Problem A. Mash

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Nikuniku designs a new programming language with only two kinds of instructions. Initially, we can consider the program as a queue P containing of n instructions. The program will need another queue of Q to run. Initially, you copy all instructions from P and add them to the back of the queue Q in order. Afterward, each time Q will pop the instruction in the front and run it. Now we give the format and usage of the two instructions:

- `echo c`: Output a lower-case character c;
- `cp m`: Copy the first m instructions from P and add them to the back of the queue Q in order. It is guaranteed $1 \le m \le n$ here.

Now you need to simulate the process after running k instructions from the queue. Specifically, please compute the string outputted by the program.

Input

The first line has two integers, n and k ($1 \le n, k \le 10^5$).

For the following n lines, each line contains an instruction in the format of s and t separated by one space. s will be either "`echo`" or "`cp`". If s is "`echo`", t will be a single lower-case character. Otherwise, t will be an integer between 1 and n.

Output

Output the result string in one line for the first k instructions. If the program terminates with less than k instructions run, output the result for all instructions.

Examples

standard input	standard output
2 20 echo a cp 2	aaaaaaaaaa
3 18 echo a cp 2 echo b	abaaaaaaaa
4 40 echo a cp 2 echo b cp 4	abaabaaabaaaabaaaaab
5 50 echo a cp 2 echo b cp 5 cp 5	abaababaaababaababaaaa

Problem B. Shuttle Bus

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2.5 seconds
Memory limit:	512 megabytes

The city contains n locations, numbered from 1 to n and connected by $n-1$ bi-directional roads. The length of each road is 1. That is to say, the city looks like a tree in graph theory.

Nikuniku is working for the Machinery Factory and needs to choose a location t to build a new workshop in the city.

There are a_i staff living in location i.

After choosing the target location, Nikuniku will design **at most** k shuttle bus routes, to help staff commute from home to the factory. The i-th route is a simple path between location s_i and t ($s_i \neq t$) inclusive.

Each staff will walk to the nearest location where at least one route passes by and get on the bus. The capacity of the bus is large enough so don't worry about it. The walking distance of one staff is the length of the shortest path between the living location and the location getting on the bus.

Nikuniku wants to know the minimum total walking distance of all staff for every location t ($1 \le t \le n$), if Nikuniku chooses location t to build the workshop and designs the proper routes.

But Nikuniku just got laid off for taking too many paid leaves, so you are asked to help her by using your programming skill.

Input

The first line of the input contains two integers n ($1 \le n \le 2 \cdot 10^5$) and k ($1 \le k \le n$) — the number of locations and the number of shuttle bus routes.

The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^6$) — the number of staff living in each location.

Each of the next $n-1$ lines contains two integers x and y ($1 \le x, y \le n$, $x \neq y$) - meaning that there is a road between location x and y.

It is guaranteed that these roads form a tree.

Output

Output n lines. The i-th line output an integer, minimum total walking distance if building the workshop in location i.

Examples

standard input	standard output
5 2	2
1 2 3 4 5	0
1 2	0
1 4	3
3 4	0
4 5	
8 3	3
3 1 4 1 5 9 2 6	3
2 1	1
3 1	2
5 1	3
2 7	1
2 4	1
8 5	1
6 2	

Problem C. Selection Sort Count

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Nikuniku learns the Selection Sort today. To better understand this algorithm, she implements it again and computes the number of swaps for each round, as the following pseudocode shows:

Algorithm 1 Selection Sort

```
1: function SORT(Permutation $P$)
2:     for $i$ in $[1, n-1]$ do
3:         $c_i \leftarrow 0$
4:         for $j$ in $[i+1, n]$ do
5:             if $P_i > P_j$ then
6:                 $swap(P_i, P_j)$
7:                 $c_i \leftarrow c_i + 1$
```

Nikuniku wants to know how many different kinds of permutations P of length n there are, so that after running the algorithm, the number of swaps of each round (denoted as the counter c in the pseudocode) matches a given $t_1, \dots, t_{n-1}$. You only have to output the answer modulo 998244353. A permutation of length n consists of $1, \dots, n$ exactly once.

Input

The first line has one integer n ($2 \le n \le 2 \cdot 10^5$).

In the following line, there are $n-1$ integers $t_1 \dots t_{n-1}$. It is guaranteed that there exists at least one permutation satisfying the requirement.

Output

Output the answer in one line, modulo 998244353.

Examples

standard input	standard output
3 1 1	2
6 0 1 1 1 0	4
6 0 1 2 1 1	6

Problem D. Tree Partition

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

You are given a tree of n vertices, indexed in 1 to n. Note that by deleting $x - 1$ edges, the tree will become x connected components.

It is required that after deleting edges, for each component, the set of indexes value in it is a continuous interval. That is, denoting the minimum and maximum value of the index value of the component as l and r, every vertice indexed in $[l, r]$ should belong to the same component.

Now for $x = 1, 2, \ldots, k$, you need to compute the number of ways to delete edges. Two ways are different if at least one edge is deleted in one of them and preserved in the other. You only need to compute the answer modulo 998244353.

Input

The first line has two integers n $(1 \le n \le 2 \cdot 10^5)$ and k $(1 \le k \le \min(n, 400))$.

In the following $n - 1$ lines, there are two integers x, y $(1 \le x, y \le n, x \ne y)$ in each line, indicating an edge (x, y).

Output

Output k lines, each line with an integer, indicating the answer modulo 998244353.

Examples

standard input	standard output

4 3
1 2
2 3
2 4
	1
	2
	2

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

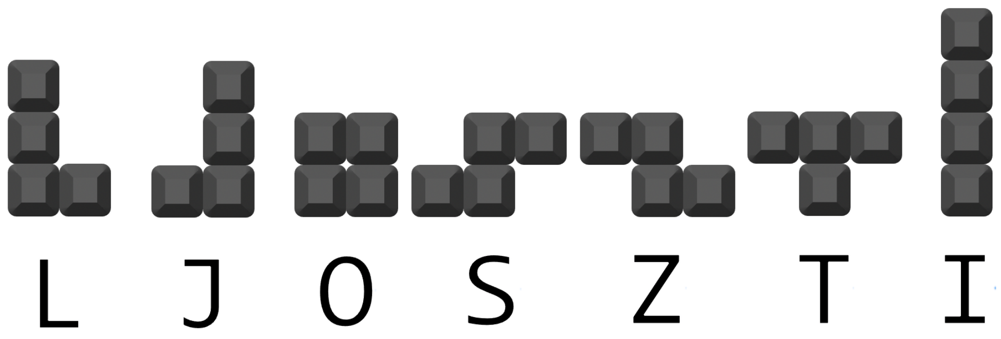
Problem E. Elegant Tetris

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Tetris is a classical puzzle video game. In Tetris, players complete lines by descending differently shaped pieces (called tetrominoes) onto the playing field. The completed lines disappear and grant the player points.

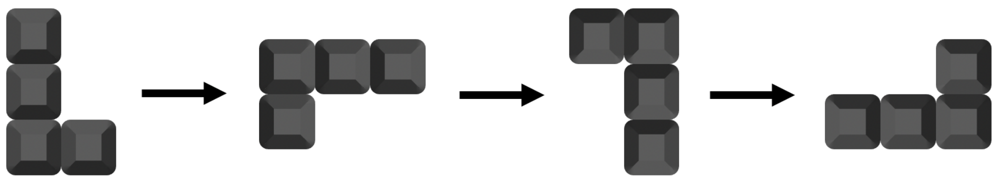
Elegant Tetris is a slightly different Tetris variant. In Elegant Tetris, you choose the shape, rotation and where to descend for each incoming piece.

There are 7 distinct shapes of tetrominoes, with each one represented by an uppercase letter, as in figure T-1.



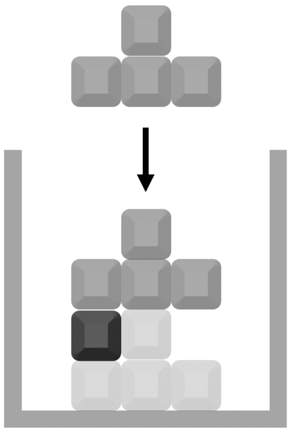
T-1

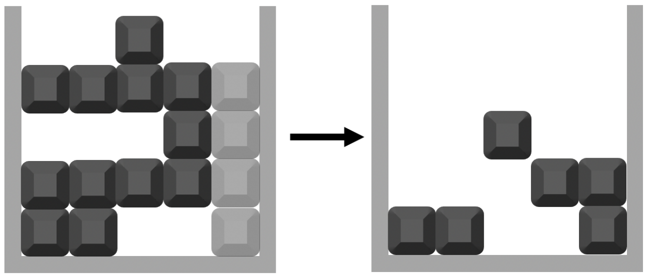
After deciding the shape of the piece, you may choose how many times to rotate the incoming piece. Each piece can be rotated at least 0 times and at most 3 times. Each rotation rotates the piece by 90° clockwise. For example, O-shaped pieces don't change after rotation, but L-shaped pieces change as in figure T-2.



T-2

You may then decide where to descend the piece. The piece stops when it touches the bottom of the playing field or another piece. Note that in Elegant Tetris, you cannot move the piece horizontally during descending. For example, as shown in figure T-3, the T-shaped piece could be put under the 1x1 block in Tetris, but not in Elegant Tetris.


T-3


T-4

When the incoming block stops, all completed (fully filled) lines disappear, and the lines above fall, as figure T-4 shows. If after that there are still filled blocks with height **not less** than 20, game over.

Given an initial playing field with width w and some blocks already filled, Nikuniku wants to know how to go back to the initial playing field with no less than 1 and no more than 10000 moves without triggering game over.

Input

The first line contains two integers separated by a space w ($4 \le w \le 1000$) and n ($0 \le n \le 15$). w is the width of the playing field.

For the following n lines, each line contains a string with length w, denoting the lowest n lines of the playing field. The string consists of two symbols, '`#`' and '`.`'. '`#`' means the block is filled, and '`.`' means empty. No lines in the initial playing field are fully filled. All lines above height n are empty in the initial playing field.

It is guaranteed that the challenge is solvable. Note that it is **not** guaranteed that there is a movement sequence to go from the empty playing field to the initial playing field.

Output

On the first line, output an integer len ($1 \le len \le 10000$) – how many moves to make. Note that len does **not** have to be the minimal number of moves possible.

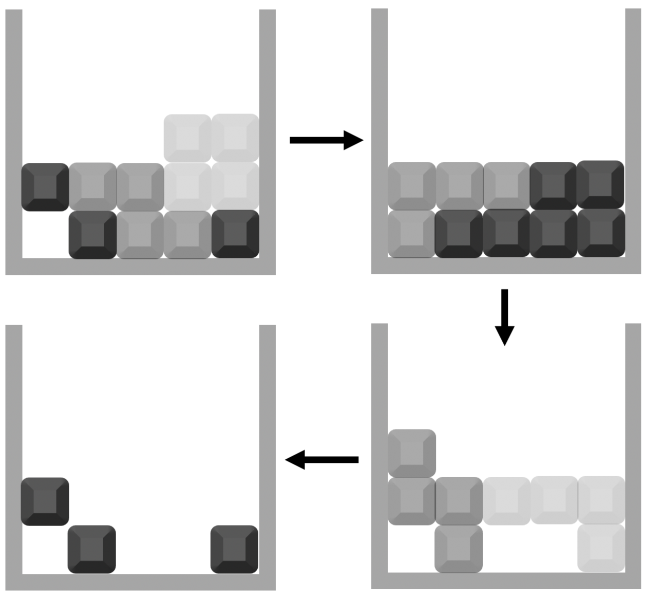
For the following len lines, output one uppercase letter ch ($ch \in \{$'`L`', '`J`', '`O`', '`S`', '`Z`', '`T`', '`I`'$\}$) and two integers a ($0 \le a \le 3$) and x ($1 \le x \le w$) separated by spaces – the shape, rotation count and the coordinate of the leftmost block, accordingly.

Examples

standard input	standard output
5 2 #.... .#..#	5 Z 0 2 O 0 4 L 1 1 S 1 1 J 3 3
5 4 #.... ###.. ####. #..#.	5 L 2 4 L 0 4 O 0 2 L 1 1 J 1 1

Note

The following figure shows the situation in the first example test case.



T-5

Problem F. Modulo

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

You are given a 1-indexed positive integer array a of length n and a positive integer x. You can rearrange all elements of this array as you wish. Then you should perform the operation $x \leftarrow x \bmod a_i$ in order from 1-th to n-th.

Your task is to rearrange the array in such a way that x is the maximum possible after n operations.

Input

The first line contains one integer n ($1 \le n \le 21$) — the length of the array.

The second line contains n space-separated positive integers a_i ($1 \le a_i \le 10^{18}$) — the elements of the array.

The third line contains one integer x ($1 \le x \le 10^{18}$) — the number to perform operations.

Output

Print a single integer in one line — the maximum value of x to the problem.

Examples

`standard input`	`standard output`

`3`
`5 6 7`
`15`
	`3`

`4`
`20 21 22 10`
`107`
	`9`

`5`
`5 6 7 8 9`
`1000000000000`
	`4`

Problem G. Integer Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Two players play a game of integers.

The game consists of n positive integer sets and an integer p greater than 1. The i-th set s_i initially contains all integers in the range $[l_i, r_i]$.

The first player makes the first move, then players alternate turns.

In one move, the player must choose a non-empty set s_i and select an integer x from the chosen set satisfying $x \times p \geq max(s_i)$. Then the player should remove all integers not less than x from this set.

The player who can not make a move loses the game.

You want to know whether the player who goes first will win the game, if both players play optimally.

Input

The first line contains one integer t ($1 \leq t \leq 2 \cdot 10^5$) — the number of test cases. The description of test cases follows.

The first line of a test case contains two integers n and p ($1 \leq n \leq 2 \cdot 10^5$; $2 \leq p \leq 10^9$).

Then n lines follow.

Each of the next n lines contains two integers l_i and r_i ($1 \leq l_i \leq r_i \leq 10^9$).

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print a single string in one line. Print "`First`" (without quotes) if the player makes the first move will win, otherwise print "`Second`" (without quotes).

Example

standard input	standard output

```
4
1 3
1 6
4 100
1 10
2 16
1 7
12 13
3 5
1 8
20 20
13 18
3 2
1 10
2 9
3 4
```
	```
	First
	Second
	Second
	First
	```

Problem H. Harie Programming Contest

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2.5 seconds
Memory limit:	512 megabytes

During 9102 A.D., n students work for Harie University Wearable Technology Lab (or WTLab in short). Each student is self-identified as being either a "Ninniku Homan" (NH) or a "Siege Guy" (SG).

Nikuniku wants to organize a new kind of programming contest among them, in which each team has two members – one NH and one SG. The students are not allowed to pick their teammates by themselves. Instead, they make proposals and let Nikuniku decide the final team. A proposal between student x and y means they are willing to work together. Since the penalty of being dishonest at WTLab is pretty high, it is guaranteed that the proposals are valid, i.e. either x or y is a Siege Guy, but not both. Nikuniku does not know who is NH or SG, though.

Nikuniku would like to arrange teams in a way that her contest has as many teams as possible. However, to keep the lab's critical infrastructure running, she has to reserve two students for emergency. The two reserved students, unfortunately, cannot participate in the contest.

But Nikuniku doesn't want to compromise – she still wants the same number of teams as if all n students are participating. Nikuniku is wondering how many different ways of picking the two reserved students there are so that she doesn't need to make a compromise.

Input

The first line contains two integers separated by a space n ($2 \le n \le 2 \cdot 10^5$) – the number of students, and m ($0 \le m \le 2 \cdot 10^5$) – the number of proposals.

The next m lines each contain two integers x ($1 \le x \le n$) and y ($1 \le y \le n, x \ne y$), meaning that there is a proposal between x and y.

It is guaranteed that the proposals are unique, i.e. for any given x and y, there is at most one proposal between them.

Output

Output an integer in one line – the answer to Nikuniku's question.

Examples

standard input	standard output

6 4
1 2
1 3
4 5
4 6
	4

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

5 4
1 2
2 3
3 4
4 5
	0

Problem I. Reverse LIS

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	512 megabytes

Nikuniku loves long, non-decreasing subsequences. She has a 0-1 string s and wants to make the longest non-decreasing subsequences of s as long as possible.

0-1 string consists of only two kinds of characters, '`0`' and '`1`', in which '`0`' is considered to be smaller than '`1`'. A subsequence of a string is a new string formed from the original string by deleting some (can be none) of the characters without disturbing the relative positions of the remaining characters. (i.e., "`010`" is a subsequence of "`0110`" while "`001`" is not).

Nikuniku decided to make some changes to the string. Each change reverses an arbitrary substring of s (i.e. make $s[l, r] = s_r s_{r-1} \dots s_l$). She would like to know how long is the longest possible non-decreasing subsequence after at most k changes (or $\mathrm{revlis}(s, k)$ in short), for q different k-s. Note that the q queries are independent.

As the string s could be very long, Nikuniku decided to present it in run-length encoded form. A run-length encoded string contains n parts, each part has p_i copies of the same character c_i. For example, string "`001111`" has two parts, with p_1 being 2, c_1 being '`0`', p_2 being 4, and c_2 being '`1`'.

Input

The first line contains an integer n ($1 \le n \le 2 \cdot 10^5$), the number of parts in the string s.

The following n lines describe the parts. Each line contains a character c_i ($c_i \in \{\text{'0'}, \text{'1'}\}$) and an integer p_i ($1 \le p_i \le 10^9$), separated by a space. It is guaranteed that $c_i \ne c_{i+1}$ for $1 \le i \le n - 1$.

The next line contains an integer q ($1 \le q \le 2 \cdot 10^5$), denoting the number of queries. For the next q lines, each line contains an integer k_i ($0 \le k_i \le n$).

Output

For each query, output one line with an integer – the value of $\mathrm{revlis}(s, k_i)$.

Examples

standard input	standard output

4
0 1
1 2
0 3
1 4
2
0
1
	8
	10

5
1 2
0 5
1 2
0 5
1 2
3
5
0
5
	16
	12
	16

7
0 1
1 3
0 7
1 6
0 6
1 6
0 6
2
1
5
	26
	35

Problem J. jwfw.harie.edu

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2.5 seconds
Memory limit:	1024 megabytes

At Harie University, you must pass a course registration code of conduct quiz before using the course registration system.

The quiz has 10 multiple choice questions, with four options (‘`A`’, ‘`B`’, ‘`C`’, ‘`D`’) for each. There is only one correct answer for each question. Each correct answer gives 10 points, and wrong answers give no points. A perfect score (100) is required to pass the quiz. You may attempt the quiz multiple times, and the set of questions for each quiz remains unchanged. When you complete the quiz, the system will tell you the final score, but it won’t tell you whether your answer is correct for individual questions.

You have failed the quiz for quite a lot of times, and are wondering how many possible answers there are according to the previous observations. More specifically, given the chosen options and scores of prior quiz attempts, count the number of remaining possible answers out of all option combinations.

It is guaranteed that the input data is consistent, i.e. there is at least one possible answer.

Input

The input consists of several independent test cases. The first line contains an integer t ($1 \le t \le 20000$), which is the number of test cases.

For each test case, the first line contains an integer n ($1 \le n \le 20000$) – the number of quizzes. The following n lines describe the results of these quizzes. Each line contains a string s ($|s| = 10$, $s_i \in \{\text{‘A’}, \text{‘B’}, \text{‘C’}, \text{‘D’}\}$) and an integer a ($0 \le a \le 90$, a is a multiple of 10), separated by a space. The i-th character of s denotes the chosen option of the i-th question, and a is the quiz score.

The sum of all n in an input file will not exceed 20000.

Output

For each test case, output an integer – the number of possible answers.

Example

standard input	standard output
3 1 CCCCCCCCCC 90 2 AAAAAAAAAA 10 ABCDABCDAB 20 3 AAAAAAAAAA 0 BBBBBBBBBB 0 CCCCCCCCCC 0	30 57456 1

Problem K. Security Plan

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	512 megabytes

Nikuniku wants to make a security plan for a warehouse. The warehouse can be represented as a grid with n rows and m columns. The upper-left cell is $(1, 1)$, while the bottom-right cell is (n, m).

A security plan is to choose some cells in the grid and place exactly one camera in each of the selected cells.

When placing a camera in cell (i, j) $(1 \le i \le n, 1 \le j \le m)$, Nikuniku can choose a cell (p, q) $(1 \le p \le n, 1 \le q \le m)$ as its parameter, and a cell (x, y) is protected by this camera if and only if both of the following conditions satisfied:

1. $min(i, p) \le x \le max(i, p)$

2. $min(j, q) \le y \le max(j, q)$

Notice that (p, q) is not necessarily different from (i, j) and can't be modified once configured. Different cameras can be configured with different parameters.

A security plan is perfect if and only if the chosen cameras can be configured with appropriate parameters and all of the $n \times m$ cells can be protected by at least one camera.

For a perfect security plan, if Nikuniku can not remove any camera from the plan and change the parameters of the remaining cameras such that the plan remains perfect, we call it a minimal security plan.

Nikuniku wants to know the number of different minimal security plans.

Two plans are different if and only if one cell is chosen in one plan but not in the other, regardless of the parameters.

Input

The first line of the input contains one integer t $(1 \le t \le 10^5)$ — the number of test cases.

Each test case contains two integers n and m $(1 \le n, m \le 10^9)$ in one line — the grid size.

Output

Print a single integer for each test case — the answer to the test case modulo 998244353.

Example

standard input	standard output

6
2 2
4 4
8 8
3 3
3 5
3 7
	4
	129
	78769
	10
	80
	273

Problem L. Paid Leave

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Nikuniku is working in a factory with a brutal work schedule. She has to work every day from 00:00 to 00:00 (the next day). Such a work schedule is known as ***007***. To keep her efficiency high, she knows she needs to choose to take a break on some days. Fortunately, there are some legal holidays that she doesn't need to go to work. Moreover, she can choose some working days to rest and still get paid, known as paid leave.

Nikuniku needs to make a plan for the following n days, enumerated from 1 to n. We already know there are m legal holidays on the p_1th, …, p_mth day. Now Nikuniku needs to choose some extra days as paid leave days. Specifically, she wants the schedule of the n days to satisfy the following two conditions:

- Any period of consecutive working days should not exceed x days.
- If there are a consecutive working days just before a single rest day, and b consecutive working days right after this rest day, the sum of a and b must not exceed y. (Note: Both legal holidays and paid leave days are considered rest days)

Nikuniku wants you to calculate the minimum paid leave days she needs to take following the above conditions.

Input

There are four integers n $(1 \le n \le 10^{18})$, m $(0 \le m \le \min(n, 2 \cdot 10^5))$, x, y $(1 \le x \le y \le \min(2 \cdot x, 10^{18}))$ in the first line.

There are m integers in the second line: $p_1, \dots, p_m$ $(1 \le p_1 < \dots < p_m \le n)$, indicating all legal holidays.

Output

Output the answer in one line.

Examples

standard input	standard output
`8 0 3 3`	`2`

`11 1 2 4`
`6`
	`2`

`17 2 5 7`
`6 12`
	`1`