Problem A. SQRT Problem

Input file: `standard input`
Output file: `standard output`

Miss Burger has three positive integers n, a, and b. She wants to find a positive integer solution x ($1 \le x \le n-1$) that satisfies the following two conditions:

- $x^2 \equiv a \pmod n$
- $\lfloor \sqrt[3]{x^2} \rfloor = b$

Additionally, it is guaranteed that n is an odd number and $\gcd(a, n) = 1$. Here $\gcd(x, y)$ denotes the greatest common divisor of x and y. We also guarantee that there exists a unique solution.

Note that $\lfloor x \rfloor$ represents the largest integer not exceeding x, such as $\lfloor 0.5 \rfloor = 0$, $\lfloor 11.3 \rfloor = 11$, $\lfloor 101.9 \rfloor = 101$, $\lfloor 99 \rfloor = 99$, $\lfloor 0 \rfloor = 0$, $\lfloor 2 \rfloor = 2$.

Input

The first line contains a single integer n ($3 \le n \le 10^{100} - 1$).

The second line contains a single integer a ($1 \le a \le n-1$).

The third line contains a single integer b ($1 \le b \le n-1$).

Output

Output a single integer denoting the solution x.

Examples

standard input	standard output
9 4 3	7
650849 253233 5059	359895
2926865854037163912204616967760553 8931 22216978925831646928504047924228222624 922652112396383261277016 2	28025732380501848167087889769592298758

Problem B. Queue Sorting

Input file: `standard input`
Output file: `standard output`

Miss Burger has learned a lot of data structures recently. Her favorite one is the queue. Now, she wants to use two queues to implement a sorting algorithm.

To do so, she creates two queues A and B in the beginning. They are all initialized to be empty. For a queue, you can only conduct two kinds of operations:

- Insert: inserting an element to the back of the queue.
- Eject: acquire and remove the first element in the front.

The integers we will sort only include 1 to n. For each i ($1 \leq i \leq n$), we let a_i denote the amount of i in total. Therefore, $\sum_{i=1}^{n} a_i$ integers are to be sorted in total, denoted as m.

Miss Burger first rearranges all m integers into a sequence b in arbitrary order. We say b is legal if and only if she can sort all integers in non-decreasing order in the following way:

- For each integer in b in order, she inserts it into one of the two queues.
- After all integers are inserted, she begins to eject them out one by one in arbitrary order. Here, each time, she can choose to eject an integer from one of the non-empty queues and concatenate it into the resulting sequence.

Now, Miss Burger wants to know how many different sequences $\{b_i\}_{i=1\ldots m}$ can be sorted by her two queues. Two sequences $\{b_i\}_{i=1\ldots m}$ are considered different if there exists a position i where their values b_i of these two sequences are different. As the answer may be large, your output should be modulo 998 244 353.

Input

The first line contains an integer n ($1 \leq n \leq 500$), denoting the types of numbers.

The second line contains n integers, $a_1, \ldots, a_n$ ($0 \leq a_i \leq n$), where a_i denotes the amount of number i.

It is guaranteed that $1 \leq \sum_{i=1}^{n} a_i \leq 500$.

Output

Output a single line containing only one integer, denoting the number of legal sequences.

As the answer may be large, your output should be modulo 998 244 353.

Example

standard input	standard output
4 1 1 1 1	14

Problem C. Cyclic Substrings

Input file:	`standard input`
Output file:	`standard output`

Mr. Ham is interested in strings, especially palindromic strings. Today, he finds a string s of length n.

For the string s of length n, he defines its *cyclic substring* from the i-th character to the j-th character ($1 \le i, j \le n$) as follows:

- If $i \le j$, the cyclic substring is the substring of s from the i-th character to the j-th character. He denotes it as $s[i..j]$.
- If $i > j$, the cyclic substring is $s[i..n] + s[1..j]$, where $+$ denotes the concatenation of two strings. He also denotes it as $s[i..j]$.

For example, if $s = \texttt{12345}$, then $s[2..4] = \texttt{234}$, $s[4..2] = \texttt{4512}$, and $s[3..3] = \texttt{3}$.

A string t is *palindromic* if $t[i] = t[n-i+1]$ for all i from 1 to n. For example, `1221` is palindromic, while `123` is not.

Given the string s, there will be many cyclic substrings of s which are palindromic. Denote P as the set of all **distinct** cyclic substrings of s which are palindromic, $f(t)(t \in P)$ as the number of times t appears in s as a cyclic substring, and $g(t)(t \in P)$ as the length of t. Mr. Ham wants you to compute

$$\sum_{t \in P} f(t)^2 \times g(t)$$

The answer may be very large, so you only need to output the answer modulo 998 244 353.

Input

The first line contains a number n ($1 \le n \le 3 \times 10^6$), the length of the string s.

The second line contains a string s of length n. Each character of s is a digit.

Output

Output a single integer, denoting the sum modulo 998 244 353.

Examples

standard input	standard output
5 01010	39
8 66776677	192

Note

In the sample, the palindromic cyclic substrings of s are:

- $s[1..1] = s[3..3] = s[5..5] = \texttt{0}$.
- $s[2..2] = s[4..4] = \texttt{1}$.
- $s[5..1] = \texttt{00}$.
- $s[1..3] = s[3..5] = \texttt{010}$.
- $s[2..4] = \texttt{101}$.

- $s[4..2] =$ 1001.
- $s[1..5] =$ 01010.

The answer is $3^2 \times 1 + 2^2 \times 1 + 1^2 \times 2 + 2^2 \times 3 + 1^2 \times 3 + 1^2 \times 4 + 1^2 \times 5 = 39$.

Problem D. Balanced Array

Input file: `standard input`
Output file: `standard output`

Mr. Ham likes balance. He applies the concept of balance to integer arrays.

A *balanced array* is defined as an integer array $a_1, a_2, \ldots a_l$ that satisfies the following condition:

- There exists an integer k, such that $1 \le k \le \frac{l-1}{2}$.
- $a_i + a_{i+2k} = 2a_{i+k}$ for each i in $1, 2, \ldots l - 2k$.

Given an array $a_1, a_2, \ldots a_n$, Mr. Ham wants to determine whether $a_{1 \ldots i}$ is a balanced array for each i in $1, 2, \ldots n$.

Please help Mr. Ham to solve the task.

Input

The first line contains an integer n ($1 \le n \le 2 \times 10^6$), denoting the length of the array A.

The second line contains n integers $a_1, a_2 \ldots a_n$ ($1 \le a_i \le 2 \times 10^8$).

To minimize the size of the input file, a_i **was encoded in base-62**, where the characters `0`...`9a`...`zA`...`Z` correspond to the numerical values $0 \ldots 61$ for each digit. For example, `Aa0` represents $36 \times 62^2 + 10 \times 62 + 0 = 139004$.

Output

Output a binary string $s_{1 \ldots n}$, such that $s_i = 1$ if $a_{1 \ldots i}$ is balanced, $s_i = 0$ otherwise.

Examples

standard input	standard output
3 1 2 3	001
9 1 2 3 2 5 4 3 8 5	001010111
9 1C 3f 4S 3h 88 6x 4W d1 8c	001010111

Problem E. Matrix Distances

Input file: `standard input`
Output file: `standard output`

Mr. Ham has a matrix of size $n \times m$, where each cell is filled with a color value $c_{i,j}$. Mr. Ham is interested in the relationships between cells of the same color and wants to calculate the sum of Manhattan distances between all pairs of cells with the same color.

The Manhattan distance between two cells (x_1, y_1) and (x_2, y_2) is given by $d_M = |x_1 - x_2| + |y_1 - y_2|$.

Formally, please compute:

$$\text{Total Sum} = \sum_{C} \sum_{(x_i,y_i)\in S_C} \sum_{(x_j,y_j)\in S_C} |x_i - x_j| + |y_i - y_j|$$

Here, C denotes the set of all distinct colors. S_C denotes the set of coordinates (x, y) with the color value equal to the specified color C.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 1000$), representing the number of rows and columns in the matrix.

The following contains n lines. Each line i contains m space-separated integers $c_{i,j}$ ($1 \leq c_{i,j} \leq 10^9$), representing the color values in the matrix.

Output

Output a single integer, the sum of Manhattan distances for all pairs of cells with the same color.

Examples

standard input	standard output
2 2 1 1 2 2	4
4 4 1 3 2 4 2 1 2 3 1 3 3 2 3 2 1 4	152

Note

In the first example, the distinct color values are 1 and 2.

1. For color 1:
   - The coordinates with color 1 are $(1, 1)$ and $(1, 2)$.
   - The Manhattan distance between $(1, 1)$ and $(1, 1)$ is $|1 - 1| + |1 - 1| = 0$.
   - The Manhattan distance between $(1, 1)$ and $(1, 2)$ is $|1 - 1| + |1 - 2| = 1$.
   - The Manhattan distance between $(1, 2)$ and $(1, 1)$ is $|1 - 1| + |2 - 1| = 1$.
   - The Manhattan distance between $(1, 2)$ and $(1, 2)$ is $|1 - 1| + |2 - 2| = 0$.
2. For color 2:
   - The coordinates with color 1 are $(2, 1)$ and $(2, 2)$.

- The Manhattan distance between $(2, 1)$ and $(2, 1)$ is $|2 - 2| + |1 - 1| = 0$.
- The Manhattan distance between $(2, 1)$ and $(2, 2)$ is $|2 - 2| + |1 - 2| = 1$.
- The Manhattan distance between $(2, 2)$ and $(2, 1)$ is $|2 - 2| + |2 - 1| = 1$.
- The Manhattan distance between $(2, 2)$ and $(2, 2)$ is $|2 - 2| + |2 - 2| = 0$.

So, the total sum of Manhattan distances for all pairs of cells with the same color is $1 + 1 + 1 + 1 = 4$.

Problem F. Colorful Balloons

Input file: `standard input`
Output file: `standard output`

Miss Burger is participating in the 2023 ICPC Asia Hefei Regional Contest. She wants to quickly solve the first blood problem by finding the easiest problem.

Miss Burger has observed the colors of all the balloons prepared by the volunteers in advance (It's not allowed, so please don't do that). If the number of balloons of a certain color is greater than 50% of the total number of balloons, then the problem represented by that color is considered the easiest problem. Each color is represented as a string consisting only of lowercase letters.

Now Miss Burger has provided you with the colors of all the balloons she saw and wants you to tell her which color represents the easiest problem.

Input

The first line contains an integer n ($1 \le n \le 10^5$), representing the total number of balloons Miss Burger saw.

The following n lines contain a string s_i ($1 \le |s_i| \le 10$), representing the color of a balloon, which consists only of lowercase letters.

Output

If the easiest problem is determined, output the color of the easiest problem.

If no the easiest problem can be determined, output "`uh-oh`" (without quotation marks).

Examples

standard input	standard output
5 red green red red blue	red
3 red blue yellow	uh-oh

Note

In the first example, Miss Burger saw 5 balloons, with colors "red", "green", "red", "red", and "blue". The problem represented by the color "red" appears 3 times, which is more than 50% of the total number of balloons, making it the easiest problem. Therefore, the output is "red".

Problem G. Streak Manipulation

Input file: `standard input`
Output file: `standard output`

This semester, Mr. Ham spent a lot of time in training for ICPC. He has n classes this semester, and he only attended some of them. He uses a binary string $s_{1..n}$ to represent which classes he attended. If the i-th character of the string is `1`, he attended the i-th class. Otherwise, he didn't attend the i-th class.

If Mr. Ham attended k consecutive classes, he would get a *streak* of length k. Formally, if $1 \le i \le j \le n$ satisfies the following conditions, we say that Mr. Ham has a streak of length $j - i + 1$:

- $s_i = s_{i+1} = \cdots = s_j = \texttt{1}$;
- $i = 1$ or $s_{i-1} = \texttt{0}$;
- $j = n$ or $s_{j+1} = \texttt{0}$.

For example, if $s = \texttt{101101}$, Mr. Ham has one streak of length 2 and two streaks of length 1.

Mr. Ham found that he was absent from too many classes. So he stole the attendance record and wants to change it (It's not allowed, so please don't do that). Given m and k, he can change **at most** m records from `0` to `1`. He wants to know the **maximum** length of the k-th longest streak he can get.

If there are less than k streaks, we define the length of k-th longest streak as -1.

Input

The first line contains three integers n, m and k ($1 \le m \le n \le 2 \times 10^5$, $1 \le k \le \min(n, 5)$).

The second line contains a string s of length n. It is guaranteed that $\forall 1 \le i \le n, s_i \in \{\texttt{0}, \texttt{1}\}$.

Output

Output the maximum length of the k-th longest streak Mr. Ham can get by changing **at most** m records from `0` to `1`.

Examples

standard input	standard output
8 3 2 10110110	3
12 3 3 100100010011	2
4 4 4 0000	-1

Problem H. Computational Complexity

Input file: `standard input`
Output file: `standard output`

Mr. Ham learned about computational complexity in the algorithm course. Let $T(n)$ be the time the algorithm takes to run on input size n. For example, for the merge sort algorithm, we have the following recursion equation,

$$T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + O(n).$$

And we can get the upper bound $T(n) = O(n \log n)$ from the algorithm textbook.

Mr. Ham is a good kid who loves to learn and explore, so he decided to try a harder problem. Consider two algorithms $A_1(n)$ and $A_2(n)$ that call each other. They satisfy the following calling relationship:

$$A_1(n) \text{ calls } A_2\left(\left\lfloor \frac{n}{2} \right\rfloor\right), A_2\left(\left\lfloor \frac{n}{3} \right\rfloor\right), A_2\left(\left\lfloor \frac{n}{5} \right\rfloor\right) \text{ and } A_2\left(\left\lfloor \frac{n}{7} \right\rfloor\right),$$

$$A_2(n) \text{ calls } A_1\left(\left\lfloor \frac{n}{2} \right\rfloor\right), A_1\left(\left\lfloor \frac{n}{3} \right\rfloor\right), A_1\left(\left\lfloor \frac{n}{4} \right\rfloor\right) \text{ and } A_1\left(\left\lfloor \frac{n}{5} \right\rfloor\right),$$

Mr. Ham wants to know the precise time taken by both algorithms.

The problem can be formally stated as follows:

Let $f(n)$ be the number of operations required by $A_1(n)$, and $g(n)$ be the number of operations required by $A_2(n)$. They satisfy the following recursion relationship

$$f(n) = \max\left(n, g\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + g\left(\left\lfloor \frac{n}{3} \right\rfloor\right) + g\left(\left\lfloor \frac{n}{5} \right\rfloor\right) + g\left(\left\lfloor \frac{n}{7} \right\rfloor\right)\right),$$

$$g(n) = \max\left(n, f\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + f\left(\left\lfloor \frac{n}{3} \right\rfloor\right) + f\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + f\left(\left\lfloor \frac{n}{5} \right\rfloor\right)\right).$$

Given the values of $f(0), g(0)$ and m, Mr. Ham wants to know what $f(m)$ and $g(m)$ are, and the result is **modulo** M.

Note that $\lfloor x \rfloor$ represents the largest integer not exceeding x, such as $\lfloor 0.5 \rfloor = 0$, $\lfloor 11.3 \rfloor = 11$, $\lfloor 101.9 \rfloor = 101$, $\lfloor 99 \rfloor = 99$, $\lfloor 0 \rfloor = 0$, $\lfloor 2 \rfloor = 2$.

Input

The first line contains four numbers, namely $f(0)$, $g(0)$, T, M $(0 \le f(0), g(0), T \le 10^5, 2 \le M \le 10^{15})$,

Each of the next T lines contains a integer m $(0 \le m \le 10^{15})$ querying the values of $f(m)$ modulo M and $g(m)$ modulo M.

Output

Output T lines, each line contains two numbers $f(m)$ modulo M and $g(m)$ modulo M, separated by spaces.

Examples

standard input	standard output
1958 920 10 100000000000 0 1 2 3 10 100 200 1000 19580920 20232023	1958 920 3680 7832 10592 9554 17504 11276 50294 64826 784112 893714 1894550 1905470 12057866 12979424 71481494756 48626708512 28127864908 7251681354
0 0 10 100000000000 0 1 2 3 4 10 20 30 40 100	0 0 1 1 2 2 3 3 4 4 11 12 25 26 41 41 55 58 162 172

Problem I. Linguistics Puzzle

Input file: `standard input`
Output file: `standard output`

When preparing for the International Collegiate Linguistics Contest, Mr. Ham meets an unknown language X. He is given an integer n ($2 \le n \le 52$) and n^2 numbers written in Language X. The n^2 numbers are generated by the following rules:

- Generate a sequence $a_0, a_1, \ldots, a_{n^2-1}$ that satisfies $a_{n \cdot i+j} = i \cdot j$ for all $0 \le i, j < n$.
- Shuffle the sequence.

Mr. Ham is an experienced ICLCer. He finds out some basic rules of Language X:

- There are n different symbols in Language X. Mr. Ham uses the first n lowercase letters to represent them if $n \le 26$. Otherwise, he uses the first 26 lowercase letters and the first $n-26$ uppercase letters to represent them.
- The numbers in Language X are written in base n. Each digit is represented by a symbol in Language X.
- Like in Arabic numerals, the digits are written from the most significant digit to the least significant digit, i.e. $\texttt{a} \cdot n + \texttt{b}$ is written as `ab` instead of `ba`. There are no leading zeros, i.e. `a` is written as `a` instead of `0a`.

Mr. Ham wants to know which symbol represents digit i in Language X for each $0 \le i < n$. He asks you for help.

Input

Each test contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 50$). The description of the test cases follows.

The first line of each test case contains an integer n ($2 \le n \le 52$), the number of symbols in Language X.

The second line contains n^2 strings $s_1, s_2, \ldots, s_{n^2}$, the numbers in Language X. Each string consists of at most 2 lowercase and uppercase letters.

It is guaranteed that the answer exists.

Output

Output a string of length n, the i-th character is the symbol that represents digit $i-1$ in Language X.

If there are multiple answers, output any of them.

Examples

standard input	standard output
2 3 a b a b b b b c cc 4 d d d d d c b a d b cd cb d a cb bc	bca dcba
2 4 d a a bc ba bc b a a a d a a cb c c 4 a b da b b d ad b db b a c da b c b	abcd bdac

Note

In the first test case of the first sample, the letter `b` represents digit 0, the letter `c` represents digit 1, and the letter `a` represents digit 2. The numbers given in the input are $1, 0, 1, 0, 0, 0, 0, 2, 4$.

Problem J. Takeout Delivering

Input file: `standard input`
Output file: `standard output`

Mr. Ham is a very diligent hamster in USTC, and he always uses his weekends to deliver takeout to earn more money.

To maximize his delivery efficiency, he wants to write a program to find the shortest route.

Specifically, we can consider the city Hefei where Mr. Ham is located as an undirected graph with n vertices and m edges. Each edge has a congestion level w. Mr. Ham starts at vertex 1 and wants to deliver the takeout to vertex n. Mr. Ham has found that the delivery time is always determined by the two edges with the highest congestion level along the path. Therefore, Mr. Ham defines the length of a path as the sum of the congestion levels of the two edges with the highest congestion level among the paths. When there is only one edge in the path, the length is defined as the congestion level of that edge.

Now, Mr. Ham wants to know the minimum length of the path from vertex 1 to vertex n. Since Mr. Ham doesn't know how to program, he has entrusted this task to you.

Input

The first line contains two integers n ($2 \le n \le 3 \times 10^5$) and m ($\max(1, n-1) \le m \le 10^6$), denoting the number of vertices and edges.

The next m lines contain the edges of the graph, one edge per line. The i-th line contains three integers u_i, v_i, and w_i ($1 \le u_i, v_i \le n, u_i \ne v_i, 1 \le w_i \le 10^9$), indicating an edge (u, v) which the congestion level is w.

There are no self-loops or multiple edges in the given graph and it is guaranteed that the given graph is connected.

Output

Output a single line containing only one integer, indicating the minimum length of a path from vertices 1 to vertices n.

Example

standard input	standard output
4 6 1 2 2 1 3 4 1 4 7 2 3 1 2 4 3 3 4 9	5

Problem K. Campus Partition

Input file: `standard input`
Output file: `standard output`

Mr. Ham is one of the most famous architects in Hefei and has been invited to participate in the planning of the new campus of USTC.

Specifically, the campus of USTC can be considered as a tree $T = (V, E)$, with n buildings and $n - 1$ undirected edges, building i having an importance value w_i. It is guaranteed that the USTC campus is connected.

The university hopes that Mr. Ham can divide the campus into several regions. We consider a partition plan is legal if and only if it satisfies the following conditions:

- Each building belongs to exactly one region.
- The induced subgraph of any region is connected.

The university considers the weight of a region to be the second highest importance value among all buildings in that region, and if the region contains only one building, its weight is 0. The university hopes that Mr. Ham can maximize the sum weights of all regions. Can you help Mr. Ham get the answer?

Induced subgraph: If a region is composed of buildings from the set $S \subseteq V$, then its induced subgraph contains all the buildings in the set S and edges whose endpoints are both in the set S.

Input

The first line contains a positive integer $n(1 \le n \le 5 \times 10^5)$ indicating the number of buildings.

The second line contains n positive integers $w_i(1 \le w_i \le 10^9)$ indicating the weight of each building.

The next $n-1$ lines contain the edges of the USTC campus, one edge per line. The i-th line contains two integers u_i, $v_i(1 \le u_i, v_i \le n, u_i \ne v_i)$ indicating an edge (u, v). It is guaranteed that it forms a tree.

Output

Output a single line containing only one integer indicating the maximum sum weight of all regions.

Example

standard input	standard output
8 2 5 4 5 3 1 1 3 1 2 1 3 1 4 2 5 2 6 3 7 4 8	8

Problem L. Information Spread

Input file: `standard input`
Output file: `standard output`

In Mr. Ham's class, there are n students numbered from 1 to n. One day, a student 1 learns a piece of information. Subsequently, the students initiate the process of spreading the information to each other.

The relationships among the students are represented by a **directed** graph with n vertices and m edges. Each edge has a weight w, a real number between 0 and 1 (inclusive). The process of information spreading is carried out according to the following pseudocode:

Algorithm 1 SPREAD

```
Let aware[1..n] be a new array initialized as False
Let visited[1..n] be a new array initialized as False
procedure DFS(u)
    if visited[u] then
        return
    end if
    visited[u] ← True
    for (u, v, w) ∈ edges starting from u do
        ▷ Enumerate edges in the order of input ◁
        if aware[u] and not aware[v] then
            with probability w, aware[v] ← True
        end if
        DFS(v)
    end for
end procedure
procedure SPREAD
    aware[1] ← True        ▷ The first student knows the information at the beginning
    DFS(1)
end procedure
```

Please compute the probability that student i becomes aware of the information through this process, for all $1 \le i \le n$. In other words, calculate the probability of $aware[u]$ being ***True*** in the above pseudocode.

Input

The first line contains two integers n and m ($3 \le n \le 10^5$, $n-1 \le m \le 3 \cdot 10^5$), denoting the number of students and the number of relationships.

The next m lines each contains four integers u_i, v_i, p_i and q_i ($1 \le a_i, b_i \le n$, $0 \le p_i \le q_i \le 10^5$, $q_i \ne 0$), denoting a relationship from student u_i to student v_i with a weight $w_i = \frac{p_i}{q_i}$.

It is guaranteed that there is no relationship from student i to student i ($1 \le i \le n$), and there is at most one relationship from student i to student j ($1 \le i, j \le n$). It is also guaranteed that all students can be reached from student 1 in the process.

Output

Output n lines, the i-th line contains a single integer x_i denoting the probability that student i becomes aware of the information after the process modulo 998 244 353.

Formally, it can be proven that the answer is a rational number $\frac{p}{q}$. To avoid issues related to precisions, please output the integer ($pq^{-1} \bmod M$) as the answer, where $M = 998\,244\,353$ and q^{-1} is the integer satisfying $qq^{-1} \equiv 1 \pmod{M}$.

Examples

standard input	standard output
4 4 1 2 1 2 2 3 1 2 2 4 1 2 4 3 1 1	1 499122177 623902721 748683265
6 12 1 2 81804 95651 2 3 39701 95895 2 4 6178 17992 3 5 72756 84510 5 6 40007 83640 2 6 60491 92219 5 3 37590 47735 4 5 6867 20289 4 3 75051 93231 6 5 48102 54448 6 1 40190 45274 1 5 37010 60312	1 947252499 124986918 535320090 929273289 551177734

Note

For the first example, the process unfolds as follows:

- Student 1 knows the information initially.
- We choose the edge $(1, 2, \frac{1}{2})$. As a result, student 2 becomes aware of the information with a probability of $\frac{1}{2}$.
- We choose the edge $(2, 3, \frac{1}{2})$.
- We choose the edge $(2, 4, \frac{1}{2})$. Consequently, student 4 becomes aware of the information with a probability of $\frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$.
- We choose the edge $(4, 3, 1)$.

Now, let's analyze the scenario where student 3 remains unaware of the information. This can happen in two cases:

- Student 2 did not become aware when we selected the edge $(1, 2, \frac{1}{2})$.
- Student 2 became aware when we selected the edge $(1, 2, \frac{1}{2})$, but student 3 did not become aware when we selected the edge $(2, 3, \frac{1}{2})$, and student 4 did not become aware when we selected the edge $(2, 4, \frac{1}{2})$.

Therefore, the probability of student 3 becoming aware is given by: $1 - \frac{1}{2} - \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = \frac{3}{8}$.