Problem A. Roller Coaster

- Time Limit: 2 sec

Problem Statement

In Japan Amusement Group (JAG), members discuss how to have better amusement to attract many people. These days, they are interested in reducing waiting time stress.

As a member of JAG, you found out the hypothesis that knowing waiting time can reduce such kind of stress. Therefore, you decided to write a program which presumes the waiting time of a roller coaster.

N groups stand in line for the roller coaster, and the groups are numbered from 1 to N. The group i has a_i people. People in line ride the roller coaster in ascending order of group number.

The first roller coaster departs at time 0 and departs every minute thereafter. The roller coaster can hold up to M people.

For each group, the whole group member must ride the roller coaster at the same time. Additionally, there is no need to get exactly M people on the roller coaster at one time. Each group wants to ride the roller coaster as soon as possible, so they ride it if they can.

You should output N lines. In the i-th line, you should output the time the group i can ride the roller coaster.

Input

$$N \ M$$
$$a_1 \ a_2 \ \ldots \ a_N$$

The first line consists of an integer N between 1 and $100{,}000$, and an integer M between 1 and 10^9, inclusive. N represents the number of groups, and M represents the capacity of the roller coaster.

The second line consists of N integers between 1 and M, inclusive. For each i $(1 \leq i \leq N)$, a_i represents the number of people in the group i.

Output

Output N lines. In the i-th line, you should output the answer for the group i.

Sample Input 1

```
3 5
2 4 1
```

Sample Output 1

```
0
1
1
```

Sample Input 2

```
2 1000000000
1000000000 1000000000
```

Sample Output 2

```
0
1
```

Problem B. Break a Prison

- Time Limit: 2 sec

Problem Statement

Jennifer is a software engineer at a Tech company. Her company decided to join ICPC (Inter-Company Prison breaking Contest) and she was chosen as a representative of the company.

In ICPC, every participant needs to escape from a prison. The prison can be represented as an $n \times m$ grid i.e. it has n rows and m columns of rooms. The room in the i-th row and j-th column in the prison is denoted as room (i, j). Two rooms (i_1, j_1) and (i_2, j_2) are adjacent if and only if $|i_2 - i_1| + |j_2 - j_1| = 1$. Weirdly, there is an unlocked door between each pair of adjacent rooms. Some rooms in the prison are under surveillance. Participants can move to a room only if it's not under surveillance. A participant will start from a room. The goal of all participants is to reach an exit. It's guaranteed that the room with the exit and the room that participants start from are not under surveillance.

To show talents in the company, the CEO asked Jeniffer not to turn right during the contest. In other words, there should **not** be any two consecutive moves between rooms that fulfill the following condition.

Condition: Given that Jeniffer moved from room (i_1, j_1) to (i_2, j_2), and then she moved to room (i_3, j_3). Then, $(i_2 - i_1) \times (j_3 - j_2) - (j_2 - j_1) \times (i_3 - i_2) = -1$ holds.

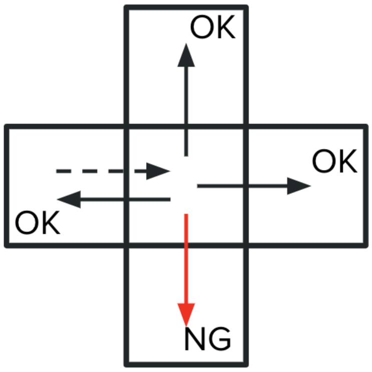

Figure B.1. Example of allowed and denied moves

For example, in figure B.1., if the last move is along the dashed arrow, you cannot move downward but you can move the other three directions.

Note that U-turns are allowed with this condition.

As a Jeniffer's colleague, your mission is to write a program to find the minimum number of moves between rooms to reach the exit for her.

Input

The input consists of a single test case in the following format.

$$
\begin{array}{l}
n\ \ m \\
c_{1,1}c_{1,2}\cdots c_{1,m} \\
c_{2,1}c_{2,2}\cdots c_{2,m} \\
\vdots \\
c_{n,1}c_{n,2}\cdots c_{n,m}
\end{array}
$$

n and m represent the size of the prison, each of which is an integer between 2 and 500. $c_{i,j}\,(1 \le i \le n, 1 \le j \le m)$ is a character that describes the status of a room in the i-th row and j-th column. The character is either

- ‘S' which means a start room for a participant,
- ‘E' which means a room with an exit,
- ‘.' which means the room is not under surveillance, or
- ‘#' which means the room is under surveillance.

It is guaranteed that ‘S' and ‘E' appear exactly once in the input respectively.

Output

Print the minimum number of moves between rooms for Jenniffer to reach the exit. If she cannot reach the exit, print -1.

Sample Input 1

```
2 4
S..#
..E.
```

Sample Output 1

```
3
```

Sample Input 2

```
2 4
S..#
##E.
```

Sample Output 2

```
-1
```

Sample Input 3

```
2 4
S...
##E.
```

Sample Output 3

```
5
```

In Sample Input 3, one of the optimal routes is below.

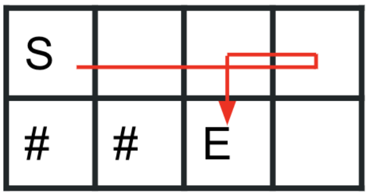

Figure B.2. The optimal route in Sample Input 3

Problem C. Camp room assignment

- Time Limit: 3 sec

Problem Statement

In the JAG country, there are a total of m universities, and we plan to invite $2n$ students to a training camp. Each student is affiliated with one of the m universities. During the training camp, the students will be accommodated in n twin rooms, meaning that each room will be assigned to exactly two students.

To promote diverse interactions among the students, our goal is to achieve a "good room assignment". A room assignment is considered good if and only if the students sharing the same room come from different universities.

Here, we are wondering how often a good room assignment is feasible. There are m^{2n} possible sequences of universities to which each student belongs, and please find for how many of them there is a good room assignment.

Actually, we don't yet know how many rooms we can provide. Therefore, for each of $n = 1, 2, \ldots, m$, please find for how many of the sequences of universities there is a good room assignment.

Since the answer may be huge, print the answers modulo $998{,}244{,}353$.

Input

The input is a single line containing an integer m between 1 and $200{,}000$, inclusive.

Output

Output m lines. In the i-th line, you should output the answer for $n = i$.

Sample Input 1	Sample Output 1
3	6 54 510

Sample Input 2	Sample Output 2
5	20 540 14300 370300 9454620

Sample Input 3	Sample Output 3
20	380 158460 63889400 636003875 443532759 163564701 433390846 160318339 979712600 445802634 862134704 374397421 898644169 181404073 884138261 856576908 608198482 349239556 724235122 812173715

Problem D. Many-hued Tree

- Time Limit: 3 sec

Problem Statement

There is a tree with N nodes numbered from 1 to N. For each $i = 1, \ldots, N-1$, the i-th edge connects node u_i and node v_i.

You are going to paint all nodes in distinct colors. Colors are represented by integers between 1 and N.

The assignment of colors on the tree is called good, if it is possible to complete the following operation $N-1$ times repeatedly.

- Select a pair of colors (A, B) which satisfies the following two conditions.
  - $|A - B| = 1$.
  - There exists an edge which connects a node painted in color A and a node painted in color B.
- Change the color of all nodes currently painted in color A to color B.

Your task is to count the number of good assignments of colors on the tree modulo $998{,}244{,}353$.

Input

The input consists of a single test case of the following format.

$$\begin{array}{l} N \\ u_1 \ v_1 \\ u_2 \ v_2 \\ \vdots \\ u_{N-1} \ v_{N-1} \end{array}$$

The first line consists of an integer N, which satisfies $1 \leq N \leq 2{,}000$. Each of the $N-1$ lines consists of two integers u_i, v_i, which satisfies $1 \leq u_i, v_i \leq N$. The given graph is guaranteed to be a tree.

Output

Output in a line the number of assignments of colors on the given tree modulo 998,244,353.

Sample Input 1	Sample Output 1
4 1 2 2 3 3 4	16

Sample Input 2	Sample Output 2
4 1 2 1 3 1 4	24

Problem E. Gacha 101

- Time Limit: 2 sec

Problem Statement

For each $i = 1, 2, \ldots, N$, there are A_i balls with i written on them. These are put into a box and mixed up. The string variable s consists of initially N "0"s. Balls are taken out of the box one by one (uniformly at random and independently). When a ball with i written on it is drawn, the i-th character of s is changed to "1" (it remains unchanged if it was already "1"). Find the probability, modulo $998{,}244{,}353$, of having a point during this process that s contains "101" as a contiguous substring.

Input

The input consists of a single test case of the following format.

$$
\begin{array}{l}
N \\
A_1\ A_2\ \ldots\ A_N
\end{array}
$$

The first line consists of an integer N between 1 and $200{,}000$, inclusive. The second line consists of N positive integers $A_1, A_2, \ldots, A_N$. For each i $(1 \leq i \leq N)$, A_i represents the number of balls i written. And they satisfy $\sum_{1 \leq i \leq N} A_i < 998{,}244{,}353$.

Output

Output in a line the probability modulo $998{,}244{,}353$.

Note

- How to find the probability modulo $998{,}244{,}353$
  - It can be proved that the sought probability is always a rational number. Additionally, the constraints of this problem guarantee that if the sought probability is represented as an irreducible fraction $\frac{y}{x}$, then x is not divisible by $998{,}244{,}353$. Here, there is a unique $0 \leq z < 998{,}244{,}353$ such that $y \equiv xz \pmod{998{,}244{,}353}$, so report this z.

Sample Input 1	Sample Output 1
3 1 2 3	465847365

Sample Input 2	Sample Output 2
10 3 1 4 1 5 9 2 6 5 3	488186016

Problem F. Digit-only subrectangles

- Time Limit: 3 sec

Problem Statement

There are H rows and W columns of square cells. Each cell has either a digit or an asterisk ('*'). The cell at the i-th row from the top and the j-th column from the left is denoted by (i, j).

In this problem we consider subrectangles, each of which is the set of cells which forms a rectangle. More precisely, a set of cells S is a subrectangle if there are four integers t, b, l and r such that $1 \leq t \leq b \leq H$, $1 \leq l \leq r \leq W$ and $S = \{(i, j) \mid t \leq i \leq b \land l \leq j \leq r\}$. A subrectangle is digit-only if every cell in the subrectangle has a digit. The score of a digit-only subrectangle is defined as the square of the sum of digits in cells in the subrectangle.

Your task is to calculate the sum of scores of all digit-only subrectangles. Since the answer may be large, output it modulo $998{,}244{,}353$.

Input

The input consists of a single test case of the following format.

$$
\begin{array}{l}
H\ W \\
A_{1,1}A_{1,2}\cdots A_{1,W} \\
A_{2,1}A_{2,2}\cdots A_{2,W} \\
\vdots \\
A_{H,1}A_{H,2}\cdots A_{H,W}
\end{array}
$$

The first line consists of two integers H and W, which satisfy $1 \leq H \leq 2{,}000$ and $1 \leq W \leq 2{,}000$. Each of the following H lines consists of W characters. Here, $A_{i,j}$ is the character in the cell (i, j), and it is either a digit between 0 and 9, inclusive, or an asterisk ('*'). It is guaranteed that there is at least one digit-only subrectangle.

Output

Output in a line the sum of scores of all digit-only subrectangles modulo $998{,}244{,}353$.

Sample Input 1	Sample Output 1
2 2 44 9*	346

Sample Input 2	Sample Output 2
2 3 314 28*	601

Sample Input 3	Sample Output 3
4 6 314159 2*6535 *89793 238*4*	37655

Sample Input 4	Sample Output 4
18 20 65929431919981098712 34182289733359024486 *5999742744659484782 03563591172305229098 55764088882794210744 65542986390400199274 24954674699538357427 65448003011829165060 0570520*394989799204 21113635765787241691 24382969673042349665 04571518994293776944 42950768895299998684 02191975238817773041 08629513210946362875 91583470151322043009 00337992511803056114 59396973995193492513	78257625

In Sample Input 1, there are five digit-only subrectangles as illustrated below. The sum of their scores is $4^2 + 4^2 + 9^2 + (4+4)^2 + (4+9)^2 = 346$.

Figure F.1. Digit-only subrectangles in Sample Input 1

Problem G. Convex Polygon MST

- Time Limit: 7 sec

Problem Statement

There is a convex polygon with n vertices on a plane. Let V be the set of vertices of this convex polygon. After removing all the edges of the convex polygon, you will create a tree with n vertices by repeating the following operation $n-1$ times:

- Select two distinct vertices $x, y \in V$. Add an edge between vertices x and y. If we denote the Euclidean distance between vertices x and y as $d(x,y)$, you gain a score of $(d(x,y))^2$ points.

Find the maximum possible total score obtained by $n-1$ operations.

Input

The input file contains multiple test cases. The first line contains an integer t representing the number of test cases. Following that, t test cases are given. Each test case is given in the following format:

$$\begin{array}{l} n \\ x_1 \ y_1 \\ \vdots \\ x_n \ y_n \end{array}$$

Here, n is an integer representing the number of vertices, where $3 \le n \le 120{,}000$. The sum of all n values in a single input file is guaranteed to be less than or equal to 120,000.
x_i and y_i represent the coordinates of the i-th vertex, where each coordinate is an integer between -10^9 to 10^9. The vertices are given in counterclockwise order when viewed from the centroid of the convex polygon. Three different vertices of the convex polygon do not lie on a single line.

Output

Output the maximum possible total score obtained by $n-1$ operations.

Note

- The Euclidean distance between coordinates (x_1, y_1) and (x_2, y_2) is calculated as $\sqrt{|x_1 - x_2|^2 + |y_1 - y_2|^2}$.
- Note that the answer can exceed 2^{64}.

Sample Input

```
2
4
0 0
1 0
1 1
0 1
6
986288255 165031740
-353860917 -935298054
-173584601 -984818960
141060317 -990001002
341839727 -939758266
662792114 -748803453
```

Sample Output

```
5
1042693651966270814б
```

Problem H. LCP Queries

- Time Limit: 2.5 sec

Problem Statement

A string x is called a *prefix* of a string y if x can be obtained by repeating the removal of the last letter of y zero or more times. For example, "abac", "aba", "ab", "a", and an empty string are the prefixes of "abac".

For two strings x and y, let $\mathrm{LCP}(x, y)$ be the length of the longest common prefix of x and y. For example, $\mathrm{LCP}(\text{“abacab”}, \text{“abacbba”}) = 4$ because the longest common prefix of these two strings is "abac". Note that $\mathrm{LCP}(x, y)$ is always defined for any strings x and y because at least an empty string is one of their common prefixes.

You are given n strings $s_1, \ldots, s_n$ and m strings $t_1, \ldots, t_m$ of lowercase English letters. Then, you are given q queries. In each query you are given an integer sequence $a_1, \ldots, a_k$. Let u be the concatenation of $t_{a_1}, \ldots, t_{a_k}$. Your task is to calculate $\sum_{i=1}^{n} \mathrm{LCP}(u, s_i)$.

Input

The input consists of a single test case of the following format.

$$
\begin{array}{l}
n \\
s_1 \\
\vdots \\
s_n \\
m \\
t_1 \\
\vdots \\
t_m \\
q \\
\mathtt{Query}_1 \\
\vdots \\
\mathtt{Query}_q
\end{array}
$$

The first line consists of an integer n. Each of the next n lines consists of a non-empty string s_i of lowercase English letters. The next line consists of an integer m. Each of the next m lines consists of a non-empty string t_j of lowercase English letters.

The next line consists of an integer q. Then q queries are given in order. Each of the queries is given in a single line in the following format.

$$k \ a_1 \ \cdots \ a_k$$

k is a positive integer which represents the length of the integer sequence of this query. Each a_i is an integer between 1 and m, inclusive.

You can assume that $1 \leq n \leq 200{,}000$, $1 \leq m \leq 200{,}000$ and $1 \leq q \leq 200{,}000$. The sum of lengths of s_i does not exceed 200,000. Similarly, the sum of lengths of t_i does not exceed 200,000. The sum of k over all queries does not exceed 200,000.

Output

Output q lines. The i-th line should be the answer to the i-th query.

Sample Input	Sample Output
5 abcde aaa a ab bcd 5 a bc de aaaa b 5 1 1 3 1 2 3 2 2 3 5 5 4 3 2 1 3 3 3 3	4 9 3 1 0

Problem I. Best parentheses

- Time Limit: 2 sec

Problem Statement

A string consisting only of parentheses '(' and ')' is called *balanced* if it satisfies one of the following conditions.

- It is an empty string.
- It is a concatenation of two non-empty balanced strings.
- It is a concatenation of '(', a, and ')', for some balanced string a.

You are given n characters $s_1, \ldots, s_n$ of parentheses and n integers $c_1, \ldots, c_n$. Then, you have to choose zero or more integers $t_1, \ldots, t_k$ so that they satisfy the following conditions.

- $1 \le t_1 < t_2 < t_3 < \cdots < t_k \le n$.
- The concatenation of $s_{t_1}, s_{t_2}, \ldots, s_{t_k}$ is a balanced string.

Note that the above conditions are always satisfied if you choose zero integers.

Your task is to maximize $\sum_{i=1}^{k} c_{ti}$.

Input

The input consists of a single test case of the following format.

n
$s_1 s_2 \cdots s_n$
$c_1 \; c_2 \; \cdots \; c_n$

The first line consists of an integer n ($1 \le n \le 300{,}000$). The second line consists of n characters $s_1 s_2 \cdots s_n$, each of which is either '(' or ')'. The third line consists of n integers $c_1 \; c_2 \cdots c_n$ ($|c_i| \le 10^9$).

Output

Output in a line the maximum possible value of $\sum_{i=1}^{k} c_{ti}$ by choosing zero or more integers $t_1, \ldots, t_k$.

Sample Input 1	Sample Output 1
5 ()(() 3 -9 -2 1 0	3

Sample Input 2	Sample Output 2
6)()()(-3 1 -4 1 -5 9	0

Problem J. Edit distance on table

- Time Limit: 3 sec

Problem Statement

You have a table with H rows and W columns. Each cell of the table contains a letter.

You are going to construct a string by the following steps.

- Step 1: Pick up a cell in the table and let S be a string of length 1 containing the letter in the cell.
- Step 2: Do either
  - stop building S, or
  - select a cell from four cells which shares an edge with the current one. Then, append the letter in the cell to S, and move to the cell. Then, repeat step 2.

You also have a string T. Your mission is to minimize the edit distance between S and T.

The edit distance (also known as Levenshtein distance) between string U and V is the minimum number of steps required to convert U into V by using the following operations.

- Replace a character in U with another one.
- Insert a character into U.
- Delete a character from U.

Input

The input consists of a single test case in the following format.

$$
\begin{array}{l}
H\ W \\
c_{1,1}c_{1,2}\ldots c_{1,W} \\
c_{2,1}c_{2,2}\ldots c_{2,W} \\
\vdots \\
c_{H,1}c_{H,2}\ldots c_{H,W} \\
T
\end{array}
$$

H and W $(2 \leq H, W \leq 100)$ represents the height and the width of the table respectively.
$c_{i,j}$ $(1 \leq i \leq H, 1 \leq j \leq W)$ is a character in the cell in the i-th row and the j-th column. T is a non-empty string. The length of T doesn't exceed $2{,}000$. $c_{i,j}$ and T consist of lowercase English letters.

Output

Output the minimum possible edit distance between S and T in one line.

Sample Input	Sample Output
2 2 ab ar abracadabra	2

Problem K. Odd trip plans

- Time Limit: 3 sec

Problem Statement

JAG is a country with n airports numbered through 1 to n. There are some airways, each of which connects two different airports bidirectionally. In other words, if an airway connects airports u and v, a passenger can move either from u to v or from v to u in a single flight. Airways may be newly established or abolished.

Mr. Oddytrip, who is a traveler loving odd numbers, plans a trip from an airport to another one by flights. Let's say that he boards k flights: A flight from airport p_1 to p_2, then from p_2 to p_3, then from p_3 to p_4, and so on, and finally from p_k to p_{k+1}. This trip plan, which begins with p_1 and ends with p_{k+1}, is written as $p_1 \to p_2 \to p_3 \to p_4 \to \cdots \to p_k \to p_{k+1}$. According to his aesthetics, a trip plan is *beautiful* if each of n airports appear an odd number of times in the trip plan. For example, if $n = 6$, trip plans $3 \to 4 \to 5 \to 6 \to 1 \to 2$ and $1 \to 2 \to 3 \to 4 \to 5 \to 3 \to 2 \to 3 \to 2 \to 6$ are beautiful while $1 \to 3 \to 6$ and $1 \to 2 \to 3 \to 4 \to 1 \to 5 \to 6$ aren't. In particular, each of the n airports appears at least one in a beautiful trip plan.

Initially, there are m airways. Then, you are given q queries, which should be processed in the order they are given. Each query is of one of the two kinds below:

- $1\ x\ y$: The existence of the airway between airports x and y changes. If there is already an airway between airports x and y, then such an airway is abolished. In other words, Mr. Oddytrip is no longer able to board the direct flight between airports x and y (until it is newly established again). On the other hand, if there wasn't such an airway before, an airway between airports x and y is newly established. In other words, Mr. Oddytrip can board a direct flight between airports x and y (until it is abolished again).
- $2\ x\ y$: You have to determine whether there can be a beautiful trip plan which begins with airport x and ends with airport y using the airways which are available at that time.

Input

The input consists of a single test case of the following format.

$$
\begin{array}{l}
n\ m\ q \\
u_1\ v_1 \\
\vdots \\
u_m\ v_m \\
t_1\ x_1\ y_1 \\
\vdots \\
t_q\ x_q\ y_q
\end{array}
$$

The first line consists of three integers n, m and q ($2 \leq n \leq 100{,}000$, $0 \leq m \leq 100{,}000$, $1 \leq q \leq 100{,}000$), where n is the number of airports in JAG country, m is the number of airways which are initially available, and q is the number of queries.

The i-th of the following m lines consists two integers u_i and v_i ($1 \leq u_i < u_i \leq n$) representing that an airway between airports u_i and v_i is initially available. It is guaranteed that these m airways are distinct.

The j-th of the following q lines consists of three integers t_j, x_j and y_j ($1 \leq t_j \leq 2$, $1 \leq x_j < y_j \leq n$) representing the type of the query and the numbers of two airports as described above. It is guaranteed that there is at least one query where $t_j = 2$.

Output

For each query where $t_j = 2$, print "Yes" in a single line if there can be a beautiful trip plan which begins with airport x and ends with airport y. Otherwise, print "No" in a single line.

Sample Input 1

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

Sample Output 1

```
No
Yes
Yes
```

Sample Input 2

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

Sample Output 2

```
Yes
No
Yes
```