AND on Graph (GRAPHAND)

You are given an undirected graph with n nodes and m edges. The nodes are numbered from 1 to n. Each edge e of the graph has two integers associated with it, cost(c_e) and value(v_e). You are given three integers s, t, and K.

The cost of a walk is the sum of costs of all the edges in the walk. Find the minimum cost of a walk from s to t ($s \neq t$), such that the bitwise AND of values for all edges in the walk is atleast K. If no such walk exists, print -1.

Note that the walk doesn't need to be a simple path. Any vertex or edge can come more than once in the walk you choose. If an edge e comes p times, we add its cost p times to the cost of the walk.

Formally, a walk from s to t is a sequence of vertices and edges $u_1, e_1, u_2, e_2, \cdots, u_{l-1}, e_{l-1}, u_l$ where e_i is the edge between vertices u_i and u_{i+1} , $u_1 = s$ and $u_l = t$.

We need to minimize $\sum_{i=1}^{l-1} c_{e_i}$, while satisfying $v_{e_1} \& v_{e_2} \& \cdots \& v_{e_{l-1}} \geq K$, where $\&$ denotes the bitwise AND operator.

Input:

- First line will contain n and m
- Each of the next m lines will contain the description of an edge in the graph. It will consist of 4 integers a, b, c, v denoting the endpoints, cost and val of the edge respectively.
- The last line contains s, t, K, as described in the statement.

Output:

Print the minimum cost of a walk from s to t, such that the bitwise AND of values for all edges in the walk is atleast K. Print -1 if no such walk exists.

Constraints

- $1 \leq n, m \leq 10^5$
- $1 \leq c_i, v_i \leq 10^9$
- $1 \leq s, t \leq n$
- $s \neq t$
- $1 \leq K \leq 10^9$
- The graph contains no self loops and multiedges.

Sample Input:

```
3 3
1 2 2 3
2 3 1 7
1 3 2 2
1 3 3
```

Sample Output:

```
3
```

Explanation:

Suppose we choose the edge between 1 and 3 and say that's our walk. Its cost is 2. But this is not a valid walk, because the AND of the edge values is 2. We need it to be $\geq$ 3.

So, instead suppose we choose the walk from 1 to 3 passing through 2. The bitwise AND is 3 & 7 = 3, and cost is 1 + 2 = 3. This is a valid walk, and you can check that you cannot reduce the cost any further. Hence 3 is the answer.

Make it Zero (INFDIV)

You are given two integers, m and n. You want to make m equal to 0. You decide to run the following experiment until m becomes 0 :

- Choose a integer uniformly randomly r from the range $[1, n]$. Set m equal to the remainder obtained on dividing it by r. That is, replace m by $m\%r$, where $\%$ denotes the modulo operator.

Find the expected number of times you have to run the experiment until m becomes 0. Let the answer be equal to $E = \dfrac{P}{Q}$ for some integers P and Q. Output the value of PQ^{-1} modulo $10^9 + 7$, where Q^{-1} denotes the modular inverse of Q modulo $10^9 + 7$.

Input:

- The only line of the input contains two integers n and m.

Output:

- Output one line containing the value of PQ^{-1} modulo $10^9 + 7$ as described above.

Constraints

- $1 \leq n \leq 200000$
- $1 \leq m \leq 10^{18}$

Sample Input:

```
3 2
```

Sample Output:

```
500000005
```

EXPLANATION:

m is initially 2. It will become 0 if we get the random number as 1 or 2 else it will remain at 3. So, we will repeat the procedure until we get a 1 or 2. The probability of m becoming 0 in each step is $\frac{2}{3}$. Hence, the answer is $\frac{3}{2}$. We print $3 \times 2^{-1} = 3 \times 500000004 = 500000005 \mod 10^9 + 7$

Mafia Syndicate (MAFIASYN)

There is a tree with N nodes, numbered from 1 to N. The tree is rooted at node 1. The tree represents the structure of a big mafia syndicate, and each node represents a criminal. Each criminal, except the leader (who is denoted by node 1), is under the supervision of his boss, who is his parent in the tree.

Initially, each criminal/node except the leader has some money. Starting now, at every second, each criminal will pass on all the money that he has, to his boss (i.e. parent). And during the same second, his 'children' might have passed on some amount to him, which he will pass on to his parent in the next second, and so on.

However, a criminal might go rogue if the money he has strictly exceeds some threshold (that is, if the money that he has is greater than a particular value, he is overpowered by greed). Going rogue means, at some point, if the amount of money that he has to pass on to his boss is greater than the threshold, he won't do so, and instead keep the amount to himself. And after this, all the money that comes to him, he will keep to himself. He will not pass on anything after this.

So, as an example if there is a tree, 1 - 2 - 3, in which 1 is the root, 2 is beneath 1, and 3 is beneath 2. And suppose the amount of money that each of the criminals initially have is (Note that the root node will not have anything in the beginning):

$have[1] = 0$

$have[2] = 10$

$have[3] = 100$

And their thresholds are:

$cap[2] = 10$

$cap[3] = 1000$

That is, if at any second, node 2 has strictly more than 10 amount of money, then he will turn rogue, and will stop passing the money to his boss.

Now, at the first second, no criminal has more money than his threshold. So, both criminals 2 and 3 pass on their money to their bosses. Thus, after the first second, the criminals will have the following:

$have[1] = 10$

$have[2] = 100$

$have[3] = 0$

But now, since $have[2] > cap[2]$, criminal 2 turns rogue, and won't pass on any money to his boss anymore. Nothing changes after this, and so this is the stable state.

You can show that after some finite amount of time, the configuration remains constant, with no more passing of money. We call this the stable state. The problem is, given the tree with N nodes, and two arrays $have[2], have[3], \ldots, have[N]$ and $cap[2], cap[3], \ldots, cap[N]$ of size $N - 1$, find out the total money which the leader (ie. root node) will have at the end, in the stable state.

Input

- The first line contains the integer N, denoting the number of criminals/nodes in the syndicate/tree.
- The i^{th} of the next $N - 1$ lines contains two space separated integers, $have[i + 1]$ and $cap[i + 1]$, which correspond to the amount that criminal $i + 1$ initally has, and the threshold beyond which he will turn rogue. Note that $have[1]$ is 0 and $cap[1]$ is irrelevant, so they are not provided in the input.
- The i^{th} of the next $N - 1$ lines contains a single integer, denoting the immediate boss of the $(i + 1)^{th}$ criminal. Note that criminal 1 (i.e. the leader) has no immediate boss, and hence is not included in this input.

Output

Print the total amount present at the root node at the end, in the stable state.

Constraints

- $1 \leq N \leq 10^5$
- $0 \leq have[i] \leq cap[i] \leq 10^9$

Example Input 1

```
3
10 10
100 1000
1
2
```

Example Output 2

```
10
```

Explanation 1

As explained above, the root will get the money from criminal 2 at the first second, but nothing after that. Hence the answer is 10.

Example Input 2

```
6
10 10
1 2
10 10
1 2
10 10
1
1
2
3
2
```

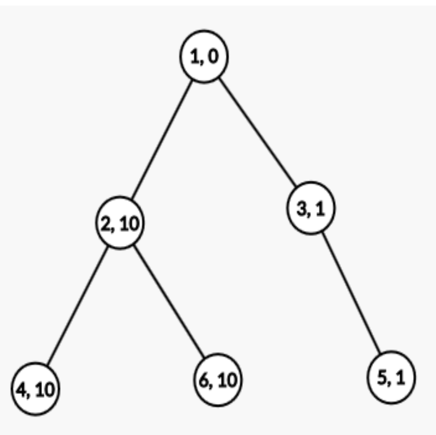
Example Output 2

```
12
```

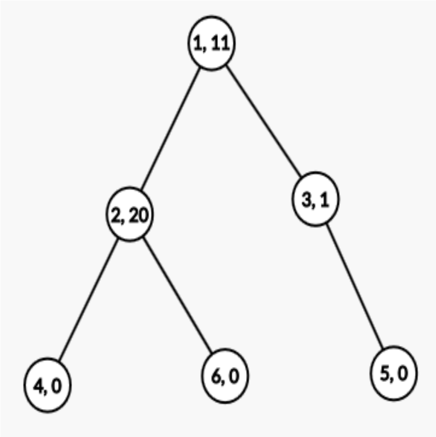
Explanation 2

In the figures below, the first integer in each node corresponds to its number, and the second integer corresponds to the amount held by that criminal at that moment.

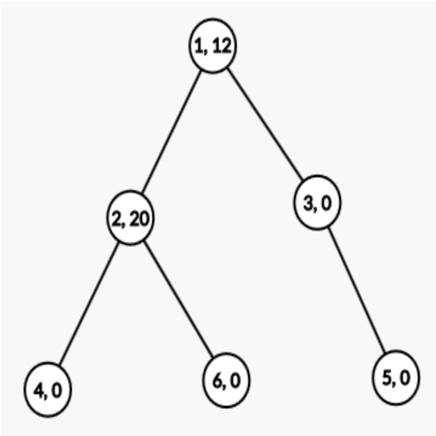
The original configuration is



During the first second, every criminal (except the leader) passes on the amount that they hold to their boss. So, after the first second, it changes to:



During the second second, criminal 2 turns rogue, because 20 > 10. So he stops passing any amount. But criminal 3 doesn't turn rogue, because 1 <= 2. And so he passes on the amount of 1 to his boss. So, after the second second, it changes to:



After this, it does not change, and hence the answer is 12.

1-2 Game (BINGAME)

There are two players P_1, P_2 playing a game on an array A which has n positive integers: $A_1, A_2, \ldots, A_n$. Both the players takes turns alternatively, and P_1 goes first. Rules of the game are as follows.

- In P_1's turn, he should choose any element A_i which is ≥ 1, and subtract 1 from it.
- In P_2's turn, he should choose any element A_i which is ≥ 2, and subtract 2 from it.
- The player who is not able to make a move loses. Given the array A, find the number of different indices i such that that P_1 can subtract from A_i in his first turn, and end up winning the game, if both the players play optimally after this.

Input

- The first line of the input contains an integer T denoting the number of test cases. The description of the test cases follows.
- The first line of each test case contains an integer n.
- The second line of each test case contains n space-separated integers denoting the array A.

Output

For each test case, output an integer corresponding to the number of possible first moves of P_1 that lead him to win.

Constraints

- $1 \leq T \leq 10^5$
- $1 \leq n \leq 10^5$
- $1 \leq A_i \leq 10^5$
- Sum of n over all the test cases doesn't exceed 10^6

Example Input

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

Example Output

```
1
0
1
```

Explanation

Testcase 1: The array is {1}. P_1 has no choice but to pick the element and subtract 1 from it. The resulting array is {0}, and now P_2 is unable to make a move. Hence P_1 wins, and there was only one winning first move. Hence the answer is 1.

Testcase 2: The array is {3}. P_1 has no choice but to pick the element and subtract 1 from it. The resulting array is {2}, and now P_2 has no choice but to subtract 2 from it. The resulting array is {0}, and now P_1 is unable to make a move. Hence P_1 loses.There is no starting move for P_1 which leads to his victory. Hence the answer is 0.

Testcase 3: The array is {1, 2}. P_1 has two choices. Either subtract 1 from 1 or subtract 1 from 2.

- Suppose he subtracts 1 from the first element, then the resulting array is {0, 2}, and now P_2 has no choice but to subtract 2 from the second element. The resulting array is {0, 0}, and now P_1 is unable to make a move. Hence P_1 loses in this case.
- But suppose he subtracts 1 from the second element, then the resulting array is {1, 1}, and now P_2 is unable to make a move. Hence P_1 wins in this case.

Thus, there is exactly one starting move for P_1 which leads to his victory. Hence the answer is 1.

Maximise the Sum (NOMATCH)

You are given an array with N integers: $A[1], A[2], \ldots, A[N]$ (where N is even). You are allowed to permute the elements however you want. Say, after permuting the elements, you end up with the array $A'[1], A'[2], \ldots, A'[N]$. Your goal is to maximize the following sum:

$$|A'[1] - A'[2]| \ + \ |A'[3] - A'[4]| \ + \ \ldots \ + \ |A'[N-1] - A'[N]|$$

Here, $|x|$ denotes the absolute value of x.

You have to print the maximum sum achievable.

Input

- The first line contains T, the number of test cases.
- Each test case starts with an integer N in the first line.
- The second line of each test case contains N space separated integers, denoting the values of array A.

Output

For each test case, output the maximum sum achievable in a new line.

Constraints

- $1 \le T \le 10^5$
- $1 \le N \le 10^5$
- N is even
- $|A[i]| \le 10^9$
- Sum of N over all test cases $\le 2 * 10^5$

Example Input 1

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

Example Output 1

```
9
```

Explanation 1

The original array is $\{1, -3, 2, -3\}$. Suppose you permute it and get the array $\{2, 1, -3, -3\}$. Then the corresponding sum would be $|2 - 1| \ + \ |-3 - (-3)| = 1 + 0 = 1$.

But suppose you permute it differently and get the array $\{-3, 2, 1, -3\}$. Then the corresponding sum would be $|-3 - 2| \ + \ |1 - (-3)| = 5 + 4 = 9$. You can check that you cannot do any better, and hence the answer is 9.

Play Piano (PLAYPIAN)

Two sisters, A and B, play the piano every day. During the day, they can play in any order. That is, A might play first and then B, or it could be B first and then A. But each one of them plays the piano exactly once per day. They maintain a common log, in which they write their name whenever they play.

You are given the entries of the log, but you're not sure if it has been tampered with or not. Your task is to figure out whether these entries could be valid or not.

Input

- The first line of the input contains an integer T denoting the number of test cases. The description of the test cases follows.
- The first line of each test case contains a string s denoting the entries of the log.

Output

- For each test case, output `yes` or `no` according to the answer to the problem.

Constraints

- $1 \leq T \leq 500$
- $2 \leq |s| \leq 100$
- $|s|$ is even
- Each character of s is either 'A' or 'B'

Example Input

```
4
AB
ABBA
ABAABB
AA
```

Example Output

```
yes
yes
no
no
```

Explanation

Testcase 1: There is only one day, and both A and B have played exactly once. So this is a valid log. Hence 'yes'.

Testcase 2: On the first day, A has played before B, and on the second day, B has played first. Hence, this is also a valid log.

Testcase 3: On the first day, A played before B, but on the second day, A seems to have played twice. This cannot happen, and hence this is 'no'.

Count Periodic Numbers (PERIODCN)

A positive integer n is said to be periodic if satisfies the following property:

Take the binary representation of n without leading zeros. Then create an array which contains the length of consecutive runs of equal bits in this binary representation. All the elements of this array should be equal. If there are two unequal elements in this array, then n is not periodic.

For example:

Suppose $n = 3$. Its binary representation is 11. And the array created would be {2}, which corresponds to the fact that there are two equal bits in the beginning. This is periodic.

Suppose $n = 51$. Its binary representation is 110011. And the array created would be {2, 2, 2}, which corresponds to the fact that there are two equal bits are the beginning, then the next two are equal, and then the next two are equal. This is also periodic.

Suppose $n = 103$. Its binary representation is 1100111. And the array created would be {2, 2, 3}, which corresponds to the fact that there are two equal bits are the beginning, then the next two are equal, and then the next three are equal. This is not periodic because the array contains two different values (2 and 3).

You are given two integers L, R. Find the number of integers in the range $[L, R]$ (both inclusive) that are periodic.

Input

- The first line of the input contains an integer T denoting the number of test cases. The description of the test cases follows.
- The first line of each test case contains two space-separated integers L, R.

Output

For each test case, output a single line containing an integer corresponding to the number of periodic numbers in the range $[L, R]$.

Constraints

- $1 \leq T \leq 10^5$
- $1 \leq L, R \leq 10^9$

Example Input

```
2
3 3
1 10
```

Example Output

```
1
6
```

Explanation

Testcase 1: The only number between L and R is 3, which is periodic. Hence the answer is 1.

Testcase 2: The periodic numbers between 1 and 10 are 1, 2, 3, 5, 7, 10. Since there are six of them, then answer is 6.

Pseudo Sorting Algorithm (SORTALGO)

Chef has devised a new algorithm which works as follows. It takes an array a with n integers ($a_0, a_1, \dots, a_{n-1}$), and returns another array b.

pseudo_sort(a):

- Array b is initially empty.
- For i = 0; i $\leq$ n-1; i++:
  - If b is empty, append a_i to b.
  - Otherwise check if $a_i \geq$ last element of b
    - If yes:
      - Append a_i to b.
    - If no:
      - Do nothing and continue
- Return b

This algorithm will return the array b whose elements are sorted in non-decreasing order.

Your task is to delete at most one element from the array a such that after calling this function on the modified array a, the length of the returned array b is maximum possible. Return the maximum possible length of the returned array.

Input

- The first line of the input contains an integer T denoting the number of test cases. The description of the test cases follows.
- The first line of each test case contains an integer n.
- The second line of each test case contains n space-separated integers denoting the array a.

Output

For each test case, output the answer in a new line.

Constraints

- $1 \leq T \leq 10^5$
- $1 \leq n \leq 10^5$
- $1 \leq a_i \leq 10^5$
- Sum of n over all the test cases doesn't exceed 10^6

Example Input

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

Example Output

```
3
4
3
```

Explanation

Testcase 1: Without deleting any element, when the algorithm is run on the original array, the returned array is $b = \{1, 2, 3\}$. This is the longest array possible, and hence the answer is 3.

Testcase 2: After deleting the second element, the array a becomes $\{1, 2, 2, 4\}$. When the algorithm is run on this array , the returned array is $b = \{1, 2, 2, 4\}$. This is the longest array possible, and hence the answer is 4.

Testcase 3: After deleting the third element, the array a becomes $\{1, 3, 2, 3\}$. When the algorithm is run on this array , the returned array is $b = \{1, 3, 3\}$. This is the longest array possible, and hence the answer is 3.

Points, Lines, Triangles and Graphs (POLITRI)

You are given an integer k. You want to choose n points on the plane and join some pairs of those points with line segments (let us call them edges), such that :

- $n < 510$
- Any two points are distinct
- No three points are collinear
- The points have non-negative integer coordinates $\leq 10^9$.
- No two edges (drawn line segments) intersect at a point which is not an endpoint of either edge.
- The number of triangles in the figure is exactly k.

Note : A triangle means a triple of points such that there is an edge between any two of them.

It is guaranteed that it is possible to select a set of points and line segments satisfying the above conditions.

Input:

- The only line of the input contains k, the number of required triangles.

Output:

- In first line print two integers, n and m, the number of chosen points and number of line segments (edges) drawn between them. They must satisfy $1 \leq n < 510$ and $0 \leq m \leq \frac{n(n-1)}{2}$
- i^{th} of the next n lines should contain two space separated integers, x_i, y_i denoting the i^{th} chosen point. The coordinates must be non-negative and $\leq 10^9$, i.e. $0 \leq x_i, y_i \leq 10^9$.
- i^{th} of the next m lines should contain two integers u_i and v_i, denoting a line segment connecting the u_i^{th} chosen point and the v_i^{th} chosen point. $1 \leq u_i, v_i \leq n$ should be satisfied. No edge should be printed more than once.

If there are multiple solutions, you can print any of them.

Constraints

- $1 \leq k \leq 1500$

Sample Input:

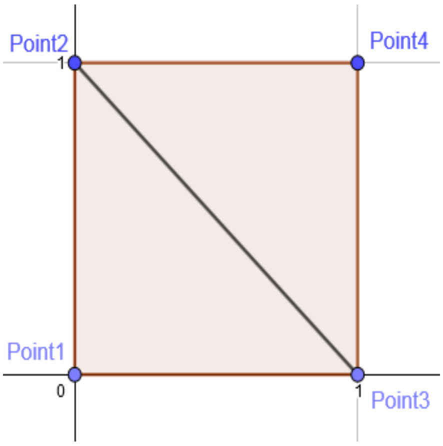
```
2
```

Sample Output:

```
4 5
0 0
0 1
1 0
1 1
1 2
1 3
2 3
2 4
3 4
```

Explanation:

The figure shows the configuration corresponding to the given output. It contains 4 points, and 5 edges. Clearly there are 2 triangles, as the input requires it to be.



Subset Sums Revisited (SUBSUMX)

You are given a sequence A of N integers: $(A_1, A_2, \ldots, A_N)$. From this sequence, you create another sequence of size N: $B = (B_1, B_2, \ldots, B_N)$, where $B_i = 2^{A_i}$.

You are also given an integer K. You need to output the number of subsequences of B whose sum is exactly K. Since the answer might be huge, output it modulo $10^9 + 7$.

Input

- The first line contains the number of testcases in file T
- Each testcase is described as follows:
- The first line contains two integers, N and K.
- The second line contains N space separated integers: $A_1, A_2, \ldots, A_N$

Output

Output T lines each containing a single integer which should be the answer.

Constraints

- $1 \leq T \leq 200$
- $1 \leq N \leq 100$
- $0 \leq A_i \leq 60$
- $1 \leq K < 2^{61}$

Sample Input

```
1
3 8
2 2 2
```

Sample Output

```
3
```

Explanation

The sequence A is $(2, 2, 2)$. Hence the sequence B is $(4, 4, 4)$. We can pick any two of these to get a sum of 8. Hence the answer is 3.

A Triangle and Two Squares (SQRTRI)

You are given two squares A and B. The square A has a side length of a and B has a side length of b. The left-bottom point of square A is at $(0, 0)$ and the top-right at (a, a). Square B's left-bottom point is (x, y) and top-right is $(x + b, y + b)$. It's guaranteed that square B lies inside square A (may not be strictly inside, can touch too). In other words, $0 \leq x, 0 \leq y, (x + b \leq a, y + b \leq a)$.

You have to tell whether you can construct a triangle T such that

- All the vertices of the triangle lie on the boundary of square A.
- One of its sides is parallel to one of the sides of the square A and this side should contain one of the sides of square B as a subsegment. That is, there should be a side of the triangle, say T_2T_3, which is parallel to one of the sides of square A, and which contains a side of square B, say Q_3Q_4. That is, the line segment Q_3Q_4 should lie within the line segment T_2T_3.
- Square B is inside the triangle T (is allowed to touch the sides of T, but shouldn't go outside the triangle T)

Input

- The first line of the input contains an integer T denoting the number of test cases. The description of the test cases follows.
- The first line of each test case contains four space-separated integers a, b, x, y.

Output

For each test case, print a single line containing the string `yes` if it is possible to construct such a triangle, or `no` if it is impossible.

Constraints

- $1 \leq T \leq 10^5$
- $1 \leq b \leq a \leq 10000$
- $0 \leq x, y \leq a - b$

Example Input

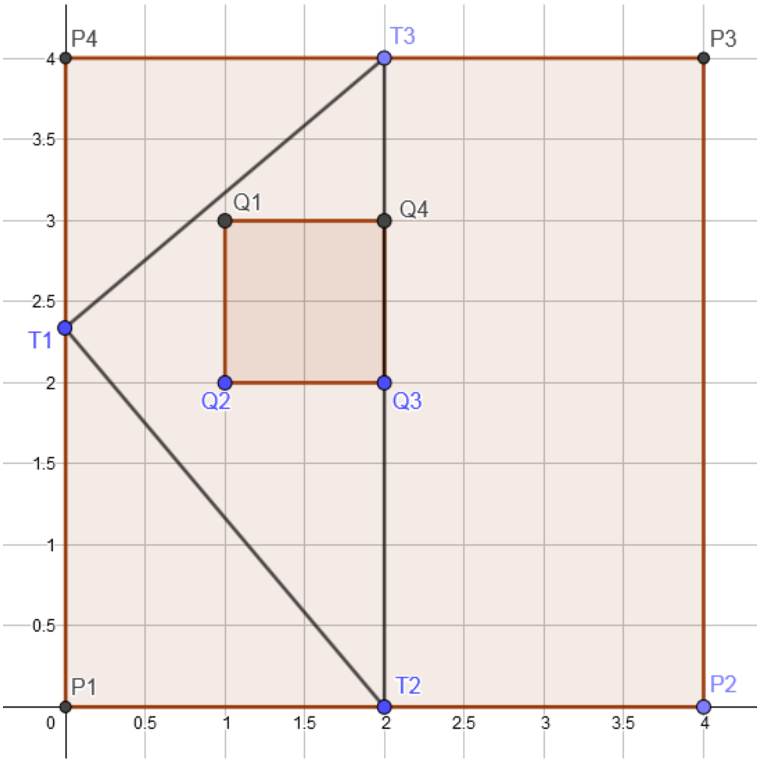
```
4
4 1 1 2
3 1 1 2
3 1 0 0
3 2 1 1
```

Example Output

```
yes
yes
yes
no
```

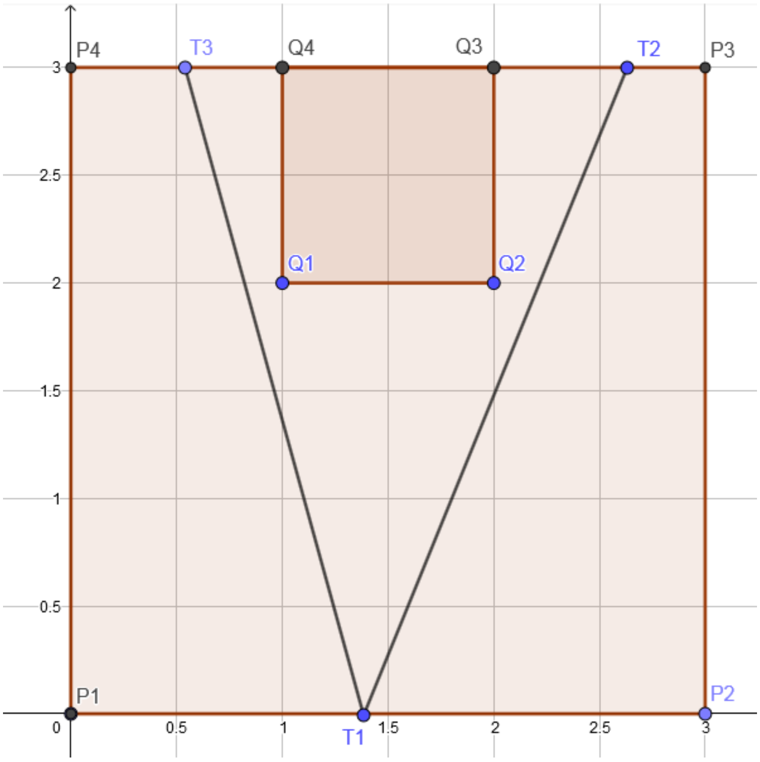
Explanation

Testcase 1: The following figure shows one possible way in which the triangle can be constructed:



Square A is P1 P2 P3 P4, square B is Q1 Q2 Q3 Q4 and the constructed triangle T is T1 T2 T3.

Testcase 2: The following figure shows one possible way in which the triangle can be constructed:



Square A is P1 P2 P3 P4, square B is Q1 Q2 Q3 Q4 and the constructed triangle T is T1 T2 T3.

Another Game of Life (VASTLIFE)

In 1970, John Conway created a cellular automaton, the Game of Life.

In this problem we will consider a modified version of the Game of Life called VastoLorde's Game of Life, and we will be focusing on just one single second of change.

Suppose you start with grid A which has dimensions $H \times W$. The top left cell being $A[1][1]$ and the bottom right cell being $A[H][W]$. Each cell (i, j) of this grid is either "dead" or "alive". An alive cell is represented by 1 and a dead cell by 0.

When Vasto clicks his fingers, this grid changes into a different grid, B. B's dimensions are $(H-1) \times (W-1)$. The state of the (i, j)-th cell of B is determined by the following rule:

Consider cells $A[i][j]$, $A[i+1][j]$, $A[i][j+1]$ and $A[i+1][j+1]$ i.e. the 2x2 block with (i, j) at the top left corner. If exactly one diagonal of this 2x2 block is alive and the other two cells are dead, then the cell $B[i][j]$ will be alive. Otherwise, $B[i][j]$ will be dead.

Formally, if either ($A[i][j] = 1, A[i+1][j+1] = 1, A[i+1][j] = 0, A[i][j+1] = 0$), or ($A[i][j] = 0, A[i+1][j+1] = 0, A[i+1][j] = 1, A[i][j+1] = 1$), then $B[i][j] = 1$. If neither of these two configurations are present, then $B[i][j] = 0$.

For example, suppose $H = 3$, $W = 3$, and grid A is

```
101
011
101
```

This would change into the the following 2 x 2 B grid:

```
10
10
```

Your task is, given grid B, determine the number of possible grids A that will generate this grid when Vasto clicks his finger.

Print your answer modulo $10^9 + 7$

Input:

- The first line of input contains two integers W and H, the width and height of B. H lines follow.
- The i-th line contains W space separated integers $B[i][1], B[i][2], \ldots, B[i][W]$ such that the cell $B[i][j]$ of the grid is alive if $B[i][j] = 1$ and dead if $B[i][j] = 0$

Output:

Print a single integer, the number of grids A that create the given grid B modulo $10^9 + 7$

Constraints

- $2 \leq W \leq 500$
- $2 \leq H \leq 10$
- $B[i][j] \in \{0, 1\}$

Sample Input:

```
1
2 2
1 1
1 1
```

Sample Output:

```
2
```

Explanation:

There are two 3x3 grids that A could be, so that it changes into the 2x2 grid in the input. They are

```
101
010
101
```

and

```
010
101
010
```