, then he may choose to end the journey immediately (he may also choose to not end).

In other words, a journey is a walk that starts from a halting state and ends in a different halting state. A walk consists of a sequence of states in which every two adjacent states are connected by a road.

The length of a journey is defined as the number of days spent before it ends, which is also the total number of roads he has traveled. A journey is called “happy” if its length is a multiple of k , where k is Hans’ lucky number. Hans wants to choose the maximum number of halting states, such that every possible journey that ends is happy. Can you help him do it?

Input Format

The first line contains three integers n, m, k . Then m lines follow. The i -th line contains two integers u_i and v_i describing a road between the states numbered u_i and v_i .

Output Format

Output one integer, the maximum number of halting states that Hans can choose such that every possible journey that ends is happy.

Technical Specification

- $2 \leq n \leq 2 \times 10^5$
- $n - 1 \leq m \leq \min(2 \times 10^5, \frac{n(n-1)}{2})$
- $1 \leq k \leq 2 \times 10^5$



- $1 \leq u_i < v_i \leq n$ for $i = 1, 2, \dots, m$
- $(u_i, v_i) \neq (u_j, v_j)$ when $i \neq j$

Sample Input 1

```
5 6 2
1 3
1 5
2 5
2 3
2 4
1 4
```

Sample Output 1

```
3
```

Sample Input 2

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

Sample Output 2

```
1
```

Hint

Remember that if no journey can possibly end, then the constraint is satisfied.

In the first example, Hans can choose the states 3, 4, and 5. He can also choose the states 1 and 2, but the number of halting states is not maximized.

In the second example, the maximum number of halting states is 1, as Hans cannot choose more than 1 halting states. For example, if the states 1 and 4 are chosen, then there exists a journey $1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 3 \rightarrow 4$ which has length 5 so is not happy. On the other hand, if state 1 is the only halting state, then no journey can possibly end, so every journey that ends is happy.