The 3rd Universal Cup

Stage 21: Ōokayama

December 14-15, 2024

This problem set should contain 13 problems on 28 numbered pages.

Prepared by

Problem A. Don't Detect Cycle

Time limit: 2 seconds
Memory limit: 1024 megabytes

You are given a graph G consisting of N vertices numbered $1, 2, \ldots, N$. Initially, G has no edges.

You will add M undirected edges to G. The final shape of the graph is predetermined, and the i-th edge to be added ($1 \le i \le M$) connects vertices u_i and v_i. We will refer to this as edge i. It is guaranteed that the resulting graph will be simple.

Determine if there exists a permutation $(P_1, P_2, \ldots, P_M)$ of $(1, 2, \ldots, M)$ that satisfies the following conditions, and if such a permutation exists, show an example.

Conditions

You must be able to add all M edges to G by following this procedure:

- For $i = 1, 2, \ldots, M$, repeat the following:
  1. If there is already a cycle in G containing either vertex u_{P_i} or vertex v_{P_i}, the condition is not satisfied, and the procedure ends.
  2. Add edge P_i (the undirected edge connecting u_{P_i} and v_{P_i}) to G.

You are given T test cases; solve each of them.

Input

The input is given in the following format:

```
T
case_1
case_2
⋮
case_T
```

Here, case_i ($1 \le i \le T$) represents the i-th test case. Each test case is given in the following format:

```
N M
u_1 v_1
u_2 v_2
⋮
u_M v_M
```

- All input values are integers.
- $1 \le T \le 2000$
- For each test case:
  - $2 \le N \le 4000$
  - $1 \le M \le 4000$
  - $1 \le u_i, v_i \le N$ $(1 \le i \le M)$
  - The graph formed by adding all given edges is simple.
- The sum of N over all test cases is at most 4000.

- The sum of M over all test cases is at most 4000.

Output

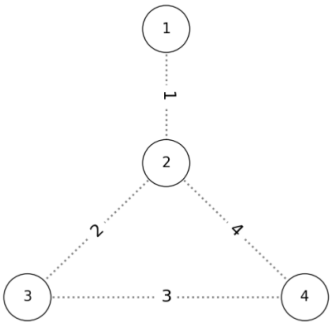
For each test case, if a permutation $(P_1, P_2, \ldots, P_M)$ satisfying the conditions exists, output such a P separated by spaces. If no such permutation exists, output `-1`.

Examples

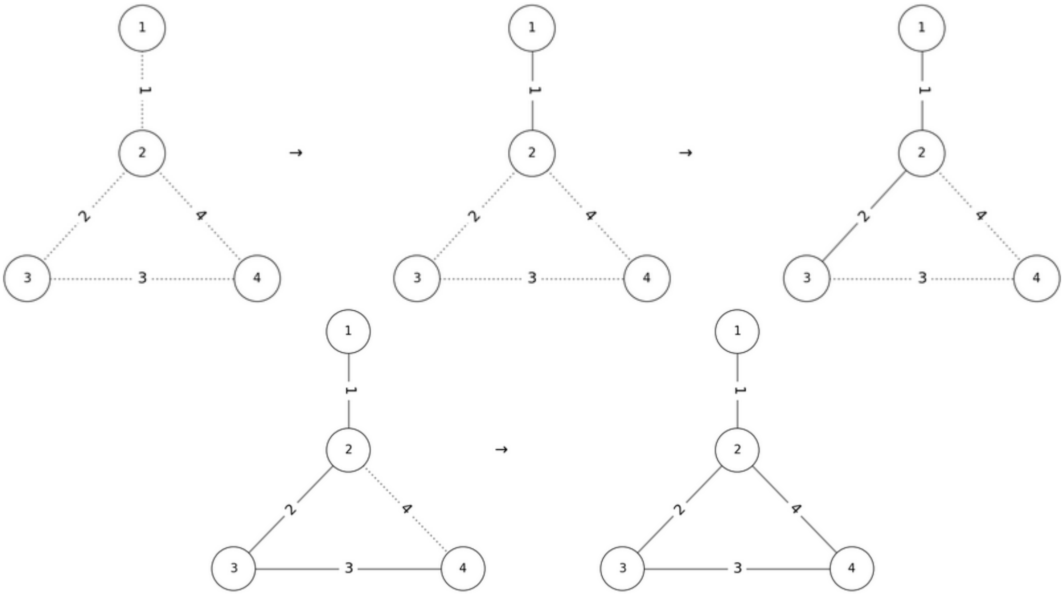
standard input	standard output
1 4 4 1 2 2 3 3 4 4 2	1 4 3 2
4 4 5 1 2 2 3 3 4 3 1 1 4 5 3 1 2 2 3 3 4 9 10 3 5 1 8 5 8 4 9 6 7 7 9 1 2 1 4 2 4 4 6 8 10 1 4 3 8 2 5 3 4 1 5 5 8 2 8 5 7 4 5 3 7	-1 1 2 3 1 2 3 4 8 9 10 7 6 5 -1

Note

In the first example, the given graph has the following shape:

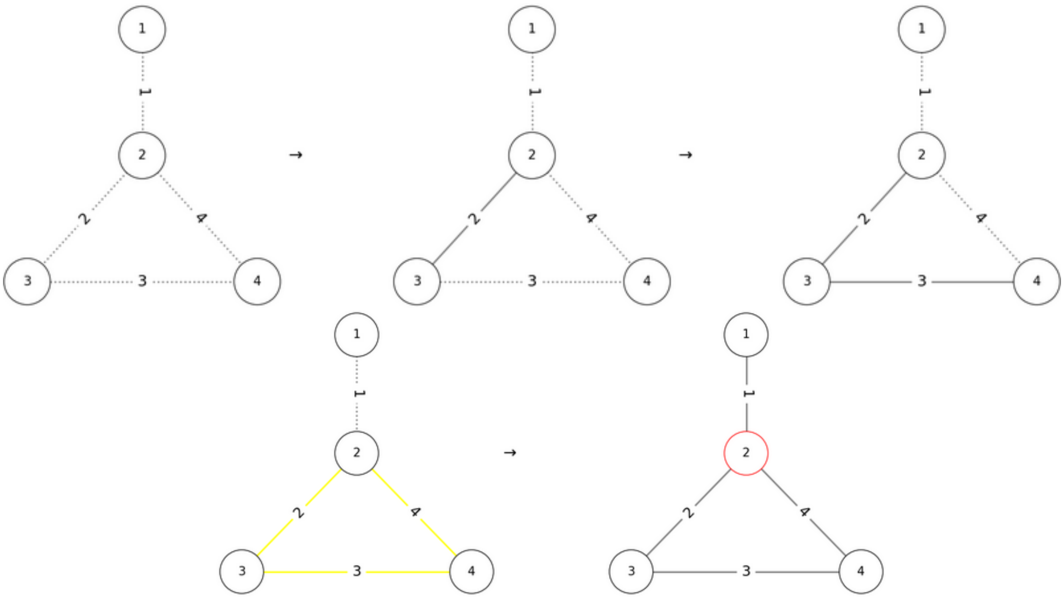


If we add the edges in the order $P = (1, 2, 3, 4)$, the conditions are satisfied as shown below:



Thus, "`1 2 3 4`" is one valid output.

However, if we add edges in the order $P = (2, 3, 4, 1)$, a cycle containing vertex 2 is created before edge 1 can be added, so the conditions are not satisfied.



Other valid outputs include $P = (1, 4, 3, 2)$ or $P = (2, 4, 1, 3)$.

Note that the graph is not necessarily connected.

Problem B. Self Checkout

Time limit: 2 seconds
Memory limit: 1024 megabytes

You are given a sequence S of length N, consisting of integers 1, 2, and 3. Determine the number of sequences A consisting of integers 1 and 2 such that the sequence T obtained after performing the following procedure matches S. Output the answer modulo 998244353. It can be proven that the number of such sequences A is finite.

- Start with an empty sequence T. Given a sequence A consisting of integers 1 and 2, perform the following process to obtain the sequence T:
  1. Set a variable $C = 0$.
  2. If A contains at least one 1, remove the first occurrence of 1 from A and add 1 to C.
  3. If A is not empty, remove the first element x of A and add x to C.
  4. Append C to the end of T.
  5. If A is empty, terminate the process. Otherwise, return to step 1.

Input

The input is given in the following format:

```
N
S_1 S_2 ... S_N
```

- All input values are integers.
- $1 \leq N \leq 10^6$
- $1 \leq S_i \leq 3$

Output

Output the number of sequences A that satisfy the conditions, modulo 998244353.

Examples

standard input	standard output
2 3 2	5
6 3 2 2 3 2 1	4
5 3 2 1 3 2	0

Note

In the first example, there are 5 possible sequences A that result in $T = (3, 2)$:

$A = (1, 2, 2), (2, 1, 2), (2, 2, 1), (2, 1, 1, 1), (1, 2, 1, 1)$.

For example, for $A = (2, 1, 1, 1)$, the process proceeds as follows:

- Remove the first occurrence of 1 in A, which is $A_2 = 1$. Now $A = (2, 1, 1)$ and $C = 1$.

- Remove the first element of A, which is $A_1 = 2$. Now $A = (1, 1)$ and $C = 3$.
- Append C to the end of T. Now $T = (3)$.
- Remove the first occurrence of 1 in A, which is $A_1 = 1$. Now $A = (1)$ and $C = 1$.
- Remove the first element of A, which is $A_1 = 1$. Now $A = ()$ and $C = 2$.
- Append C to the end of T. Now $T = (3, 2)$.

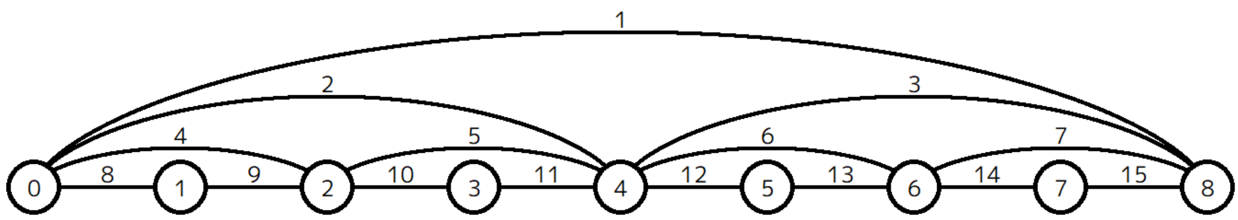
Problem C. Segment Tree

Time limit: 6 seconds
Memory limit: 1024 megabytes

You are given an undirected graph G with $2^N + 1$ vertices and $2^{N+1} - 1$ edges. The vertices are numbered $0, 1, \ldots, 2^N$, and the edges are numbered $1, 2, \ldots, 2^{N+1} - 1$.

Each edge in G belongs to one of $N + 1$ types, ranging from type 0 to type N. For type i $(0 \le i \le N)$, there are exactly 2^i edges, which are numbered $2^i + 0, 2^i + 1, \ldots, 2^i + (2^i - 1)$. The edge numbered $2^i + j$ $(0 \le j \le 2^i - 1)$ is an undirected edge of length C_{2^i+j} that connects vertex $j \times 2^{N-i}$ and vertex $(j + 1) \times 2^{N-i}$.

For example, when $N = 3$, G looks like the following graph:



You are given Q queries to process. There are two types of queries:

- `1 j x`: Change the length of edge j to x.
- `2 s t`: Find the shortest path length from vertex s to vertex t.

Input

The input is given in the following format. Note that vertex numbering starts from 0, while edge numbering starts from 1.

N
$C_1\ C_2\ \cdots\ C_{2^{N+1}-1}$
Q
query_1
query_2
$\vdots$
query_Q

Here, query_i represents the i-th query. Each query is given in one of the following formats:

1 j x

2 s t

- All input values are integers.
- $1 \le N \le 18$
- $1 \le C_j \le 10^7$ $(1 \le j \le 2^{N+1} - 1)$

- $1 \le Q \le 2 \times 10^5$
- In the query `1 j x`, $1 \le j \le 2^{N+1} - 1$ and $1 \le x \le 10^7$.
- In the query `2 s t`, $0 \le s < t \le 2^N$.
- There is at least one `2 s t` query.

Output

Let m be the number of queries of type `2 s t`. Output m lines, where the i-th line contains the answer to the i-th `2 s t` query.

Example

standard input	standard output
3 7 1 14 3 9 4 8 2 6 5 5 13 8 2 3 10 2 0 1 2 0 4 2 4 6 2 4 8 2 3 5 1 6 30 2 3 5 2 4 6 1 1 10000000 2 0 8	2 1 4 8 17 18 13 15

Note

- In the first query, using edge 8, the path $0 \to 1$ results in a total distance of 2.
- In the second query, using edge 2, the path $0 \to 4$ results in a total distance of 1.
- In the third query, using edge 6, the path $4 \to 6$ results in a total distance of 4.
- In the fourth query, using edges $2, 1$, the path $4 \to 0 \to 8$ results in a total distance of 8.
- In the fifth query, using edges $11, 6, 13$, the path $3 \to 4 \to 6 \to 5$ results in a total distance of 17.
- In the sixth query, the length of edge 6 is updated from 4 to 30.
- In the seventh query, using edges $11, 12$, the path $3 \to 4 \to 5$ results in a total distance of 18.
- In the eighth query, using edges $2, 1, 15, 14$, the path $4 \to 0 \to 8 \to 7 \to 6$ results in a total distance of 13.
- In the ninth query, the length of edge 1 is updated from 7 to 10000000.
- In the tenth query, using edges $2, 3$, the path $0 \to 4 \to 8$ results in a total distance of 15.

Problem D. A xor B plus C

Time limit: 8 seconds
Memory limit: 1024 megabytes

You are given non-negative integers A, B, and C. Define a sequence of non-negative integers $X = (X_1, X_2, \dots)$ as follows:

- $X_1 = A$
- $X_2 = B$
- $X_{i+2} = (X_i \oplus X_{i+1}) + C \quad (i = 1, 2, \dots)$

Here, $\oplus$ represents the bitwise XOR operation.

You are also given a positive integer N. Calculate $X_N \bmod 998244353$.

Input

A B C N

- All input values are integers.
- $0 \le A, B, C < 2^{20}$
- $1 \le N \le 10^{18}$

Output

Output $X_N \bmod 998244353$.

Examples

standard input	standard output
1 2 3 4	7
123 456 789 123456789	567982455
0 0 0 1000000000000000000	0

Note

In the first example, $X = (1, 2, 6, 7, \dots)$. Here, $X_4 = 7$ is the answer.

Problem E. ReTravel

Time limit: 2 seconds
Memory limit: 1024 megabytes

On the xy-plane, there are N points labeled $1, 2, \ldots, N$. The coordinates of point i ($1 \le i \le N$) are (X_i, Y_i).

There is a robot at the origin of this plane. Your task is to control the robot to visit all the points $1, 2, \ldots, N$ **in this order**.

The robot has a string variable S, which is initially an empty string. You can move the robot using the following four types of operations:

- **Operation 1:** Increase the robot's x coordinate by 1 and append `X` to the end of S. This operation costs 1.
- **Operation 2:** Increase the robot's y coordinate by 1 and append `Y` to the end of S. This operation costs 1.
- **Operation 3:** Decrease the robot's x coordinate by 1 and remove the last character of S. This operation can only be performed if the last character of S is `X`. This operation costs 0.
- **Operation 4:** Decrease the robot's y coordinate by 1 and remove the last character of S. This operation can only be performed if the last character of S is `Y`. This operation costs 0.

Find the minimum cost required for the robot to visit all points $1, 2, \ldots, N$ in this order.

Input

The input is given in the following format:

N
X_1 Y_1
X_2 Y_2
$\vdots$
X_N Y_N

- All input values are integers.
- $1 \le N \le 500$
- $0 \le X_i, Y_i \le 10^9$

Output

Output the minimum cost required for the robot to visit all points in order.

Examples

standard input	standard output
2 3 3 1 2	6
3 2 2 3 3 1 3	7

Note

In the first example, by performing **Operation 1** once, **Operation 2** three times, and **Operation 1** twice, you can reach point 1. Then, by performing **Operation 3** twice and **Operation 4** once, you can reach point 2.

The total cost of this sequence of operations is the sum of the number of times **Operations 1** and **2** are performed, which is 6.

Problem F. Origami Warp

Time limit: 3 seconds
Memory limit: 1024 megabytes

You are given N lines on the xy-plane. The i-th line ($1 \le i \le N$) passes through two distinct points (a_i, b_i) and (c_i, d_i). Specifically, $(a_1, b_1, c_1, d_1) = (0, 0, 1, 0)$ and $(a_2, b_2, c_2, d_2) = (0, 0, 0, 1)$. That is, the first line represents the x-axis, and the second line represents the y-axis.

Alice is on the xy-plane. She can perform the following operation any number of times (including zero):

> Choose one of the N given lines. Alice moves to the position symmetric to her current position with respect to the chosen line.

Additionally, we define that point P is **reachable** from point S if the following condition is satisfied:

> For any real number $\varepsilon > 0$, there exists a point Q such that the Euclidean distance between Q and P is at most ε, and Alice can move from S to Q in a finite number of operations.

Answer Q queries. For the i-th query ($1 \le i \le Q$), you are given integers X_i, Y_i, Z_i, W_i. Output `Yes` if (Z_i, W_i) is reachable from (X_i, Y_i). Otherwise, output `No`.

You are given T test cases; solve each of them.

Input

The input is given in the following format:

T
case_1
case_2
$\vdots$
case_T

Here, case_i represents the i-th test case. Each test case is given in the following format:

N
a_1 b_1 c_1 d_1
a_2 b_2 c_2 d_2
$\vdots$
a_N b_N c_N d_N
Q
X_1 Y_1 Z_1 W_1
X_2 Y_2 Z_2 W_2
$\vdots$
X_Q Y_Q Z_Q W_Q

- All input values are integers.
- $1 \le T \le 100$
- For each test case:
  - $2 \le N \le 2000$
  - $1 \le Q \le 2000$

- $-10^8 \le a_i, b_i, c_i, d_i \le 10^8$ $(1 \le i \le N)$
- $(a_i, b_i) \ne (c_i, d_i)$
- $(a_1, b_1, c_1, d_1) = (0, 0, 1, 0)$
- $(a_2, b_2, c_2, d_2) = (0, 0, 0, 1)$
- All given lines are distinct.
- $-10^8 \le X_i, Y_i, Z_i, W_i \le 10^8$ $(1 \le i \le Q)$

Output

For each test case, output Q lines. The i-th line $(1 \le i \le Q)$ should contain the answer to the i-th query. The output should be `Yes` if (Z_i, W_i) is reachable from (X_i, Y_i), otherwise `No`. The output is case-insensitive.

Example

standard input	standard output
2 3 0 0 1 0 0 0 0 1 0 2 2 0 4 1 0 2 3 1 -2 -1 2 1 1 -1 0 3 3 3 3 3 0 0 1 0 0 0 0 1 -2 1 2 3 2 2 1 -1 5 -1 -1 3 3	Yes Yes No Yes Yes Yes

Note

Let us explain the first test case. For the first query, Alice can use the second and third lines in sequence to move $(1, 0) \to (-1, 0) \to (2, 3)$. Thus, $(2, 3)$ is reachable from $(1, 0)$. Note that for the fourth query, if $(X_i, Y_i) = (Z_i, W_i)$, the destination is always reachable.

Now let us explain the second test case. For the first query, Alice can use the first and third lines in sequence to move $(2, 1) \to (2, -1) \to \left(-\frac{6}{5}, \frac{27}{5}\right)$. The distance between $(-1, 5)$ and $\left(-\frac{6}{5}, \frac{27}{5}\right)$ is $\frac{1}{\sqrt{5}}$. This means that for $\varepsilon \ge \frac{1}{\sqrt{5}}$, Alice can find a point Q that satisfies the "reachable" condition. Moreover, for any $\varepsilon > 0$, Alice can find such a point Q. As a result, $(2, 1)$ is reachable from $(-1, 5)$.

Problem G. Diverge and Converge

Time limit: 2 seconds
Memory limit: 1024 megabytes

You are given a tree A with N vertices. The vertices of A are numbered $1, 2, \ldots, N$, and the i-th edge $(1 \leq i \leq N-1)$ connects vertices u_i and v_i of A.

Additionally, you are given another tree B with N vertices. The vertices of B are also numbered $1, 2, \ldots, N$, and the j-th edge $(1 \leq j \leq N-1)$ connects vertices x_j and y_j of B.

Your task is to find a pair of permutations $((P_1, P_2, \ldots, P_{N-1}), (Q_1, Q_2, \ldots, Q_{N-1}))$ that satisfies the following conditions:

- Perform the following operations 1. and 2. in order for $k = 1, 2, \ldots, N-1$. After completing operations 1. and 2. **for each k**, both A and B must remain as trees.
  1. In tree A, remove the edge connecting vertices u_{P_k} and v_{P_k}, and add an edge connecting vertices x_{Q_k} and y_{Q_k}.
  2. In tree B, remove the edge connecting vertices x_{Q_k} and y_{Q_k}, and add an edge connecting vertices u_{P_k} and v_{P_k}.

It can be proven that under the constraints of this problem, a valid solution always exists.

Input

The input is given in the following format:

N
u_1 v_1
u_2 v_2
$\vdots$
u_{N-1} v_{N-1}
x_1 y_1
x_2 y_2
$\vdots$
x_{N-1} y_{N-1}

- All input values are integers.
- $2 \leq N \leq 1000$
- $1 \leq u_i, v_i, x_j, y_j \leq N$
- The given A and B form trees.

Output

Output the answer in the following format:

P_1 P_2 $\cdots$ P_{N-1}
Q_1 Q_2 $\cdots$ Q_{N-1}

Examples

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

Note

In the first example, before the operation, A is a path graph, and B is a star graph.

After the operation for $k = 1$, A becomes a star graph, and B becomes a path graph.

In the operations for $k = 2$, the same edge (in terms of vertex numbers) is removed and added, so the shape of the tree remains unchanged.

After completing the operations for $k = 3$, the original shapes of trees A and B are completely swapped.

Problem H. Two Convex Sets

Time limit: 4 seconds
Memory limit: 1024 megabytes

A set of points U in the xy-plane is called **good** if no point in U lies in the **interior** of the convex hull of U. Note that the empty set is considered good.

You are given N distinct points $v_1, v_2, \ldots, v_N$ in the xy-plane. The coordinates of the point v_i are (x_i, y_i). No three distinct points are collinear.

Count the number of subsets S of $V = \{v_1, v_2, \ldots, v_N\}$ such that both S and $V \setminus S$ are good sets.

Input

The input is given in the following format:

N
x_1 y_1
x_2 y_2
$\vdots$
x_N y_N

- All input values are integers.
- $1 \le N \le 40$
- $|x_i|, |y_i| \le 10^6$
- $(x_i, y_i) \ne (x_j, y_j)$ for $i \ne j$
- No three distinct points are collinear.

Output

Output the answer.

Examples

standard input	standard output
4 0 0 3 0 0 3 1 1	14
8 1 0 2 0 3 1 3 2 2 3 1 3 0 2 0 1	256
10 0 0 1 1 7 1 1 7 3 2 2 3 4 2 2 4 5 4 4 5	0

Note

In the first example, except for the empty set $\emptyset$ and the full set V, all other sets S satisfy the condition.

Problem I. Near Pair

Time limit: 2 seconds
Memory limit: 1024 megabytes

This is an **interactive problem**, where your program interacts with the judge system via Standard Input and Output.

You are given integers N, K, and Q. The judge holds a hidden permutation $(a_1, a_2, \ldots, a_N)$ of $(1, 2, \ldots, N)$. You may ask up to Q queries to the judge, where each query is as follows:

- Choose a subset S from $\{1, 2, \ldots, N\}$. The judge will return the number of distinct pairs (i, j) such that $i < j$ and $|a_i - a_j| \le K$ where $i, j \in S$.

Let x be the index t where $a_t = 1$, and y be the index t where $a_t = N$. Your task is to identify the set $\{x, y\}$. (You do not need to distinguish which is x and which is y.)

The judge is non-adaptive, meaning the permutation $(a_1, a_2, \ldots, a_N)$ is fixed before any interaction begins.

Input

- All input values are integers.
- $N = 20000$
- $1 \le K \le 10$
- $Q = 30(K + 1)$

Interaction Protocol

First, read integers N, K, and Q from the Standard Input:

```
N K Q
```

Then, repeat the following process until you identify the set $\{x, y\}$:

For a query, output the following format to the Standard Output:

```
? s_1s_2...s_N
```

Here, $s_1 s_2 \ldots s_N$ is a binary string of length N representing the subset S, where $s_i = 1$ if $i \in S$, and $s_i = 0$ otherwise.

The response to your query will be provided in the Standard Input in the following format:

```
T
```

Here, T is the answer to the query, representing the number of distinct pairs (i, j) such that $i < j$ and $|a_i - a_j| \le K$ where $i, j \in S$.

Once you identify the set $\{x, y\}$, output the two elements in the following format (order does not matter) and terminate your program immediately:

```
! x y
```

Note

- **Print a newline and flush Standard Output at the end of each message. Otherwise, you may get a Time Limit Exceeded verdict.**
- If your query format is invalid or you exceed the number of queries, the judge will respond with $T = -1$. If you receive this response, you should immediately terminate your program. Otherwise, you may get a Time Limit Exceeded verdict.
- Beware that unnecessary newlines are considered as malformed.

Sample Interaction

The following is a sample interaction for $N = 5$, $K = 2$, $Q = 90$. Note that this example does not meet the constraints and will not appear in the test cases.

Input	Output	Explanation
		The judge holds the permutation $(3, 5, 2, 1, 4)$.
`5 2 90`		First, integers N, K, and Q are provided.
	`? 11000`	You query with $S = \{1, 2\}$.
`1`		Only the pair $(1, 2)$ satisfies the condition, so the judge responds with 1.
	`? 10011`	You query with $S = \{1, 4, 5\}$.
`2`		The pairs $(1, 4)$ and $(1, 5)$ satisfy the condition, so the judge responds with 2.
	`! 2 4`	You output $\{2, 4\}$ as the answer. Since $a_4 = 1$ and $a_2 = N$, this output is correct.

Problem J. Grid Construction

Time limit: 2 seconds
Memory limit: 1024 megabytes

You are given positive integers H and W. You aim to draw an H-row by W-column grid on the coordinate plane using several "U-shapes."

To draw one U-shape, perform the following operation:

- Choose integers $1 \le x \le H$ and $1 \le y \le W$.
- From the following four line segments, select any three distinct ones:
  - The line segment connecting $(x-1, y-1)$ and $(x-1, y)$,
  - The line segment connecting $(x-1, y-1)$ and $(x, y-1)$,
  - The line segment connecting (x, y) and $(x-1, y)$,
  - The line segment connecting (x, y) and $(x, y-1)$.
- Draw the selected three line segments on the coordinate plane.

However, the line segments you draw must not share any points (other than endpoints) with any line segments drawn previously.

Is it possible to draw all the length-1 line segments connecting grid points with $0 \le x \le H$ and $0 \le y \le W$ by repeating this operation? If possible, provide an example.

Input

The input is given in the following format:

```
H W
```

- All input values are integers.
- $1 \le H, W \le 1000$

Output

If it is impossible to draw the grid by repeating the operation, output `No`.

If it is possible, output such an operation sequence in the following format:

```
Yes
S_1
⋮
S_H
```

Here, $S_1, \ldots, S_H$ are strings of length W, and the j-th character of S_i ($1 \le i \le H$, $1 \le j \le W$) is defined as follows:

- If there is no operation with $(x, y) = (i, j)$, the j-th character of S_i is '`.`'.
- Otherwise, there is exactly one operation with $(x, y) = (i, j)$. In that operation:
  - If the three selected line segments are those *excluding* "the line segment connecting $(x-1, y-1)$ and $(x-1, y)$," then the j-th character of S_i is '`v`'.

- If the three selected line segments are those *excluding* "the line segment connecting $(x-1, y-1)$ and $(x, y-1)$," then the j-th character of S_i is '>'.
- If the three selected line segments are those *excluding* "the line segment connecting (x, y) and $(x-1, y)$," then the j-th character of S_i is '<'.
- If the three selected line segments are those *excluding* "the line segment connecting (x, y) and $(x, y-1)$," then the j-th character of S_i is '^'.

Note that the judge is case-insensitive for `Yes` and `No`.

Refer to the sample inputs and the visualizer for further clarification.

Examples

standard input	standard output

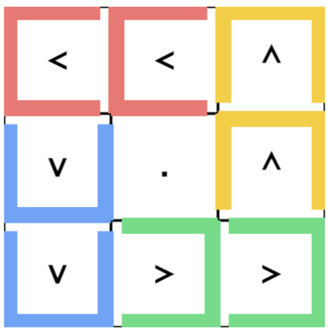
`3 3`	`Yes`
	`<<^`
	`v.^`
	`v>>`

`4 4`	`No`
`4 5`	`No`

Note

You can download the visualizer from the "Attachments" of this problem.

In the first example, as shown in the figure, you can draw a 3×3 grid by drawing U-shapes. Note that no U-shape is drawn at the location corresponding to the center cell. (For clarity, the U-shapes are colored, but this is irrelevant to the problem.)



Problem K. Sum is One

Time limit: 2 seconds
Memory limit: 1024 megabytes

You are given a sequence $A = (A_1, A_2, \ldots, A_N)$ of length N consisting of 0s and 1s. Define a simple undirected graph $G = (V, E)$ with $\frac{N(N-1)}{2}$ vertices as follows:

- For every pair of integers (i, j) satisfying $1 \le i < j \le N$, $(i, j) \in V$. This vertex is called vertex (i, j).
- For every triple of integers (i, j, k) satisfying $1 \le i < j < k \le N$ and $A_i + A_j + A_k = 1$, there is an edge connecting vertex (i, j) and vertex (j, k).
- No other pairs of vertices have edges.

Determine the number of connected components in G.

You are given T test cases. For each test case, compute the answer.

Input

The input is given in the following format:

$$
\begin{array}{l}
T \\
\text{case}_1 \\
\text{case}_2 \\
\vdots \\
\text{case}_T
\end{array}
$$

Here, case_i represents the i-th test case. Each test case is given in the following format:

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

- All input values are integers.
- $1 \le T \le 10^5$
- $3 \le N \le 10^6$
- A_i is 0 or 1 $(1 \le i \le N)$
- The total sum of N over all test cases in a single input is at most 10^6.

Output

For each test case, output the answer.

Example

standard input	standard output
4 5 1 0 0 1 0 5 1 1 1 1 1 12 0 0 1 1 1 0 0 0 1 0 1 0 20 0 0 1 0 0 1 1 1 0 0 1 0 0 1 1 1 1 0 1 1	4 10 13 58

Note

In the first test case, the connected components are as follows:

- $\{(1, 2), (2, 3), (2, 4), (2, 5), (3, 4), (4, 5)\}$
- $\{(1, 3), (3, 5)\}$
- $\{(1, 4)\}$
- $\{(1, 5)\}$

In the second test case, the graph has no edges, so the number of connected components is 10.

Problem L. Long Sequence Inversion 2

Time limit: 2 seconds
Memory limit: 1024 megabytes

In this problem, when we refer to a "permutation of length K", we mean a permutation of $(0, 1, \ldots, K-1)$. Also, we denote the k-th element (0-based) of a sequence X as $X[k]$.

You are given a permutation P of length L, and L permutations of length B, denoted as $V_0, V_1, \ldots, V_{L-1}$. We define a sequence A of length B^L as follows:

For each $0 \le n < B^L$, when n is represented as an L-digit number in base B, let $N[i]$ be the value at the B^i place $(0 \le i < L)$. Then,

$$A[n] = \sum_{i=0}^{L-1} V_i[N[i]] \cdot B^{P[i]}$$

Find the inversion number of the sequence A modulo 998244353.

Input

The input is given in the following format:

L B
$P[0]$ $P[1]$ $\cdots$ $P[L-1]$
$V_0[0]$ $V_0[1]$ $\cdots$ $V_0[B-1]$
$\vdots$
$V_{L-1}[0]$ $V_{L-1}[1]$ $\cdots$ $V_{L-1}[B-1]$

- All input values are integers.
- $1 \le L$
- $2 \le B$
- $L \times (B+1) \le 5 \times 10^5$
- P is a permutation of length L.
- For each $0 \le i < L$, V_i is a permutation of length B.

Output

Output the inversion number of the sequence A modulo 998244353.

Examples

standard input	standard output
3 2 2 0 1 1 0 1 0 0 1	14
2 4 1 0 2 0 3 1 1 2 3 0	60
9 10 2 5 7 3 8 1 4 6 0 9 2 4 0 1 6 7 3 5 8 4 1 6 7 8 0 5 9 2 3 1 9 2 4 6 8 5 7 0 3 9 0 8 2 5 1 6 7 3 4 1 6 0 7 3 9 2 4 5 8 4 5 2 9 1 6 7 3 0 8 7 0 5 6 1 9 2 4 3 8 3 2 1 6 7 0 8 9 4 5 9 2 4 3 5 8 0 6 7 1	138876070

Note

In the first example, $n = 5$ corresponds to $N = (1, 0, 1)$ in base $B = 2$. Thus,

$$A[5] = V_0[1] \cdot 2^{P[0]} + V_1[0] \cdot 2^{P[1]} + V_2[1] \cdot 2^{P[2]} = 3.$$

By calculating similarly, we get $A = (5, 1, 4, 0, 7, 3, 6, 2)$. The inversion count of A is 14, so the output is 14.

In the second example, $A = (9, 1, 13, 5, 10, 2, 14, 6, 11, 3, 15, 7, 8, 0, 12, 4)$. The inversion count of A is 60, so the output is 60.

Problem M. Cartesian Trees

Time limit: 3 seconds
Memory limit: 1024 megabytes

You are given a permutation $A = (A_1, A_2, \ldots, A_N)$ of $(1, 2, \ldots, N)$.

For a pair of integers l, r $(1 \le l \le r \le N)$, we define the **Cartesian Tree** $\mathrm{C}(l, r)$ as follows:

- $\mathrm{C}(l, r)$ is a rooted binary tree with $r - l + 1$ nodes. We denote the root of this tree as rt.
- Let m be the unique integer such that $A_m = \min\{A_l, A_{l+1}, \ldots, A_r\}$.
- If $l < m$, then the left subtree of rt is $\mathrm{C}(l, m-1)$. If $l = m$, then rt has no left child.
- If $m < r$, then the right subtree of rt is $\mathrm{C}(m+1, r)$. If $m = r$, then rt has no right child.

You are given Q pairs of integers $(l_1, r_1), (l_2, r_2), \ldots, (l_Q, r_Q)$. Determine how many different Cartesian Trees are there among $\mathrm{C}(l_1, r_1), \mathrm{C}(l_2, r_2), \ldots, \mathrm{C}(l_Q, r_Q)$.

Two Cartesian Trees X and Y are considered the same if and only if all of the following conditions are satisfied:

- Let the root of X be rt_X, and the root of Y be rt_Y.
- If rt_X has a left child, then rt_Y also has a left child, and the left subtrees of rt_X and rt_Y are the same Cartesian Tree.
- If rt_X has no left child, then rt_Y also has no left child.
- If rt_X has a right child, then rt_Y also has a right child, and the right subtrees of rt_X and rt_Y are the same Cartesian Tree.
- If rt_X has no right child, then rt_Y also has no right child.

Input

The input is given in the following format:

N
$A_1\ A_2\ \ldots\ A_N$
Q
$l_1\ r_1$
$l_2\ r_2$
$\vdots$
$l_Q\ r_Q$

- All input values are integers.
- $1 \le N \le 4 \times 10^5$.
- A is a permutation of $(1, 2, \ldots, N)$.
- $1 \le Q \le 4 \times 10^5$.
- $1 \le l_i \le r_i \le N$ $(1 \le i \le Q)$.
- $(l_i, r_i) \ne (l_j, r_j)$ $(1 \le i < j \le Q)$.

Output

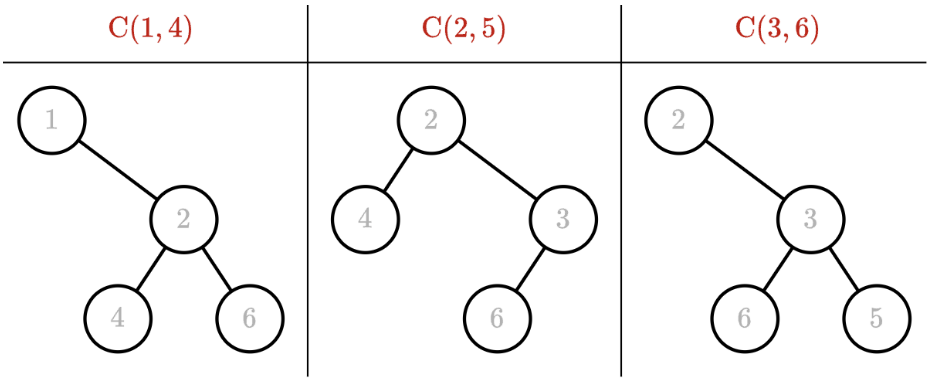
Print the number of different Cartesian Trees among the given pairs $(l_1, r_1), (l_2, r_2), \ldots, (l_Q, r_Q)$.

Examples

standard input	standard output
6 1 4 2 6 3 5 3 1 4 2 5 3 6	2
4 1 2 3 4 10 1 1 2 2 3 3 4 4 1 2 2 3 3 4 1 3 2 4 1 4	4
10 3 8 4 7 2 5 9 10 1 6 13 5 8 2 6 7 9 3 8 3 5 2 4 4 6 1 9 3 7 6 9 2 10 4 9 3 9	11

Note

In the first example, $\mathrm{C}(1,4), \mathrm{C}(2,5), \mathrm{C}(3,6)$ are the following Cartesian Trees:



C(1, 4) and C(3, 6) are the same Cartesian Tree, while C(2, 5) is a different Cartesian Tree from these. Therefore, there are 2 different Cartesian Trees.