0_Hello World

(30 points)

Time Limit: 1 second
Memory Limit: 256MB

Introduction

YTP Contest has started!

Let's verify everything first.

Is the internet setting correct?

Is the source code submission working well?

Do you use STDOUT output for program solutions?

Everything is ready! Go get 30 points now!! Go! Go! Go!

Statement

Please write a program to output Hello World!

Input Format

This problem requires no input.

Output Format

[A~Z][a~z], space, and common English punctuation.

Constraints

[A~Z][a~z], space, and exclamation mark "!".

Test Cases

Input 1

(no input)

Output 1

```
Hello World!
```

Illustrations

Input 1 has no input, simply output Hello World!

1_No_1_can_solve_this_problem!

(2 points /8 points)

Time Limit: 1 second
Memory Limit: 256MB

Statement

As the title says, no 1 can solve this problem!

Given a positive integer N, output the minimum non-negative integer K such that there is no 1's in the decimal representation of the integer $N + K$.

Input Format

The input contains a single integer, N. The input is guaranteed to have no leading zeros.

Output Format

Output a non-negative integer K as requested in the problem. Please note that unless your output is 0, there should be no leading zeros in your output.

Constraints

- $1 \leq N < 10^{100000}$

Subtasks

- Subtask 1 satisfies that $N < 10^6$.
- Subtask 2 has no additional constraints.

Test Cases

Input 1

```
1
```

Output 1

```
1
```

Input 2

```
11
```

Output 2

```
9
```

Input 3

```
42
```

Output 3

```
0
```

Illustrations

For Input 1, adding 1 to 1 results in 2, which has no 1's in its decimal representation.

For Input 2, adding 9 to 11 results in 20, which has no 1's in its decimal representation.

For Input 3, there is already no 1's in 42, so output 0.

2_Energy_Crisis

(2 points/8 points)

Time Limit: 1 second

Memory Limit: 256 MB

Statement

In the year 2050, Earth's resources have become severely scarce, and humanity, in a bid to secure energy, has developed a type of energy crystal known as "Numcrystals." Each Numcrystal bears a number that, under the right conditions, can release immense energy, especially when specific multiplicative pairings, known as the energy multiplier "K", are found. In other words, if you can find two Numcrystals whose numbers multiply to equal K, a large amount of energy can be harnessed.

As a Numcrystal engineer, you are tasked with the following: from an initial set of N Numcrystals $A_1, A_2, \ldots, A_N$ (where A_i represents the number on the i^{th} Numcrystal), perform precise operations to harness the energy of the multiplier "K". Your operations include:

1. **Energy Amplification (cost x)** — Select a Numcrystal and amplify its energy by increasing its value by one.
2. **Energy Suppression (cost y)** — Select a Numcrystal and suppress its energy by decreasing its value by one.
3. **Numcrystal Fusion (cost z)** — Select two Numcrystals for fusion; the new Numcrystal returned to the set will have a number equal to the sum of the two fused Numcrystals' numbers, and the fused Numcrystals will disappear.

After the operations on the Numcrystal set are completed, you aim to find two Numcrystals whose numbers multiply to equal K, and you must calculate the minimum cost to achieve this. Demonstrate your prowess as a Numcrystal engineer by writing a program to solve this problem.

Input Format

The first line inputs two numbers N, K, representing the initial number of Numcrystals and the energy multiplier, respectively.

The second line inputs three numbers x, y, z, representing the costs of each operation.

The third line inputs N numbers $A_1, A_2, \ldots, A_N$, representing the numbers on the initial set of Numcrystals.

Output Format

Output a single number representing the minimum cost to achieve a pair of numbers in the set whose product equals K.

Constraints

- $2 \leq N \leq 10$.

- $1 \leq K \leq 1000$.
- $1 \leq A_i \leq 1000$ $(\forall i \in [1, N])$.
- $1 \leq x, y, z \leq 10^7$.

Subtasks

- Subtask 1 satisfies that $x = 1, y = 1, z = 10^7$.
- Subtask 2 has no additional restrictions.

Test Cases

Input 1

```
3 10
1 1 1
2 2 2
```

Output 1

```
2
```

Input 2

```
5 20
1 1 10
1 2 3 5 7
```

Output 2

```
1
```

Illustrations

In Example 1, we can first fuse two of the Numcrystals to obtain a Numcrystal with a value of 4, and then increase this Numcrystal's value by 1 to get Numcrystals with values of 2 and 5, which can then be multiplied to equal 10.

3_Queries_on_Destring

(10 points)

Time Limit: 2 seconds
Memory Limit: 1024 MB

Statement

De-Cheng Liu likes the number 49 very much, so he defines that a **destring** is a string where every character is `4` or `9`.

De-Cheng Liu received a **destring** s from his friend LDC. He wants to get 49 points on IMO, so for all positive integers i which satisfies $1 \leq i \leq Q$, De-Cheng Liu wants to two positive integers x, y such that

- $L_i \leq x \leq y \leq R_i$
- $s[x:y]$ is not a **perfect square**
- When x, y satisfy the two conditions above, $y - x + 1$ should be as small as possible

For all positive integers i which satisfies $1 \leq i \leq Q$, determine whether De-Cheng Liu can find two positive integers x, y which satisfy all conditions. If he can, find the minimum possible $y - x + 1$ and how many pairs x, y satisfy all condtions.

The statements below are the definitions used in this problem :

- For a string t and two positive integers l, r which satisfy $1 \leq l \leq r \leq |t|$ ($|t|$ is the length of t), define $t[l:r]$ to be the substring of t which starts at the l-th character of t and ends at the r-th character of t, which is $t_l t_{l+1} \cdots t_r$.
- For a string t which every character is a digit, define t to be a **perfect square** if $\sum_{i=1}^{|t|} t_i \cdot 10^{|t|-i}$ is a **perfect square**
- For a non-negative integer x, define x to be a **perfect square** if at least one of the following two conditions holds
  - There is a non-negative integer y such that $x = y^2$
  - $x = 4$

Input Format

The first line of the input contains a positive integer $|s|$.

The second line of the input contains a **destring** s.

The third line of the input contains a positive integer Q.

For all positive integers i which satisfies $1 \leq i \leq Q$, the $i+3$-th line of the input contains two positive integers L_i, R_i separated by a space.

Output Format

For all positive integers i which satisfies $1 \leq i \leq Q$, if De-Cheng Liu can't find positive integers x, y which satisfy all conditions, then the i-th line of the output is `-1`, otherwise the i-th line of the output contains two positive integers, the first positive integer is the minimum value of $y - x + 1$, and the second positive integer is the number of pairs x, y which satisfy all conditions.

Constraints

- $1 \leq |s| \leq 10^6$
- s is a **destring**
- $1 \leq Q \leq 10^6$
- $1 \leq L_i \leq R_i \leq |s|$

Subtasks

- Subtask 1 no other constraints (10 points).

Test Cases

Input 1

```
3
499
2
1 2
2 3
```

Output 1

```
-1
2 1
```

Illustrations

When $i = 1$, because $4, 9, 49$ are all **perfect squares**, there is no x, y which satisfy all conditions.
When $i = 2$, because $4, 9$ are **perfect squares**, but 99 is not, $x = 2, y = 3$ is the only choice.

4_Badminton_Game

(10 points)

Time Limit: 1 second
Memory Limit: 256 MB

Statement

Momo and Sakura love playing badminton and often meet up for matches. A badminton match consists of multiple round, with each round having one of them serving. Before the beginning of the first round, their scores are both zero. After a series of rallies, the winner of each round is determined, and the winner earns a point. The serving rules are as follows:

- Momo serves in the first round, and thereafter, the winner of the previous round serves in the next round.
- If the server's current score is even, they must serve from the right side of the court; otherwise, they serve from the left side.

For example, the following is a match consisting of four rounds:

Round	Server	Momo's Score when Serving	Sakura's Score when Serving	Serving Side	Winner of the Round
1	Momo	0	0	Right	Momo
2	Momo	1	0	Left	Sakura
3	Sakura	1	1	Left	Sakura
4	Sakura	1	2	Right	Momo

Yesterday, Momo and Sakura played an N-round badminton match, but today they realized they forgot the final scores. They only remember whether each round was served from the left or the right side. Momo wants to know the highest possible score she could have achieved given this information. Can you help her? If there is no possible legitimate sequence of the match that can result in the given serving information, report this error as well.

Input Format

The first line contains a positive integer T, representing the number of test cases.

Each test case consists of two lines: the first line contains a positive integer N, representing the number of rounds in the badminton match.

The second line contains a string S of length N, where S_i is $\mathtt{L}$ if the i-th round was served from the left side, and $\mathtt{R}$ if it was served from the right side.

Output Format

Output T lines, where the i-th line contains `-1` if the serving information for the i-th test case is invalid. Otherwise, output the highest possible score Momo could have achieved.

Constraints

- $1 \leq T \leq 10000$
- $1 \leq N \leq 5 \times 10^5$
- $|S| = N$
- $S_i \in \{\mathtt{L}, \mathtt{R}\}$
- The sum of all N across all test cases does not exceed 5×10^5

Test Cases

Input 1

```
4
4
RLLR
3
LRR
2
RR
7
RLRLRLR
```

Output 1

```
3
-1
-1
7
```

Illustrations

In the first sample input, if the winners of each round are "Momo, Sakura, Momo, Momo", the resulting serving order is $\mathtt{RLLR}$. It can be proven that there is no better result, so the answer is 3.

In the second sample input, it is invalid because the first round must be served from the right side.

5_Competitive_Programming_Is_A_World_of_Imagination

(6 points /9 points)

Time Limit: 1 second
Memory Limit: 256 MB

Statement

Little Lee, Little Ming, and Little Yee formed a team to participate in the YTP Programming Challenge Camp this year. However, their recent practice performances have not been very good. To enhance their team cohesion and change their mood, they decided to take a walk on Yangmingshan. In the evening, they sat on the grass together and watched the stars. Thinking about the many experts and their recent poor practice, Little Lee murmured, "I can't even imagine myself winning the YTP Programming Challenge Camp."

At this moment, Little Po suddenly appeared and said, "Competitive programming is a world of imagination. If you can't imagine it, you can't achieve it!"

Little Ming found this very insightful, so he suggested that they all practice their imagination together. Looking at the N visible stars in the sky, he noticed that they precisely form the vertices of a convex polygon! More specifically, the entire sky can be seen as a 2D plane, with each star being a point on this plane. The coordinates of the i-th star are $p_i = (x_i, y_i)$, and there is a way to renumber all the stars so that connecting p_1 to p_2, p_2 to p_3, ..., p_{N-1} to p_N, and p_N to p_1 forms a convex polygon with N vertices.

After Little Ming pointed this out, Little Lee had another idea: using different sequences to connect the stars could produce different shapes, so why not use the stars to draw a star! Here's how Little Lee wants to draw a star:

1. Divide the N stars into $k \geq 1$ groups, with each group containing at least 3 stars. Each star must belong to exactly one group.
2. For each group of stars, arrange them in a certain order, then connect the first star to the second, the second to the third, ..., and the last star back to the first.

In simple terms, Little Lee wants to draw k strokes, each starting from a star, passing through some stars, and then returning to the starting star, with each star being drawn exactly once (a star that is both the start and end counts only once).

Obviously, Little Lee will draw N line segments. He hopes that:

- For any line segment, except for itself and its two adjacent line segments, all other line segments in the same stroke intersect with it. A line segment's adjacent segments are the previous and next segments in the stroke; the previous segment of the first segment is the last segment, and the next segment of the last segment is the first segment.
- For any line segment, there is at least one line segment in any other stroke that intersects with it.

Although Little Yee doesn't like Little Po, who appeared out of nowhere and eavesdropped on their conversation, this might be useful for the YTP Programming Challenge Camp, so please help them draw the star.

Input Format

The first line contains an **odd number** N, the number of stars.

The next N lines each contain two integers x_i and y_i, representing the coordinates of the i-th star.

Output Format

The first line should output an integer k, the number of groups the stars are divided into.

The next k lines should describe each group of stars. Each line should start with an integer $t_i \geq 3$, the number of stars in that group, followed by t_i integers $s_{i,1}, s_{i,2}, \ldots, s_{i,t_i}$, representing the indices of the stars in the order they are connected. This means connecting star $s_{i,1}$ to $s_{i,2}$, $s_{i,2}$ to $s_{i,3}$, ..., and s_{i,t_i} back to $s_{i,1}$.

If there are multiple ways to draw the stars that satisfy the requirements, output any one of them.

Constraints

- $3 \leq N \leq 1000$
- N is odd
- $\forall 1 \leq i \leq N,\ |x_i|, |y_i| \leq 10^9$
- The stars are precisely all vertices of a convex polygon
- The stars are **not necessarily** given in the order of the vertices of the convex polygon.

Subtasks

- Subtask 1: The stars are given in the order of the vertices of the convex polygon in counterclockwise order (6 points)
- Subtask 2: No additional constraints (9 points)

Test Cases

Input 1

```
9
6 8
6 4
10 2
14 2
18 4
20 8
18 12
12 14
8 12
```

Output 1

```
1
9 1 6 2 7 3 8 4 9 5
```

Input 2

```
3
4 9
8 1
1 1
```

Output 2

```
1
3 1 2 3
```

Input 3

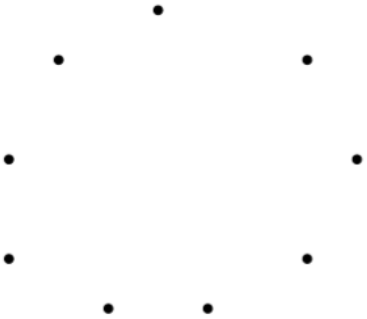
```
5
4 1
19 14
17 17
8 0
13 0
```

Output 3

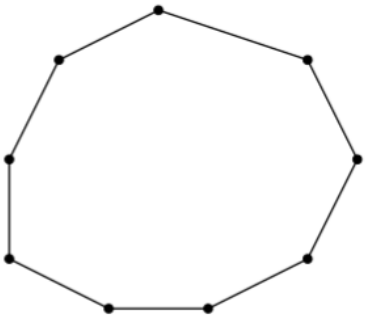
```
1
5 1 5 3 4 2
```

Illustrations

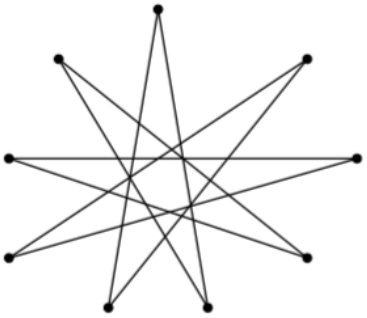
The picture below illustrates Sample 1, with some examples of "adjacent segments" listed on the right. (b) represents the way of drawing a star in Output 1.



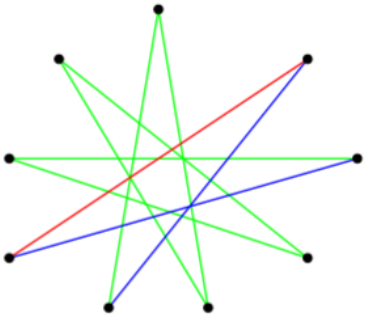
(a) Stars in the sky



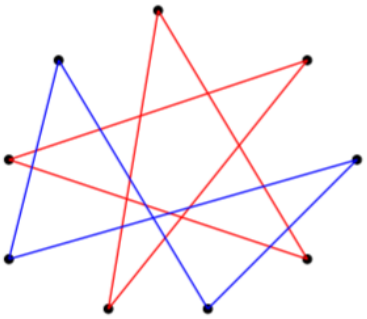
(b) Connect the stars to form a convex polygon



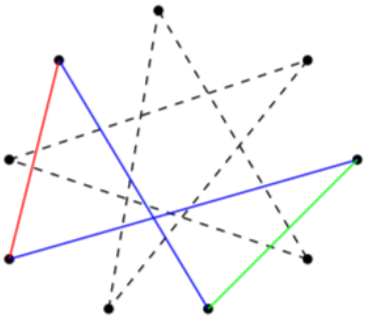
(c) A way to draw a star



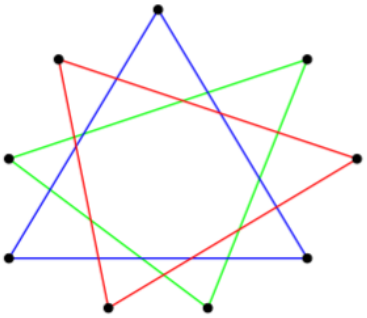
(d) The two blue segments are adjacent segments of the red one. The red segment should intersect all green segments. (Whatever the starting star is, each segment has the same adjacent segments!)



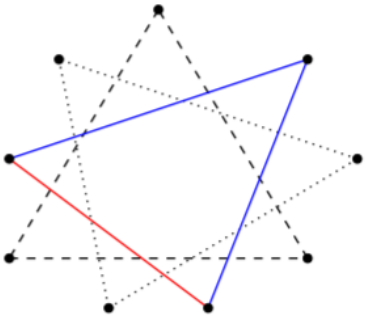
(e) An **incorrect** way to draw a star, where the stars are divided into 2 groups.



(f) The two blue segments are adjacent segments of the red one. The green segment should intersect the red segemnt.



(g) Another way to draw a star, where the stars are divided into 3 groups.



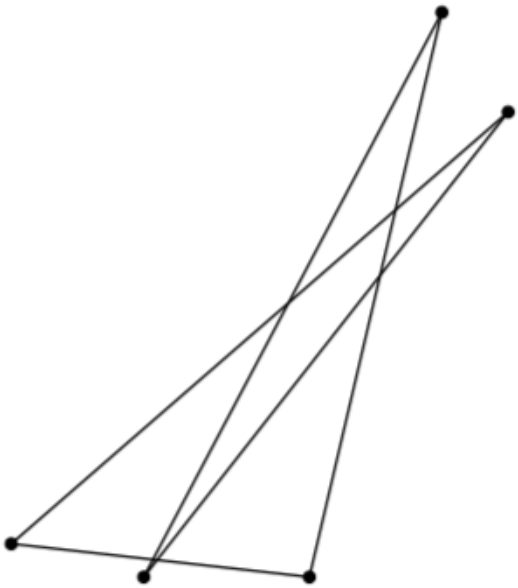
(h) The two blue segments are adjacent segments of the red one.

The following output is also a correct answer, corresponding to (g) in the picture above.

```
3
3 1 4 7
3 2 5 8
3 3 6 9
```

The picture below shows the results of Sample 2 and 3:

(a) Sample 2



(b) Sample 3

6_Omelet_Loves_Ramen

(5 points / 10 points)

Time Limit: 2 seconds
Memory Limit: 256 MB

Statement

Omelet plans to open a ramen shop after extensive preparations! The shop will offer M different types of ramen he has carefully developed. Omelet uses pots to cook the ramen. Since the flavors of his ramen are quite strong, if a pot is used to cook one type of ramen and then used to cook another type without being washed, the flavors will mix and make the ramen taste bad. To avoid this, Omelet follows these rules to decide which pot to use when he needs to cook ramen type x:

- If there is a pot that was last used to cook ramen type x, he will use that pot.
- Otherwise, if there is at least one unused pot, he will use one of those.
- Otherwise, he will wash the pot that has not been used for the longest time and use it to cook the ramen.

For example, if Omelet has 2 pots and needs to cook ramen types in the order 1, 2, 3, 2, 3, 1, the sequence will be as follows:

Ramen Type	Pot Number	Washed or Not	Explanation
1	1	No	Pot 1 has not been used before
2	2	No	Pot 2 has not been used before
3	1	Yes	Pot 1 hasn't been used for the longest time
2	2	No	Pot 2 was last used for ramen type 2
3	1	No	Pot 1 was last used for ramen type 3
1	2	Yes	Pot 2 hasn't been used for the longest time

Omelet has predicted the number of customers who will arrive on the opening day and the type of ramen each customer will order. For all K from 1 to M, help Omelet calculate the number of times he needs to wash a pot if he has K pots.

Input Format

The first line contains two positive integers N and M, representing the number of customers and the number of ramen types, respectively.

The second line contains N positive integers $a_1, a_2, \ldots, a_N$, where a_i represents the type of ramen ordered by the i-th customer.

Output Format

Output a single line with M numbers. The i-th number represents the number of times Omelet needs to wash a pot if he has i pots.

Constraints

- $1 \leq M \leq N \leq 5 \times 10^5$
- $1 \leq a_i \leq M$

Subtasks

- Subtask 1: $N \leq 2000$ (5 points)
- Subtask 2: No additional constraints (10 points)

Test Cases

Input 1

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

Output 1

```
5 2 0
```

Input 2

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

Output 2

```
3 2 0
```

Input 3

```
9 9
8 6 4 1 9 7 5 3 2
```

Output 3

```
8 7 6 5 4 3 2 1 0
```

Input 4

```
20 7
4 5 2 7 2 7 1 3 1 3 4 2 4 1 5 1 2 5 4 1
```

Output 4

```
19 12 10 5 3 0 0
```

Illustrations

In the second example for $K = 2$, the explanation is as follows:

Ramen Type	Pot Number	Washed or Not	Explanation
1	1	No	Pot 1 has not been used before
1	1	No	Pot 1 was last used for ramen type 1
3	2	No	Pot 2 has not been used before
2	1	Yes	Pot 1 hasn't been used for the longest time
2	1	No	Pot 1 was last used for ramen type 2
1	2	Yes	Pot 2 hasn't been used for the longest time

Thus, Omelet needs to wash the pots 2 times in total.

7_Duck_Collection

(15 points)

Time Limit: 1 second

Memory Limit: 256 MB

Statement

In the country of YTP, there are n ponds, each containing an infinite number of ducks. All ducks in the same pond are of the same type.

One day, the duck manager "Gua Gua" decided to play a game of collecting ducks. The goal of the game is to collect at least one duck of each type, and since there are n ponds, there are naturally n types of ducks.

Each round of the game involves making a request. During each request, you can select two ponds (which can be the same) and take one duck from each. If the chosen ponds are (i, j), then the cost of this request will be $c_{i,j}$. Note that the order is important in each request, which means that it is not gauranteed that $c_{i,j} = c_{j,i}$.

The objective, besides collecting all types of ducks, is to minimize the cost. Please design a program to calculate the minimum cost.

Input Format

The first line contains a positive integer n, representing the number of ponds and types of ducks.

The next n lines each contain n positive integers. The j-th integer on the i-th line is $c_{i,j}$, which is as described above.

Output Format

Output one integer in a single line, representing the minimum cost to collect all type of ducks.

Constraints

- $1 \leq n \leq 18$.
- $1 \leq c_{i,j} \leq 10^9$.

Test Cases

Input 1

```
3
10 9 10
4 5 10
7 5 1
```

Output 1

```
5
```

Input 2

```
3
1 10 10
10 1 10
10 10 1
```

Output 2

```
3
```

Input 3

```
3
1 1 1
10 10 10
10 10 10
```

Output 3

```
2
```

Illustrations

In example 1, we can perform the following operations in each turn:

- In the first turn, we take the 2-nd and 1-st ducks, costing 4.
- In the first turn, we take two 3-rd ducks, costing 1.
  The total cost is 5.

8_Kita_Storm

(15 points)

Time Limit: 1 second
Memory Limit: 512MB

Statement

NASA (Network Administration and System Administration) is a course in NTU, which gives students hands on experience with various network services and systems.
This is one of the homework problems in the course:

"One computer in the department's labs suddenly started sending large amounts of broadcasts, radiating **Kita-aura** through their **Kita-packets**. Bocchi was overloaded by the **Kita-aura** and started shaking in low self esteem. All of the sudden, the Kita Storm suddenly stopped. Given the IP address and the structure of the network, can you find where the rogue computer is, to prevent Kita Storm from happening again?" - based on NASA 2023 HW2

In the `.pka` file given in the homework, the network structure of the department's lab computers is in the form of a rooted tree, with nodes labeled from 1 to N, node 1 is the root. The leaf nodes represent lab computers, and all other nodes represent network switches. Let x be the node index of the rogue computer. From the Kita-packets, we know that the IP of the rogue computer is `192.168.0.53`. Then from typing the following commands on switch u

```
ping 192.168.0.53
show arp
show mac address-table
```

We can find the port from u in which the rogue computer is located. More specifically, we can find a node a_u adjacent to u, so the simple path from x to u goes through a_u.

Some switches are locked, and you cannot run the commands on those switches. Under these constraints, it is possible that there are multiple candidates for the rogue computer x. Write a program to list all possible rogue computers. Notice that a rogue computer must be a leaf node in the tree.

Input Format

The first line contains an integer N, which is the number of nodes on the tree.
The next $N-1$ lines have two integers u, v, which means that there is an edge (u, v) in the tree. Note that the edges are undirected.
The final line contains N integers $a_1, a_2, \ldots, a_N$. If $a_i = 0$, then you don't have access to node i, otherwise a_i is the node adjacent to i between the path from i to x.

Output Format

On the first line output k, the number of candidates for the rogue computer.
On the second line output k integers $b_1, b_2, \ldots, b_k$ in increasing order, which are the indices that are possible rogue computer candidates.

Constraints

- $2 \leq N \leq 2 \times 10^5$
- $1 \leq u, v \leq N$
- $0 \leq a_i \leq N$，If $a_i \neq 0$，then node a_i is adjacent to node i。
- If node p a leaf，$a_p = 0$
- There is no conflicting information given by a_i

Subtasks

There is only one subtask.

Test Cases

Input 1

```
7
1 2
1 3
2 4
2 5
3 6
3 7
2 4 1 0 0 0 0
```

Output 1

```
1
4
```

Input 2

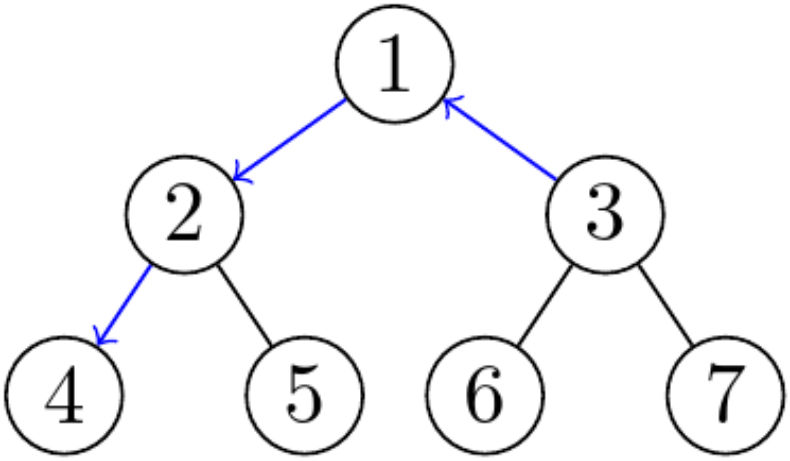
```
7
1 2
1 3
2 4
2 5
3 6
3 7
0 0 1 0 0 0 0
```

Output 2

```
2
4 5
```

Illustrations

The following diagram illustrates Sample Testcase 1.



The blue arrows denote the nodes a_u which point to the rogue computer from network switch u. The only lab computer that satisfies the constraints is node 4.

9_Palindromes

(20 points)

Time Limit: 1 second
Memory Limit: 256 MB

Statement

Kevin has recently developed an interest in palindromes, which are strings that read the same forwards and backwards. For example, `aba`, `abba`, and `b` are palindromes, while `ba`, `abc`, and `abaa` are not. Kevin's mind is now obsessed with coming up with palindrome-related questions for everything he sees. And true to form, he has received a string consisting of only characters `0` and `1` and immediately thought of a new problem:

"Is it possible to quickly answer queries about the number of **distinct-looking** non-empty palindrome subsequences in a given range?"

Kevin is unsure how to solve this problem, so he seeks your help, knowing your fondness for algorithms. Can you assist him in solving this intriguing problem?

Note: A subsequence of a sequence S refers to a sequence obtained by deleting zero or more elements from S, without changing the order of the remaining elements.

Input Format

The first line contains two positive integers N and Q, representing the length of the binary string Kevin received and the number of queries, respectively.

The second line contains a string of length N consisting of only characters `0` and `1`, denoted as $s_1 s_2 \ldots s_N$, which is the string Kevin received.

The following Q lines each contain two positive integers l_i and r_i, representing that the i-th query wants to know the number of distinct-looking non-empty palindrome subsequences in the given range $[l_i, r_i]$ of the string.

Output Format

Output Q lines, where each line corresponds to the answer for the i-th query. Since the answers can be large, output the remainder of each answer divided by 998244353.

Constraints

- $1 \leq |S| = N \leq 3000$
- $1 \leq Q \leq 3 \times 10^5$
- $s_i \in \{$ `0`, `1` $\}$
- $1 \leq l_i \leq r_i \leq N$

Test Cases

Input 1

```
6 4
011011
1 4
4 6
3 5
1 6
```

Output 1

```
6
3
4
10
```

Illustrations

In Example 1, there are a total of ten distinct-looking palindrome subsequences in the entire string: `0`, `1`, `00`, `11`, `010`, `101`, `111`, `0110`, `1111`, `11011`. This is the answer for the fourth query. Additionally, each query corresponds to finding the respective number of distinct-looking palindrome subsequences within the specified range.

10_Expiring_Tic-Tac-Toe

(20 points)

Time Limit: 1 second
Memory Limit: 256MB

Statement

Alice loves playing Japanese Mahjong and often gathers four people to form a four-player hanchan game. On this particular day, Alice was looking for people to play as usual. After finding three players, she approached Bob to join the game. However, Bob declined Alice's invitation because he was preparing for his final exams. Unwilling to give up when she was just one player short, Alice persistently pestered Bob. Finally, unable to resist Alice's relentless pleading, Bob decided to challenge her to a game. If Alice won, he would agree to play Japanese Mahjong with her.

Bob decided to play a "Expiring Tic-Tac-Toe" game with Alice. The game is similar to regular Tic-Tac-Toe: Alice goes first, and the two players take turns placing their pieces on a 3x3 board. The first player to align their pieces in a row, column, or diagonal wins. However, the twist is that each piece placed by a player will expire after three of that player's turns. That is, any piece placed on turn x will expire on turn $x+3$. Notably, a piece cannot be placed on the same square in turn x and turn $x+3$. Additionally, if the game has not ended by the 45510-th turn, it will be declared a draw. Specially, before determining if any player has won in a certain round, if there are pieces that are about to expire, the pieces will first expire, and then determine if there is a winner.

Alice and Bob, after playing a few turns, got tired and didn't want to spend time thinking through their strategies. Thus, they turned to you for help. Based on the current state of the game, they hope you can determine the outcome if both players continue to play optimally, with Alice making the next move.

Input Format

The input consists of three lines, each containing three integers.
The integer $a_{i,j}$ on the i-th row and j-th column represents the current state of cell of the board on i-th row and j-th column:

- $a_{i,j} > 0$ indicates that the cell contains Alice's piece, which will expire when Alice places her $a_{i,j}$-th piece.
- $a_{i,j} < 0$ indicates that the cell contains Bob's piece, which will expire when Bob places his $-a_{i,j}$-th piece.
- $a_{i,j} = 0$ indicates that the cell is empty.
  It is guaranteed that the provided board does not have a winner yet and that fewer than ten turns have been played.

Output Format

Output a string representing the outcome of the game when both players use optimal strategies:

- "Alice"(with quotation marks) if Alice will win
- "Bob"(with quotation marks) if Bob will win

- "Draw"(with quotation marks) if the game will end in a draw

Constraints

- $-3 \leq a_{i,j} \leq 3$
- Each of the numbers $1, 2, 3, -1, -2, -3$ appears exactly once among all $a_{i,j}$s

Test Cases

Input 1

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

Output 1

```
Alice
```

Input 2

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

Output 2

```
Bob
```

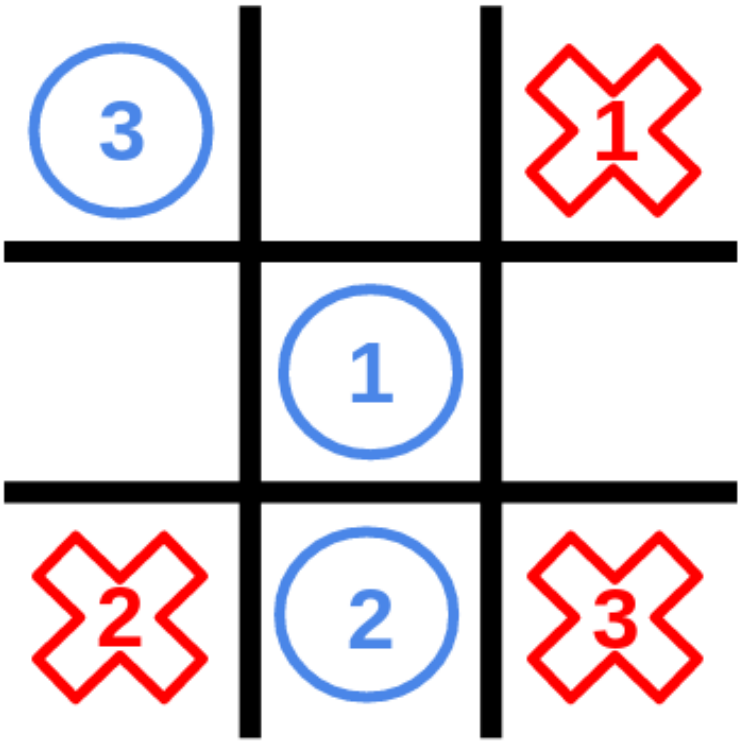
Input 3

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

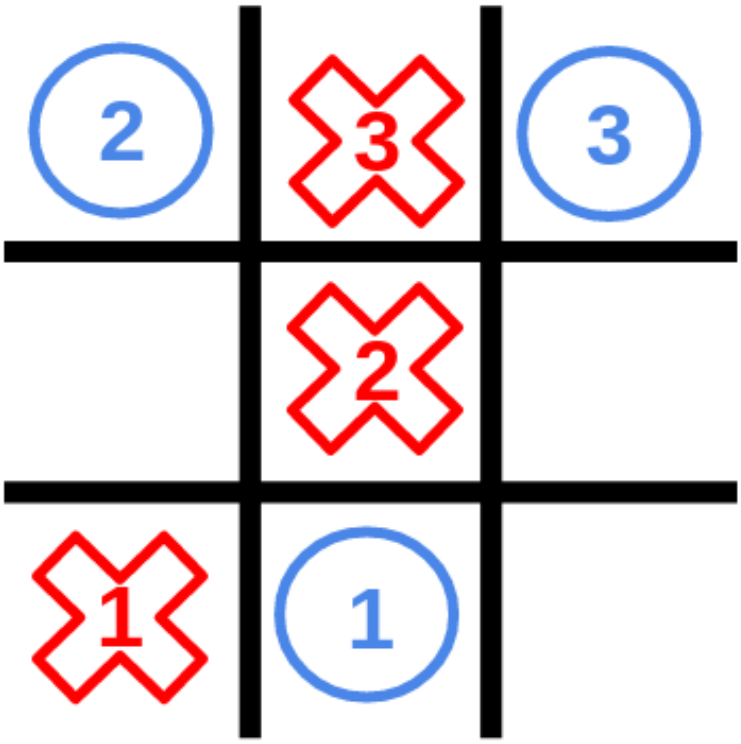
Output 3

```
Draw
```

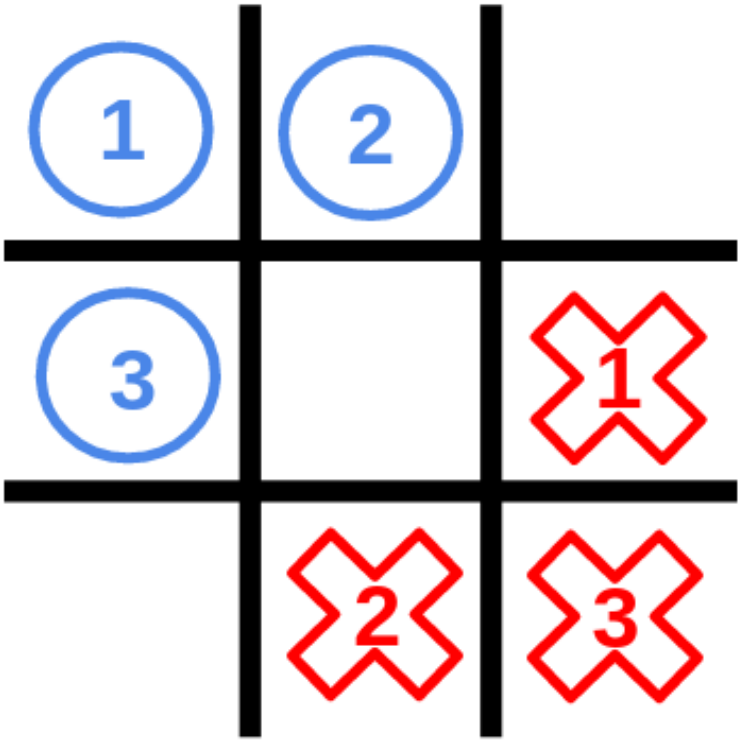
Illustrations



In test case 1, we can see that if Alice places her piece in the top middle position, she will definitely win in the next round by placing her piece in the top left position.



In test case 2, no matter where Alice places her piece, Bob will definitely win in the next round by placing his piece in the bottom middle position.



In test case 3, both players must block each other from winning in the next round, resulting in a draw.

11_Stairs

(6 points/6 points/8 points))

Time Limit: 1 second

Memory Limit: 1024 MB

Statement

NJ has N stairs numbered from 1 to N.

Each stair is composed of a set of pillars, where each pillar has its own height y and its position x.

We loosely refer to the collection of pillars as the "pillar set" of the stair.

Each stair extends infinitely in length.
In this stair, the height at position k is defined as the maximum y value among all the pillars in the pillar set of the stair with $x \leq k$.

Initially, the pillar set of each stair is empty.

NJ wants to manipulate his stairs crazily.
Specifically, NJ will modify the pillar sets of these stairs.
The operations that NJ can perform are as follows:

- $1 \;\; id \;\; x \;\; y$
- $2 \;\; a \;\; b$
- $3 \;\; id \;\; k$

For the first type of operation, NJ will add a new pillar with height y and position x to the pillar set of the stair numbered id.

For the second type of operation, NJ will move all the pillars from the pillar set of stair b to the pillar set of stair a.

For the third type of operation, NJ wants to know the height at position k in the stair numbered id.

Input Format

The first line of the input will contain two positive integers, N and Q.

In the following Q lines, each line will contain one of the operations mentioned in the statement.

Output Format

For each operation of the third type, output the height NJ wants to know, line by line.

If the "pillar set" that NJ is asking is empty, output -1 instead.

Constraints

- $1 \leq N, Q \leq 200\,000$.

- $op \in \{1, 2, 3\}$.
- $1 \leq x, y, k \leq 1\,000\,000\,000$.
- $1 \leq a, b, id \leq N$.
- $a \neq b$.

Subtasks

- Subtask 1 satisfies that $N = 1$.
- Subtask 2 satisfies that $N \leq 200$.
- Subtask 3 has no additional constraints.

Test Cases

Input 1

```
4 7
1 3 424849053 592732474
1 1 787225144 158123046
2 4 3
3 4 534558786
2 2 4
3 2 806668402
3 3 679670154
```

Output 1

```
592732474
592732474
-1
```

Input 2

```
10 13
1 7 205588596 580018058
2 5 7
1 6 511038935 266614169
1 6 486422924 865076365
1 10 445734405 877372041
3 10 843207541
2 4 9
2 7 8
1 1 708876592 402376498
2 9 5
2 2 6
2 6 5
3 6 323976805
```

Output 2

```
877372041
-1
```

Illustrations

In Example 1:

For the first query, the tallest pillar in the fourth pillar set with coordinates less than or equal to 534558786 is the one inserted by Operation 1, with a height of 592732474.

For the second query, the tallest pillar in the second pillar set with coordinates less than or equal to 806668402 is also the one inserted by Operation 1, with a height of 592732474 (at this point, this pillar has been moved to the second pillar set through Operation 5).

For the third query, since there are no pillars in the third pillar set with coordinates less than or equal to 679670154, the answer is -1.

12_Taking_Photos

(4 points/16 points)

Time Limit: 2 seconds

Memory Limit: 1024 MB

Statement

"Waimai~ How should we eat? Let's have waimai~ What kind of waimai should we have? Curry rice~ Curry rice~"

Taking photos is my habit, and takeout is my daily routine. Every day, I hold my camera to document my takeout life. Over time, the lifespan of the camera is coming to an end. In its remaining life, it can only take K photos. Now, it is known that there are N curry rice shops in the city. As a curry rice enthusiast, I want to capture every single curry rice without missing any. The locations of these curry rice shops can be considered as a two-dimensional plane, and the camera can capture a rectangular area each time, where the four corners of the rectangle have integer coordinates. To make the curry rice look more delicious, I plan to take close-up shots, but this will also reduce the coverage area of each photo. Here we define the area of a point as 0. Now, I want to know the minimum total coverage area when I can only take K shots and capture all the curry rice.

Input Format

The first line of the input contains two positive integers N and K, representing the number of curry rice shops and the number of times the camera can take photos.

Next, there are a total of N lines, each containing two positive integers X_i and Y_i, representing the coordinates of the curry rice shops.

Output Format

Output an integer representing the minimum total area covered by the photos.

Constraints

- $1 \leq N \leq 3000$.
- $1 \leq K \leq N$.
- $1 \leq X_i, Y_i \leq 10^7$ ($1 \leq i \leq N$).
- $X_i < X_{i+1}$ and $Y_i < Y_{i+1}$ ($1 \leq i \leq N-1$).

Subtasks

- Subtask 1 satisfies that $N \leq 500$.
- Subtask 2 has no additional constraints.

Test Cases

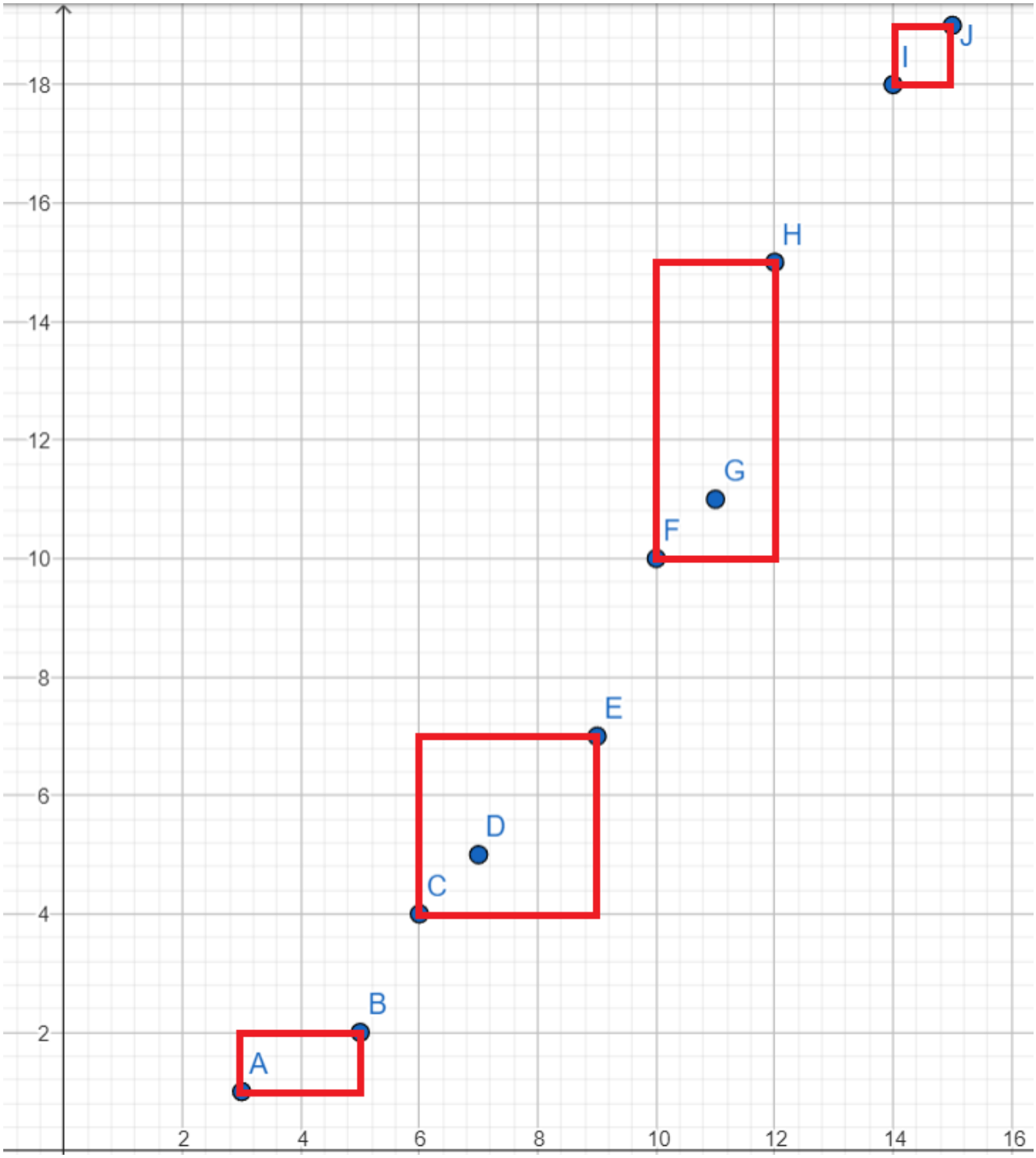
Input 1

```
10 4
3 1
5 2
6 4
7 5
9 7
10 10
11 11
12 15
14 18
15 19
```

Output 1

```
22
```

Illustrations



The minimum coverage is shown in the above image, with total area of
$2 \times 1 + 3 \times 3 + 2 \times 5 + 1 \times 1 = 22$。

13_Sacred_Alliance

(3 points/3 points/14 points)

Time Limit: 5 seconds

Memory Limit: 1024 MB

Statement

In this world, there are N countries. Countries may be connected by roads, if there is a road between country u and country v, we call them neighbors. It should be noted that a country cannot be its own neighbor.

We guarantee that all countries are interconnected by roads.

Among them, there are T countries ruled by monsters, and the remaining $N-T$ countries are composed of humans.

Multiple human countries can unite to form a Sacred Alliance. For a Sacred Alliance S, given a number p, if for any country x in the alliance, it satisfies $\frac{|N(x)\cap S|}{|N(x)|} \geq p$, we call the strength of the Sacred Alliance S as p. Here $N(x)$ represents the set formed by neighboring countries connected to country x.

As a member of the human side, you want to know how strong a Sacred Alliance the humans can form.

Input Format

The first line of input contains two positive integers N, M, T, representing the total number of countries, the number of roads, and the number of countries ruled by monsters, respectively.

The second line of input contains T positive integers $a_1, a_2, \ldots, a_T$, representing the indices of the countries ruled by monsters.

The next M lines each contain two positive integers u_i, v_i, representing there is a road between country u_i and country v_i.

Output Format

If the strength of the strongest Sacred Alliance is $\frac{p}{q}$, where p, q are positive integers and $\gcd(p,q)=1$, please output $p \times q^{-1} \bmod 10^9+7$.

Constraints

- $2 \leq N \leq 10^5$.
- $N-1 \leq M \leq \min(\frac{N(N-1)}{2}, 2\times 10^5)$.
- $0 \leq T < N$.
- $1 \leq u_i, v_i \leq N$ for $i = 1, 2, \ldots, N$.
- $u_i \neq v_i$ for $i = 1, 2, \ldots, N$.

Subtasks

- Subtask 1 satisfies that $N \leq 15$.
- Subtask 2 satisfies that $N \leq 26$.
- Subtask 3 has no additional constraints.

Test Cases

Input 1

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

Output 1

```
500000004
```

Input 2

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

Output 2

```
333333336
```

Input 3

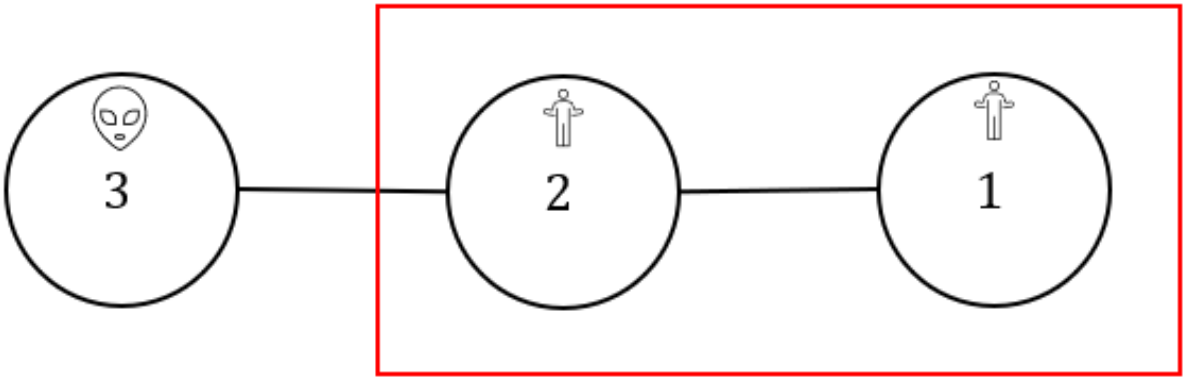
```
6 6 3
1 2 3
1 4
2 4
3 4
4 5
4 6
5 6
```

Output 3

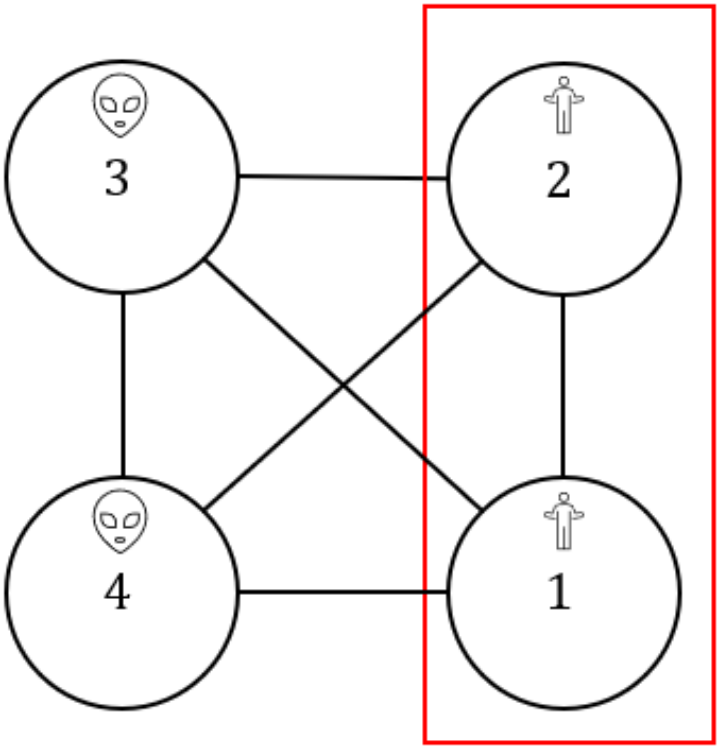
```
500000004
```

Illustrations

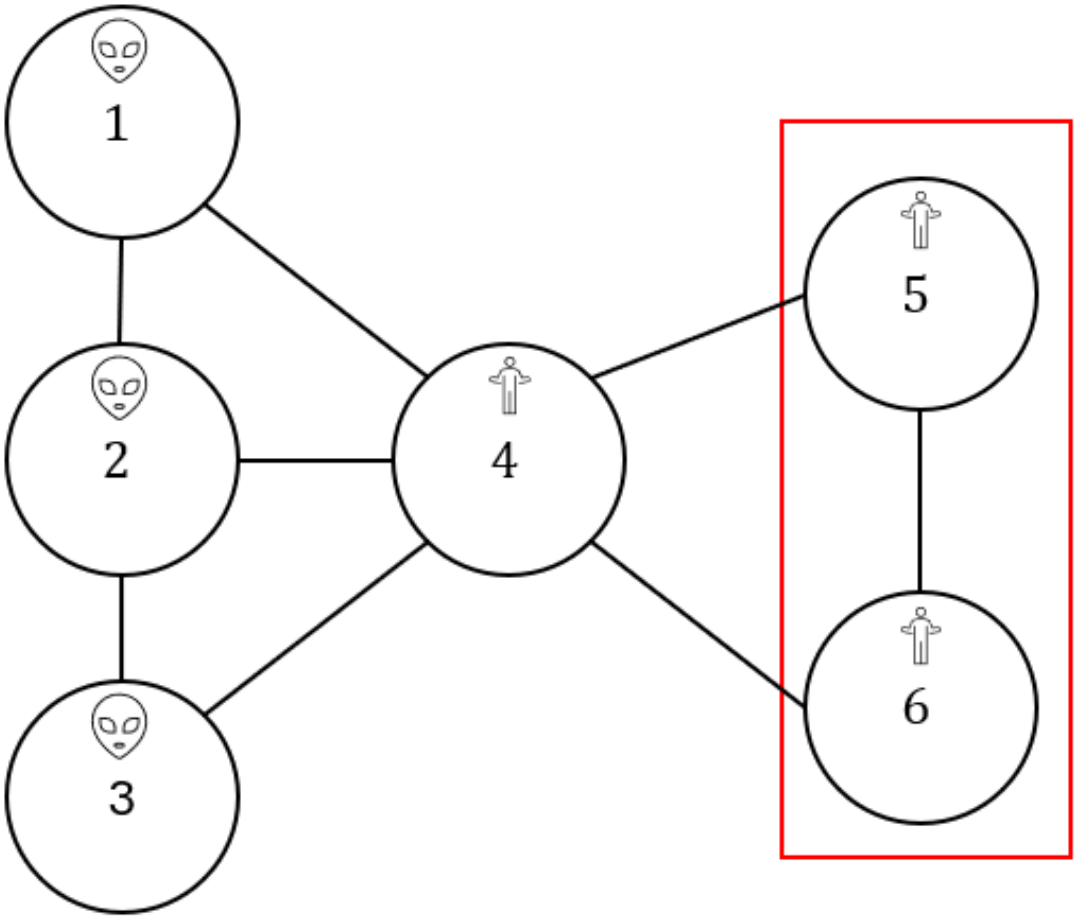
In Example 1, in the Sacred Alliance formed by countries 1 and 2, country 1 has a proportion of $\frac{1}{1}$ of its neighbors in the alliance, while country 2 has a proportion of $\frac{1}{2}$ of its neighbors in the alliance. Therefore, the strength of the alliance is $\frac{1}{2}$.



In Example 2, in the Sacred Alliance formed by countries 1 and 2, both country 1 and country 2 have a proportion of $\frac{1}{3}$ of their neighbors in the alliance. Thus, the strength of the alliance is $\frac{1}{3}$.



In Example 3, in the Sacred Alliance formed by countries 4 and 5, both country 4 and country 5 have a proportion of $\frac{1}{2}$ of their neighbors in the alliance. Therefore, the strength of the alliance is $\frac{1}{2}$.


1
2
3
4
5
6

14_Convex_Hull_Area

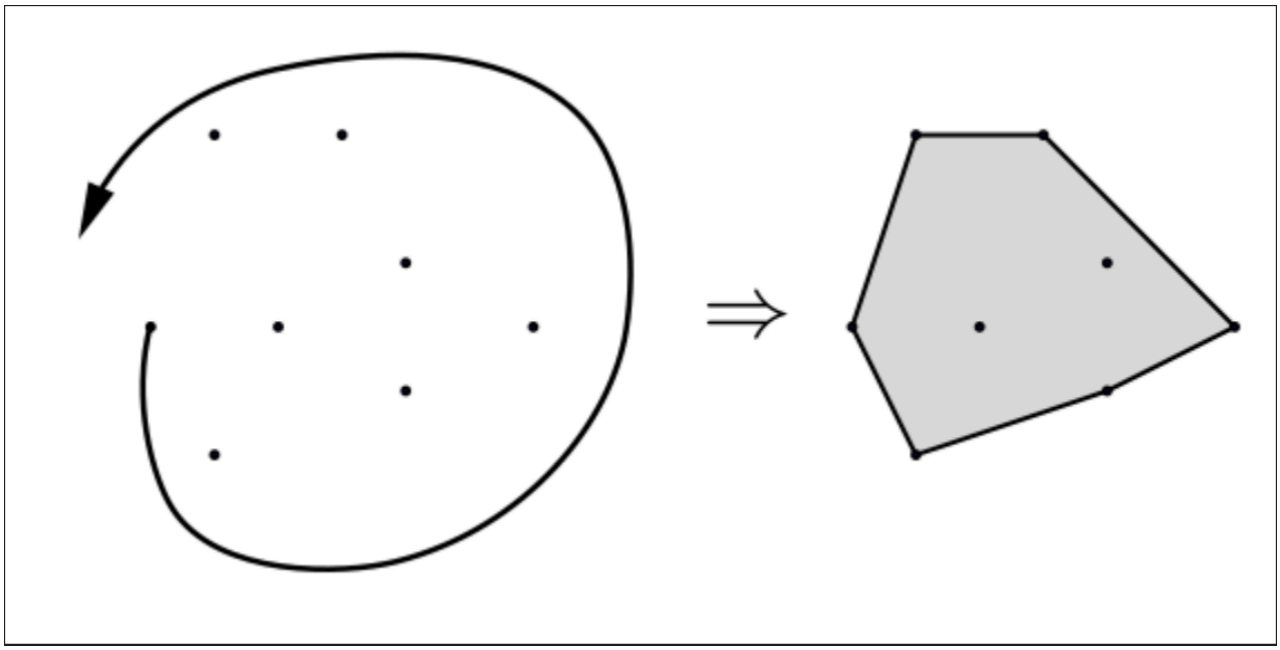
(20 points)

Time Limit: 1 second

Memory Limit: 512 MB

Statement

The **Convex Hull** is a frequently used tool in computational geometry. For a given set of N points in the 2D-plane, we define the convex hull as the minimum area convex polygon that encloses all the points.



Calculating the area of a set of points in a 2D-plane is also a classic problem. However, as someone who enjoys solving algorithmic problems, you certainly won't settle for such a simple task. Now, we randomly select a subset uniformly at random from the 2^N possible subsets of these N points. What is the expected area of the convex hull for this chosen subset? (Consider the convex hull area of an empty subset as 0.)

Input Format

The first line contains an integer N, representing the number of points.

The next N lines contain two integers each, x_i and y_i, representing the coordinates of the i-th point.

Output Format

Output a real number representing the expected area of the covex hull. Your answer will be considered correct if the relative or absolute error compared to the correct answer is within 10^{-6}.

Constraints

- $1 \leq N \leq 300$.

- $|x_i|, |y_i| \leq 10^6$ $(1 \leq i \leq N)$.
- All given points are distinct.

Test Cases

Input 1

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

Output 1

```
0.125000000000000
```

Input 2

```
4
0 0
1 0
2 0
1 1
```

Output 2

```
0.187500000000000
```

Input 3

```
10
44 8
35 12
-46 -27
16 -17
-13 0
-36 -10
-37 -13
-36 10
35 18
-23 -39
```

Output 3

```
1485.140625000000000
```

Illustrations

In Input 1, there are a total of $2^3 = 8$ subsets. Among them, only the subset $\{(0,0),(2,0),(0,1)\}$ can form a convex hull with an area of 1. Since the probability of selecting a particular subset is $\frac{1}{8}$, the expected area of the convex hull is $1 \times \frac{1}{8} = \frac{1}{8}$.

In Input 2, there are a total of $2^4 = 16$ subsets. The subsets $\{(0,0),(1,0),(1,1)\}$, $\{(0,0),(2,0),(1,1)\}$, $\{(1,0),(2,0),(1,1)\}$, and $\{(0,0),(1,0),(2,0),(1,1)\}$ can form convex hulls with areas of $\frac{1}{2}$, 1, $\frac{1}{2}$, and 1 , respectively. Since the probability of selecting a particular subset is $\frac{1}{16}$, the expected area of the convex hull is $(\frac{1}{2}+1+\frac{1}{2}+1) \times \frac{1}{16} = \frac{3}{16}$.

Input 3 is provided as a test case with a slightly larger input size for self-testing purposes.

15_A_Typical_IOI_Problem

(10 points / 10 points)

Time Limit: 2 seconds
Memory Limit: 1024 MB

Statement

LDC, who just became a member of the Argentine IOI team this year, has been troubled recently: IOI is known for difficult math problems, generating functions and polynomial operations are just the basics, and IOI gold medalists are almost always proficient at analysis and algebra! She, who is an expert of data structures, flows, and implementation, has a hard time adapting to IOI problems, so she asked her teacher Frank for an IOI practice problem. The problem statement is as follows:

Frank has N students numbered from 1 to N. The relationships among the N students can be represented by $p_1, p_2, \cdots, p_N$, which is a permutation of $1, 2, \cdots, N$, and t, which is a string of length N and only consists of characters `0` and `1`. An array of length N $a_1, a_2, \cdots, a_n$ is defined as a permutation of $1, 2, \cdots, n$ if each of the numbers $1, 2, \cdots, n$ appears exactly once in array a. For example, $N = 3, p = [3, 1, 2], t =$ `010` is a valid relationships.

They are about to have an German exam, so Frank prepared N test papers to be distributed by his assistant, LDC. Each student should receive one test paper. Since there are too many students, she distributes the test papers using the following steps:

1. If she runs out of test papers (i.e., every student has received a test paper), the process ends.
2. She hands over all the test papers she has to the student with the smallest number who hasn't received one yet.
3. She drinks a glass of milk.
4. She waits until a student raises their hand. When a student raises his/her hand, she takes back the test paper from that student and goes back to step 1.

When student i receives x test papers (whether distributed by LDC or passed on by other students), student i performs the following operations:

1. Keeps one test paper for himself/herself.
2. If at least one of the following three conditions holds, student i raises their hand and returns $x - 1$ test papers to LDC (once LDC receives the test papers, student i puts his/her hand down), otherwise, student i passes $x - 1$ test papers to student p_i.
    - Student p_i already has a test paper.
    - $x = 1$
    - $t_i =$ `1`

However, LDC has limited knowledge of the students' relationships. The information she has can be represented by a string s of length N, where each character is `0`, `1`, or `?`. For all integers i satisfying $1 \leq i \leq N$, if $s_i \neq$ `?`, LDC knows that $t_i = s_i$, otherwise, she doesn't know what t_i is. LDC has no information about the permutation $p_1, p_2, \cdots, p_N$.

Given $p_1, p_2, \cdots, p_N$ as a uniformly random permutation of $1, 2, \cdots, N$, and for all integers i satisfying $1 \leq i \leq N$ with $s_i = $ `?`, there is a $\frac{1}{2}$ probability that $t_i = $ `1` and a $\frac{1}{2}$ probability that $t_i = $ `0`. Assuming all choices are independent events, what is the expected number of glasses of milk LDC drinks after distributing the test papers? Since LDC does not like decimals, please output the answer modulo 998244353.

As LDC is not good at math, she can't solve this problem herself. Please help her by writing a program that can solve this problem.

If a rational number $\frac{Q}{P}$ satisfies that both P and Q are integers, $\gcd(P, Q) = 1$, and P is not divisible by 998244353, the value of $\frac{Q}{P}$ modulo 998244353 is defined as the unique integer R satisfying $0 \leq R < 998244353$ and $RP \equiv Q \pmod{998244353}$.

It can be proven that the expected value is a rational number which can be expressed in its simplest form $\frac{Q}{P}$, where the denominator P is not divisible by 998244353.

Input Format

The first line of the input contains an integer T, which means there are T test cases.

The format of each test case is as follows:

The first line of the test case contains an integer N.

The second line of the test case contains a string s of length N, where each character is `0`, `1`, or `?`.

Output Format

For each test case output one line. That line should have only one number, which is the expected number of glasses of milk LDC drinks.

Constraints

- $1 \leq T \leq 5000$
- $1 \leq N \leq 5000$
- $|s| = N$
- In a single test file, the sum of N is at most 5000

Subtasks

- Subtask 1 satisfies that in a single test file, the sum of N is at most 400 (10 points).
- Subtask 2 no other constraints (10 points).

Test Cases

Input 1

```
2
2
?1
8
100?0?1?
```

Output 1

```
249561090
522444260
```

Illustrations

For the first test case, there are two possibilities for t: `01` and `11`, and two possibilities for p: $\{1, 2\}$ and $\{2, 1\}$.

If $t =$ `01` and $p = \{1, 2\}$, the process of distributing the test papers is as follows:

1. LDC distributes 2 test papers to student 1.
2. LDC drinks a glass of milk.
3. Student 1 keeps 1 test paper for himself/herself.
4. Since student $p_1 = 1$ has already received a test paper, student 1 raises his/her hand and returns 1 test paper to LDC.
5. LDC distributes 1 test paper to student 2.
6. LDC drinks a glass of milk.
7. Student 2 keeps 1 test paper for himself/herself.
8. Since student $p_2 = 2$ has already received a test paper, student 2 raises his/her hand and returns 0 test papers to LDC.
9. Since LDC has no test papers left, the process ends.

In this case, LDC drinks 2 glasses of milk.

If $t =$ `01` and $p = \{2, 1\}$, LDC would drink 1 glass of milk.
If $t =$ `11` and $p = \{1, 2\}$, LDC would drink 2 glasses of milk.
If $t =$ `11` and $p = \{2, 1\}$, LDC would drink 2 glasses of milk.

Each of these cases has a probability of $\frac{1}{4}$, so the expected number of glasses of milk LDC drinks is $\frac{2+1+2+2}{4} = \frac{7}{4}$. Since $249561090 \times 4 \equiv 7 \pmod{998244353}$, the output should be 249561090.

16_Omelet_Loves_Wading

(3 points/10 points/12 points)

Time Limit: 6 seconds

Memory Limit: 512 MB

Statement

"How come it's suddenly raining heavily? Poor Taipei."

Taipei Basin often experiences heavy rain, which forces Omelet, who loves eating ramen everywhere, to wade through the water. However, Omelet, with a passion for ramen, doesn't find it troublesome to go to distant ramen restaurants in the rain. In fact, he considers it an interesting challenge and enjoys it. As a result, he has developed the ability to walk without getting wet or soaking his shoes, even on typhoon days. With this skill, Omelet never misses the annual Enjoy-Rain Competition. The organizers of the competition, noticing Omelet's exceptional abilities, are concerned that he may overshadow other participants and leave them with no chance to perform. Therefore, they hope to lower the difficulty level of the competition.

The competition venue consists of N squares and $N-1$ bidirectional roads, with each road connecting two squares. It is guaranteed that from any square, one can reach any other square by traversing these roads. The organizers need to choose K squares in the venue and set up "shelters" in these squares. In this competition, each participant will have a random starting position and the goal is to reach any shelter as quickly as possible in heavy rain. Therefore, the difficulty of a square is defined as the minimum number of roads one needs to traverse from that square to a square with a shelter. The overall difficulty of the competition is the maximum difficulty among all squares.

The organizers want to know the minimum difficulty of the competition for each $K = 1, 2, \ldots, N$.

Input Format

The first line contains an integer N, representing the number of squares.

In the next $N-1$ lines, the i-th one of them contains two integers u_i and v_i, representing that the i-th road connects squares u_i and v_i.

Output Format

Output N lines, where the i-th line contains an integer representing the minimum difficulty of the competition when $K = i$.

Constraints

- $1 \leq N \leq 3 \times 10^5$.
- $1 \leq u_i, v_i \leq N$ $(1 \leq i \leq N-1)$.
- It is guaranteed that any two squares are connected by some roads.

Subtasks

- Subtask 1 satisfies that $N \leq 3000$.
- Subtask 2 satisfies that $N \leq 50000$.
- Subtask 3 has no additional constraints.

Test Cases

Input 1

```
9
1 2
1 3
3 4
3 5
4 6
4 7
7 8
7 9
```

Output 1

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

Input 2

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

Output 2

```
2
1
1
1
0
```

Input 3

```
16
5 3
16 11
10 11
14 16
13 15
9 13
9 12
6 7
4 6
1 8
8 10
15 2
4 3
1 2
5 1
```

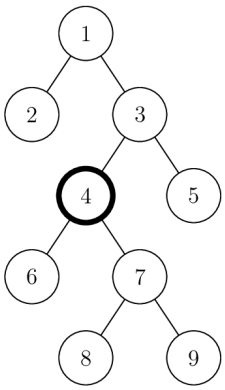
Output 3

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

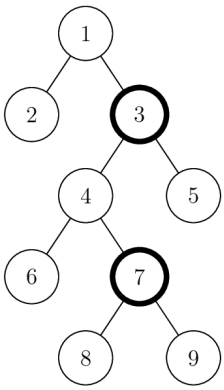
Illustrations

The following figure illustrates one possible configuration of shelters for squares in Example 1, where K ranges from 1 to 4, with the aim of minimizing the difficulty of the competition. The bold borders indicate squares where shelters should be set up.

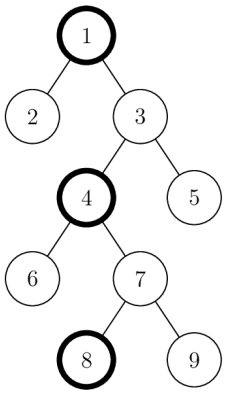
$K = 1, ans = 3$



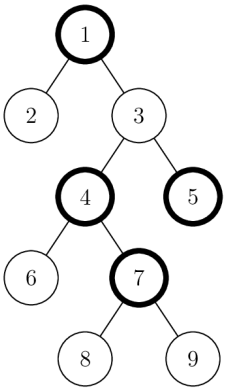
$K = 2, ans = 2$



$K = 3, ans = 2$



$K = 4, ans = 1$



17_Yuna_and_Mithril

(8 points / 17 points)

Time Limit: 4 seconds
Memory Limit: 512 MB

Statement

In a different world, Mithril is the most precious material, and tools crafted from it are incredibly sharp. To ensure that many players have access to Mithril tools, the efforts of miners and blacksmith shops are required. In this world, there are a total of n blacksmith shops, numbered from 0 to $n-1$. However, due to the high level of skill required for Mithril processing, only k of these shops are capable of working with Mithril. Every morning, miners dig several pieces of raw Mithril ore from the mine and transport them to the blacksmith shops at noon for processing and sale in the afternoon. Since Mithril is a highly valuable material, in order to avoid conflicts among the shops over the Mithril ore, for each piece of Mithril ore, they select one of the k blacksmith shops capable of working with Mithril with equal probability and allocate the ore to that shop. Additionally, due to the high demand for Mithril, all the ore mined in the morning will be sold out on the same day.

Yuna is an adventurer living in this different world, and she wants to make a Mithril dagger for her partner, Fina, to help her with her dissection work. However, Yuna doesn't know which blacksmith shops are capable of working with Mithril. Therefore, she first selects one of the blacksmith shops with equal probability as her starting point. If she is currently at the i-th blacksmith shop, she follows the process below:

- If the blacksmith shop does not have Mithril, she moves to the $(i+1) \bmod n$-th blacksmith shop and continues the process.
- Otherwise, she has the Mithril dagger made at that blacksmith shop and happily returns home.

Now that you know which k shops can work with Mithril, but the number of Mithril ore mined by the miners each morning is unknown, with only the information that they can mine up to m pieces of ore in a day. You want to output, for each i from 1 to m, the expected number of shops Yuna needs to visit in order to obtain the Mithril dagger if the miners dug up i pieces of Mithril ore in the morning. Since this number could be a fraction, if the number is $\frac{P}{Q}$, please output $PQ^{-1} \pmod{998244353}$, where Q^{-1} satisfies $QQ^{-1} \equiv 1 \pmod{998244353}$.

Input Format

The first line of input will contain three positive integers, n, m, and k, representing the number of blacksmith shops, the upper limit on the number of ore mined per day, and the number of blacksmith shops capable of working with Mithril, respectively.

The second line of input will contain k strictly increasing integers, $a_1, a_2, \ldots, a_k$, representing the indices of the blacksmith shops capable of working with Mithril.

Output Format

Please output a single line containing m numbers, where the i-th number represents the expected number of shops Yuna needs to visit in order to successfully obtain the Mithril dagger if the miners dug up i pieces of Mithril ore.

Constraints

- $1 \leq n < 998244353$
- $1 \leq m, k \leq 10^5$
- $k \leq n$
- $0 \leq a_1 < a_2 < \ldots < a_k < n$

Subtasks

- Subtask 1 satisfies that $m, k \leq 2000$. (8 points)
- Subtask 2 has no additional constraint. (17 points)

Test Cases

Input 1

```
6 6 2
1 5
```

Output 1

```
499122180 831870297 499122179 332748120 748683267 457528664
```

Input 2

```
864197532 1 1
565656562
```

Output 2

```
931220943
```

Input 3

```
20 10 5
0 3 7 14 19
```

Output 3

```
499122187 411276681 367353928 350583422 846511216 333512644 47264079 475033985 232520342
449759810
```

Illustrations

In Example Testcase 1:

- For $i = 1$, there is a $\frac{1}{2}$ probability that the first blacksmith shop has mithril, and a $\frac{1}{2}$ probability that the fifth blacksmith shop has mithril. Therefore, the answer is $\frac{1}{2} \times \frac{2+1+6+5+4+3}{6} + \frac{1}{2} \times \frac{5+4+3+2+1+6}{6} = \frac{7}{2}$. Since $2^{-1} \equiv 499122177 \pmod{998244353}$, we need to output $7 \times 499122177 \equiv 499122180 \pmod{998244353}$.
- For $i = 2$, there is a $\frac{1}{4}$ probability that only the first blacksmith shop has mithril, a $\frac{1}{4}$ probability that only the fifth blacksmith shop has mithril, and a $\frac{1}{2}$ probability that both the first and fifth blacksmith shops have mithril. Therefore, the answer is $\frac{1}{4} \times \frac{2+1+6+5+4+3}{6} + \frac{1}{4} \times \frac{5+4+3+2+1+6}{6} + \frac{1}{2} \times \frac{2+1+4+3+2+1}{6} = \frac{17}{6}$. Since $6^{-1} \equiv 166374059 \pmod{998244353}$, we need to output $17 \times 166374059 \equiv 831870297 \pmod{998244353}$.
- For $i = 3$, the answer is $\frac{5}{2}$.
- For $i = 4$, the answer is $\frac{7}{3}$.
- For $i = 5$, the answer is $\frac{9}{4}$.
- For $i = 6$, the answer is $\frac{53}{24}$.

In Example Testcase 2:

- For $i = 1$, the answer is $\frac{864197533}{2}$.

18_Nachoneko_and_Beautiful_Clothes

(1 point/1 point/8 points/15 points)

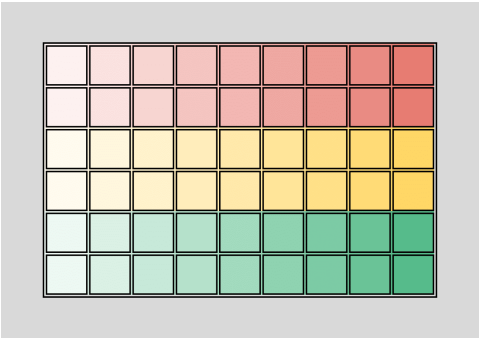
Time Limit: 5 seconds

Memory Limit: 1024 MB

Statement

Nachoneko mama is making clothes for her adorable children, and the first step is to find suitable colors. She takes out a fabric swatch card and plans to cut some fabric from it for experimentation.

The fabric swatch card can be imagined as a rectangle with a height of h centimeters and a width of w centimeters.
For ease of communication, we divide it into h rows and w columns.
We label the 1×1 squares with coordinates (r, c), where r represents the row number from top to bottom, and c represents the column number from left to right.



A fabric swatch card with dimensions `(h, w) = (6, 9)`.
The red square in the top right corner is labeled as `(1, 9)`, and the white-green corner in the bottom left corner is labeled as `(6, 1)`.

Now, Nachoneko wants to cut out n rectangles with a height of 1 from the fabric swatch card.
Each rectangle can be represented by three parameters $(r[i], c[i][0], c[i][1])$, which indicate the coordinates of the two endpoints of the rectangle: $(r[i], c[i][0])$ and $(r[i], c[i][1])$.
Being mindful of the fabric, she wants to avoid cutting outside these n rectangles (for which she only needs the rectangles where r is odd).
However, within these rectangles, the fabric can be cut into any number of pieces.

Nachoneko has given you three kinds of operation that can be performed. Can you tell her how to perform the operations so that all of it can be utilized?

1. `combine` $c[0]$ $c[1]$: Compress column $c[0]$ to column $c[1]$ together in each row, while keeping the labeling of the fabric intact.
2. `cut` r c: Cut out all the fabric that overlaps with the fabric labeled as (r, c).
3. `spread`: Unfold the entire fabric swatch card. You **must** and can **only** perform this operation once, and it must be your final operation.

Due to Nachoneko's limited physical stamina, if you make her perform more than $q_{\max} = 300\,000$ operations, she will start to roll on the floor and be unable to give you the AC verdict.

Input Format

- line 1: $\texttt{subtask}$: Indicating the subtask this testcase belongs to.
- line 2: $h \;\; w \;\; n$
- line $3+i$ ($0 \le i \le n-1$): $r[i] \;\; c[i][0] \;\; c[i][1]$

Output Format

- line 1: q
- line $2+i$ ($0 \le i \le q-1$): $\text{Operation}[i]$
- Your output value for q must satisfy $1 \le q \le q_{\max}$.
- The format and limitations for each **operation** are as follows:
  1. $\texttt{combine} \;\; c[0] \;\; c[1]$
     - Your output values must satisfy $1 \le c[0] \le c[1] \le w$.
  2. $\texttt{cut} \;\; r \;\; c$
     - Your output values must satisfy $1 \le r \le h$ and $1 \le c \le w$.
  3. $\texttt{spread}$
     - You **must** perform this operation exactly once (even if you performed zero $\texttt{combine}$ operations), and it must be your last operation.
- Please print a newline character after each operation, and **avoid adding extra spaces at the end of each line**, or you might get Wrong Answer.

Constraints

- $1 \le \texttt{subtask} \le 3$.
- $q_{\max} = 300\,000$.
- $1 \le h, w \le 120\,000$.
- $1 \le n \le 500$.
- $1 \le r[i] \le h$ ($0 \le i \le n-1$).
- $r[i]$ is odd ($0 \le i \le n-1$).
- $1 \le c[i][0] \le c[i][1] \le w$ ($0 \le i \le n-1$).
- All the inputs are integers.

Subtasks

- Subtask 1 satisfies that $h \le 100, w \le 100$.
- Subtask 2 satisfies that $c[i-1][1] < c[i][0]$ ($1 \le i \le n-1$).
- Subtask 3 satisfies that $n \le 100$.
- Subtask 4 has no other constraints.

In each subtask you can obtain a partial score.
Let $q_1, q_2, \ldots, q_k$ be the values of q that you output for each test case in the subtask,
$q'_1, q'_2, \ldots, q'_k$ be the theoretical minimum number of operations for each test case in the subtask,
and let $Q = \max\{q_1, q_2, \ldots, q_k\}$.
Your score for each subtask is calculated according to the following table:

Operation Count	Score for subtask 1 and 2
$6001 \leq Q \leq 300\,000$	0.5
$Q \leq 6000$	1

Operation Count	Score for subtask 3
$2001 \leq Q \leq 300\,000$	1
$Q \leq 2000$ and there exist i such that $q_i > q'_i$	3
$Q \leq 2000$ and for each i there are $q_i \leq q'_i$	8

Operation Count	Score for subtask 4
$180\,001 \leq Q \leq 300\,000$	2
$30\,001 \leq Q \leq 180\,000$	4
$10\,001 \leq Q \leq 30\,000$	6
$Q \leq 10\,000$ and there exist i such that $q_i > q'_i$	10
$Q \leq 10\,000$ and for each i there are $q_i \leq q'_i$	15

Test Cases

Input 1

```
1
10 10 5
1 3 7
3 4 5
5 10 10
7 1 10
9 3 9
```

If we don't consider the variable **`subtask`**, this sample input satisfies the constraints of Subtasks 1, 3, 4.

Output 1

```
10
combine 4 5
cut 3 5
combine 3 7
cut 1 5
cut 5 10
combine 6 9
cut 9 8
combine 1 10
cut 7 1
spread
```

It can be proven that there is no solution with $q \leq 9$.

Input 2

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

If we don't consider the variable **`subtask`**, this sample input satisfies the constraints of Subtasks 1, 3, 4.

Output 2

```
3
combine 1 5
cut 1 3
spread
```

It can be proven that there is no solution with $q \leq 2$.

Input 3

```
2
1 100 3
1 1 2
1 3 98
1 99 100
```

Output 3

```
3
combine 1 100
cut 1 1
spread
```

A test case may meet part of subtasks

Input

```
2
1 100 3
1 1 2
1 3 98
1 99 100
```

If we don't consider the variable `subtask`, this sample input satisfies the constraints of Subtasks 2, 3, 4.

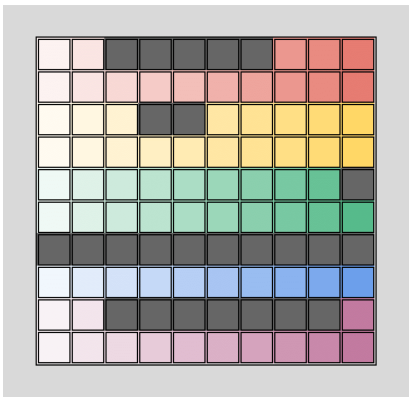
Output

```
10
combine 1 2
cut 1 1
cut 1 2
combine 3 98
combine 31 41
cut 1 3
combine 99 100
cut 1 99
cut 1 1
spread
```

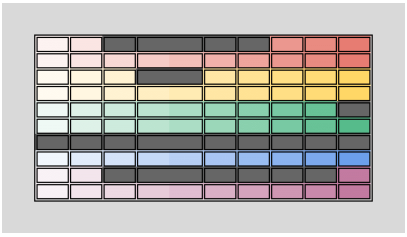
The optimal solution has $q = 3$. The output looks like this will only receive partial scores in Subtasks 3 and 4.

Illustrations

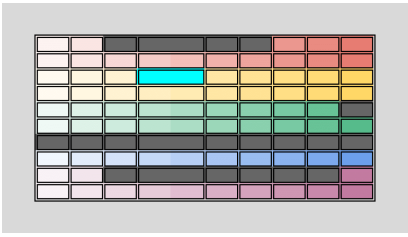
The illustration for the first example is as follows.
The gray areas represent all the fabric to be cut,
the light blue areas represents the fabric currently being cut by the `cut` operation,
the bright green areas represent the fabric that has already been cut,
and the disappearing separator lines indicate that the corresponding columns have been compressed together by the `combine` operation.



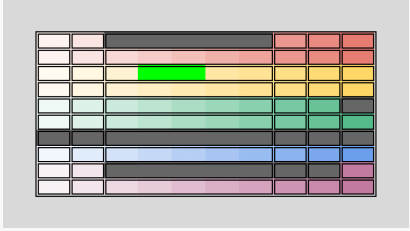
Initial State



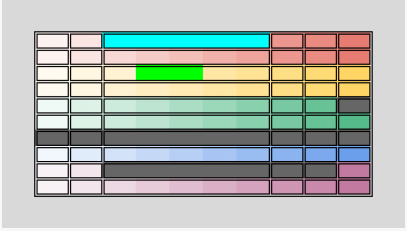
`combine 4 5`



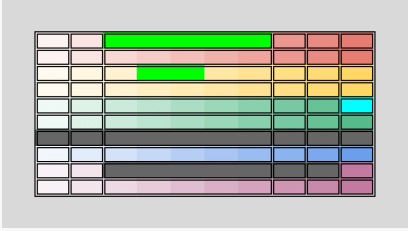
`cut 3 5`



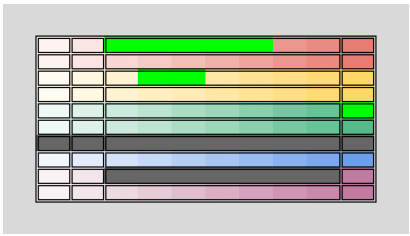
`combine 3 7`



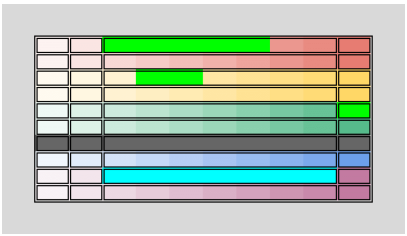
`cut 1 5`



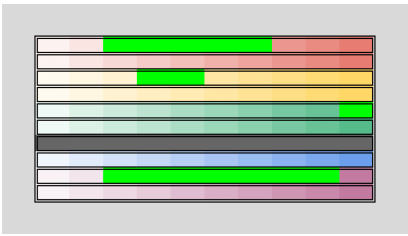
`cut 5 10`



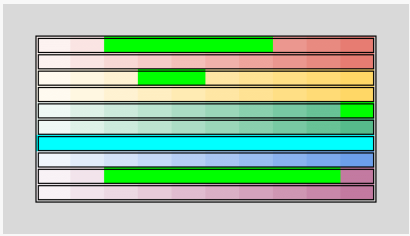
`combine 6 9`



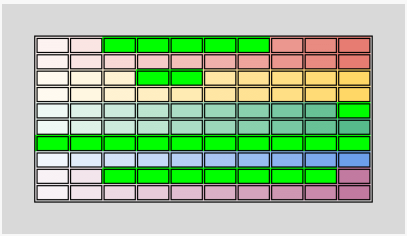
`cut 9 8`



`combine 1 10`



`cut 7 1`

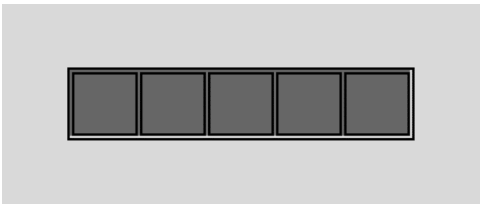


`spread`

For the sixth operation, `combine` 6 9, since columns 3 to 7 have already been compressed, this operation is equivalent to `combine` 3 9.

Please remember that the last operation **must** be `spread`.

The illustration for the second example is as follows. The gray areas represent all the fabric to be cut.



Even if a position is specified in the input multiple times (e.g., $(1, 3)$), you only need to perform one `cut` operation at that position.

In the third example, even if it is not necessary, you can still cut the same position multiple times, and you can also compress intervals that have already been compressed before.

Hint: Partial scores for Subtask 4 are easier to achieve compared to Subtask 3.

Afterword

Thanks to your help, Nachoneko mama successfully made the perfect clothes for the children, and P-Cat fulfilled her wish to wear the cutest dress. It's truly a cause for celebration!

19_Interesting_Tree_Structure

(2 points/10 points/3 points/4 points/6 points)

Time Limit: 5 seconds

Memory Limit: 1024 MB

Statement

One day after school, Rikki found the cat Raana sleeping on a tree in the school courtyard. To track Ranna's elusive whereabouts, Rikki used the matcha parfait tactic to lure her. Ranna told Rikki that she only sleeps on *interesting tree*s and plays with kittens near trees that are *almost broken*, and the rules for distinguishing these trees are as follows:

1. There are $2k$ tree holes on this tree, numbered $1, 2, \ldots, 2k$.
2. Any two different tree holes u and v may not have an edge between them, or there may be an one-way edge, but there will be no duplicate edges, nor will there be edges in both directions.
3. Raana will give k requirements in the form of "There is a path from tree hole a_i to tree hole b_i".
4. For all tree hole relationship graphs that satisfy the first three rules, the tree with the fewest number of edges used (let the number be r) is an **interesting tree**.
5. Among the trees with r edges used, as long as they do not satisfy all first three rules, they are **almost broken tree**s.
6. All trees not with r edges used are **boring tree**s.

Tomori and Rikki found the tree hole relationships $E = \{(u_j, v_j)\}_{j=1}^{m}$ in the school courtyard, but they couldn't solve this problem, so they called out the programmer Neko-chan for help.

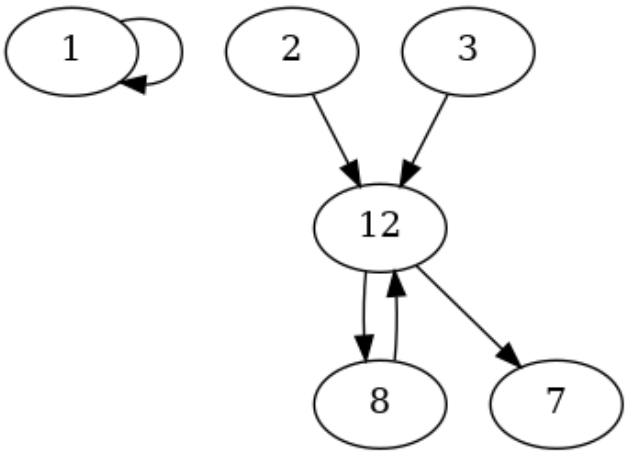
~~Not surprisingly, Neko-chan handed the problem to you again.~~ Now, given Raana's requirements and the tree hole relationship graph, please help Rikki classify these trees.

There are t test cases.

P.S. The tree holes form a directed graph. Although the tree holes are on the tree, this graph is not necessarily a tree.

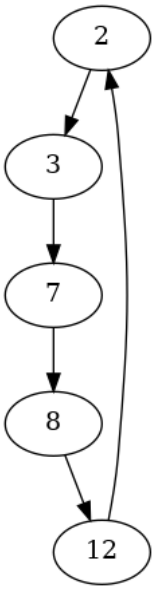
Please refer to the next page for examples.

For example, let $k = 6$ and $\{(a_i, b_i)\}_{i=1}^{k} = \{(1, 1), (2, 12), (12, 7), (3, 12), (12, 8), (8, 12)\}$. The requirement graph is as follows:

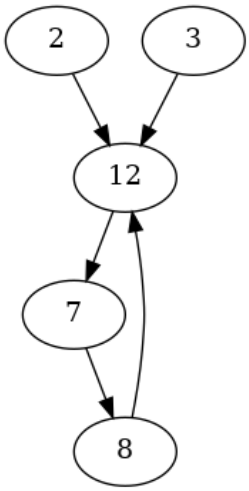


Requirement graph

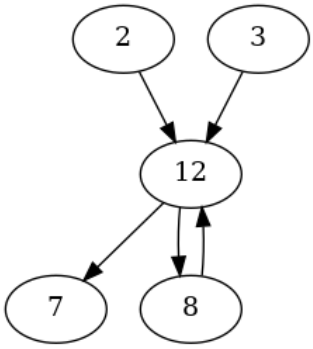
Consider the following different tree hole relationship graphs:



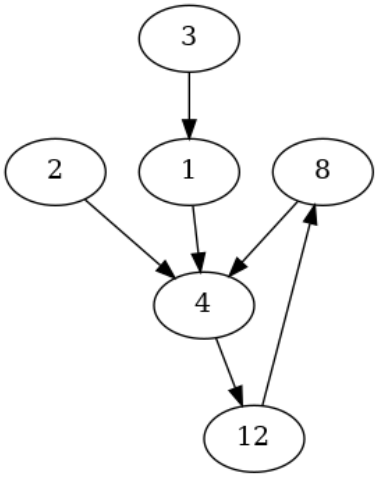
1st tree



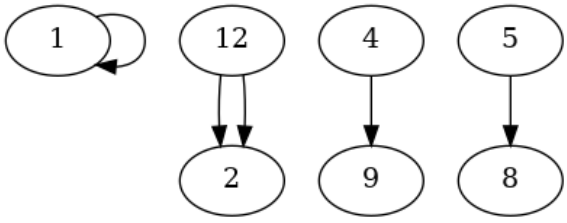
2nd tree



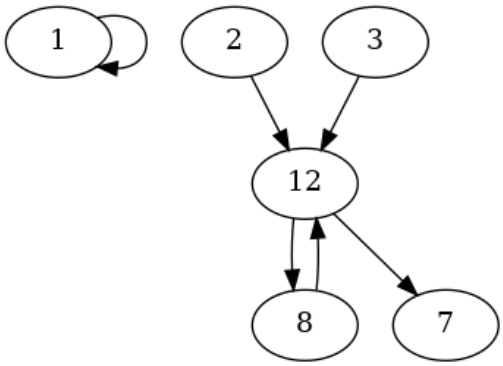
3rd tree



4th tree



5th tree



6^{th} tree

	1^{st} **tree**	2^{nd} **tree**	3^{rd} **tree**	4^{th} **tree**	5^{th} **tree**	6^{th} **tree**
Satisfies second rule	True	True	False	True	False	False
Satisfies all Raana's requirements	True	True	True	False	False	True
# of edges	5	5	5	6	5	6
Type of tree	Interesting	Interesting	Almost broken	Boring	Almost broken	Boring

Input Format

- line 1: t
- block c ($1 \leq c \leq t$): $\mathrm{TestCase}_c$

Where $\mathrm{TestCase}_c$ is in the following format:

- line 1: k
- line $1+i$ ($1 \leq i \leq k$): a_i b_i
- line $k+2$: m
- line $k+2+j$ ($1 \leq j \leq m$): u_j v_j

Output Format

- line c ($1 \leq c \leq t$): Answer_c

Where Answer_c is in the following format:

- Interesting tree: `Omoshiree`
- Almost broken tree: `Kowaresou`
- Boring tree: `Tsumannee`

Constraints

- $1 \leq t \leq 10$.
- $1 \leq k \leq 100\,000$.
- $1 \leq a_i, b_i \leq 2k$ ($1 \leq i \leq m$).
- $1 \leq m \leq 100\,000$.
- $1 \leq u_j, v_j \leq 2k$ ($1 \leq j \leq m$).
- All numbers in the input are integers.

Subtasks

- Subtask 1 satisfies that $a_i < b_i$ ($1 \leq i \leq k$); $u_j = v_j = 1$ ($1 \leq j \leq m$).
- Subtask 2 satisfies that $u_j = v_j = 1$ ($1 \leq j \leq m$).
- Subtask 3 satisfies that $k \leq 800$; $m \leq 800$.
- Subtask 4 satisfies that $k \leq 20\,000$; $m \leq 20\,000$.
- Subtask 5 satisfies has no additional constraints.

Test Cases

Input 1

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

This sample input satisfies the constraints of Subtasks 3, 4, 5.

Output 1

```
Omoshiree
Omoshiree
```

Input 2

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

This sample input satisfies the constraints of Subtasks 3, 4, 5.

Output 2

```
Tsumannee
Kowaresou
Kowaresou
```