A - Twice Subsequence

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 400 points

Problem Statement

There is a sequence $A = (A_1, \ldots, A_N)$. Determine whether there are at least two subsequences of A that match the sequence $B = (B_1, \ldots, B_M)$. Two subsequences are distinguished if they are taken from different positions, even if they coincide as sequences.

▶ Subsequence

Constraints

- $1 \leq M \leq N \leq 2 \times 10^5$
- $1 \leq A_i \leq 10^9$
- $1 \leq B_i \leq 10^9$
- All input values are integers.

Input

The input is given from Standard Input in the following format:

$$
\begin{array}{l}
N \quad M \\
A_1 \quad A_2 \quad \ldots \quad A_N \\
B_1 \quad B_2 \quad \ldots \quad B_M
\end{array}
$$

Output

If there are at least two subsequences of A that match B, print `Yes`. Otherwise, print `No`.

Sample Input 1

```
4 2
1 2 1 2
1 2
```

Sample Output 1

```
Yes
```

There are three subsequences of A that match B: $(A_1, A_2), (A_1, A_4), (A_3, A_4)$.

Sample Input 2

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

Sample Output 2

```
No
```

There is only one subsequence of A that matches B: (A_1, A_2).

Sample Input 3

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

Sample Output 3

```
No
```

There are no subsequences of A that match B.

B - Uniform Sum

Time Limit: 5 sec / Memory Limit: 1024 MiB

Score : 500 points

Problem Statement

There are two sequences $A = (A_1, \ldots, A_N)$ and $B = (B_1, \ldots, B_N)$. You can perform the following three types of operations any number of times in any order:

- Choose an index i such that $A_i = -1$, and replace A_i with any non-negative integer.
- Choose an index i such that $B_i = -1$, and replace B_i with any non-negative integer.
- Rearrange the elements of sequence A in any order.

Determine whether it is possible, after these operations, for all elements of A and B to be non-negative and satisfy $A_1 + B_1 = A_2 + B_2 = \cdots = A_N + B_N$.

Constraints

- $2 \leq N \leq 2000$
- $-1 \leq A_i \leq 10^9$
- $-1 \leq B_i \leq 10^9$
- All input values are integers.

Input

The input is given from Standard Input in the following format:

$$
\begin{array}{l}
N \\
A_1 \ A_2 \ \ldots \ A_N \\
B_1 \ B_2 \ \ldots \ B_N
\end{array}
$$

Output

If it is possible, after the operations, for all elements of A and B to be non-negative and satisfy $A_1 + B_1 = A_2 + B_2 = \cdots = A_N + B_N$, print `Yes`. Otherwise, print `No`.

Sample Input 1

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

Sample Output 1

```
Yes
```

Consider the following operations:

- Replace A_3 with 1.
- Replace B_2 with 1.
- Rearrange A to $(1, 3, 0, 2)$.

After these operations, $A = (1, 3, 0, 2)$ and $B = (3, 1, 4, 2)$: all elements of A and B are non-negative, and $A_1 + B_1 = A_2 + B_2 = A_3 + B_3 = A_4 + B_4 = 4$ is satisfied.

Sample Input 2

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

Sample Output 2

```
No
```

No matter how you perform the operations, it is impossible to satisfy $A_1 + B_1 = A_2 + B_2 = A_3 + B_3$.

Sample Input 3

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

Sample Output 3

```
No
```

C - Hamiltonian Pieces

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 600 points

Problem Statement

There is a board with 10^9 rows and 10^9 columns, and R red pieces and B blue pieces. Here, $R+B$ is not less than 2. The square at the r-th row from the top and the c-th column from the left is called square (r,c). A red piece can move vertically or horizontally by one square in one move, and a blue piece can move diagonally by one square in one move. More precisely, a red piece on square (r,c) can move to $(r+1,c),(r,c+1),(r-1,c),(r,c-1)$ in one move if the destination square exists, and a blue piece on square (r,c) can move to $(r+1,c+1),(r+1,c-1),(r-1,c+1),(r-1,c-1)$ in one move if the destination square exists.

We want to place all $(R+B)$ pieces on the board in any order, one by one, subject to the following conditions:

- At most one piece is placed on a single square.
- For each i $(1 \leq i \leq R+B-1)$, the i-th piece placed can move in one move to the square containing the $(i+1)$-th piece placed.
- The $(R+B)$-th piece placed can move in one move to the square containing the 1-st piece placed.

Determine whether there is a way to place the $(R+B)$ pieces satisfying these conditions. If it exists, show one example.

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

Constraints

- $1 \leq T \leq 10^5$
- $0 \leq R, B$
- $2 \leq R+B \leq 2 \times 10^5$
- The sum of $(R+B)$ over all test cases is at most 2×10^5.
- All input values are integers.

Input

The input is given from Standard Input in the following format:

$$T \\ \mathrm{case}_1 \\ \mathrm{case}_2 \\ \vdots \\ \mathrm{case}_T$$

Each case is given in the following format:

$$R \quad B$$

Output

Print the answer for each test case in order, separated by newlines.

If there is no way to place the pieces satisfying the conditions for a test case, print `No`.

Otherwise, print such a placement in the following format:

$$\texttt{Yes} \\ p_1 \quad r_1 \quad c_1 \\ \vdots \\ p_{R+B} \quad r_{R+B} \quad c_{R+B}$$

Here, p_i is `R` if the i-th piece placed is red, and `B` if it is blue. r_i and c_i are integers between 1 and 10^9 (inclusive), indicating that the i-th piece is placed on square (r_i, c_i).

Sample Input 1

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

Sample Output 1

```
Yes
B 2 3
R 3 2
B 2 2
B 3 3
R 2 4
No
Yes
R 1 1
R 1 2
R 2 2
R 2 1
```

For the 1st test case, if we extract the top-left 4×5 squares of the board, the placement of the pieces is as follows:

```
.....
.BBR.
.RB..
.....
```

Here, `R` indicates a red piece on that square, `B` indicates a blue piece on that square, and `.` indicates an empty square.

For the 2nd test case, there is no placement of the pieces that satisfies the conditions.

D - Swap and Erase

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 700 points

Problem Statement

There is a sequence $A = (A_1, \ldots, A_N)$. You can perform the following two types of operations any number of times in any order:

- Let K be the length of A just before the operation. Choose an integer i such that $1 \leq i \leq K-1$, and swap the i-th and $(i+1)$-th elements of A.
- Let K be the length of A just before the operation. Choose an integer i such that $1 \leq i \leq K$ and all the values from the 1-st through the i-th elements of A are equal, and delete all the elements from the 1-st through the i-th of A.

Find the minimum total number of operations required to make A an empty sequence.

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

Constraints

- $1 \leq T \leq 10^5$
- $2 \leq N \leq 2 \times 10^5$
- $1 \leq A_i \leq N$
- The sum of N over all test cases is at most 2×10^5.
- All input values are integers.

Input

The input is given from Standard Input in the following format:

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

Each case is given in the following format:

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

Output

Print the answer for each test case in order, separated by newlines.

Sample Input 1

```
3
5
1 1 2 1 2
4
4 2 1 3
11
1 2 1 2 1 2 1 2 1 2 1
```

Sample Output 1

```
3
4
8
```

For the 1st test case, A can be made empty by the following three operations:

- Swap the 3rd and 4th elements of A. Now, A is $(1, 1, 1, 2, 2)$.
- Delete the 1st through 3rd elements of A. Now, A is $(2, 2)$.
- Delete the 1st through 2nd elements of A. Now, A is an empty sequence.

For the 2nd test case, A can be made empty by deleting the 1st element four times. Also, it is impossible to make A empty in three or fewer operations.

E - Random Tree Distance

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 900 points

Problem Statement

There is an integer sequence $A = (A_2, A_3, \ldots, A_N)$. Also, for an integer sequence $P = (P_2, P_3, \ldots, P_N)$ where $1 \leq P_i \leq i-1$ for each i $(2 \leq i \leq N)$, define the weighted tree $T(P)$ with N vertices, rooted at vertex 1, as follows:

- A rooted tree where, for each i $(2 \leq i \leq N)$, the parent of i is P_i, and the weight of the edge between i and P_i is A_i.

You are given Q queries. Process them in order. The i-th query is as follows:

- You are given integers u_i and v_i, each between 1 and N. For each of the possible $(N-1)!$ sequences P, take the tree $T(P)$ and consider the distance between vertices u_i and v_i in this tree. Output the sum, modulo 998244353, of these distances over all $T(P)$. Here, the distance between two vertices u_i and v_i is the sum of the weights of the edges on the unique path (not visiting the same vertex more than once) that connects them.

Constraints

- $2 \leq N \leq 2 \times 10^5$
- $1 \leq Q \leq 2 \times 10^5$
- $1 \leq A_i \leq 10^9$
- $1 \leq u_i < v_i \leq N$
- All input values are integers.

Input

The input is given from Standard Input in the following format:

$$
\begin{array}{l}
N \quad Q \\
A_2 \quad A_3 \quad \dots \quad A_N \\
u_1 \quad v_1 \\
u_2 \quad v_2 \\
\vdots \\
u_Q \quad v_Q
\end{array}
$$

Output

Print Q lines. The i-th line should contain the answer to the i-th query.

Sample Input 1

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

Sample Output 1

```
2
3
```

- If $P = (1, 1)$, then in the tree $T(P)$, the distance between vertices 1 and 2 is 1, and the distance between vertices 1 and 3 is 1.
- If $P = (1, 2)$, then in the tree $T(P)$, the distance between vertices 1 and 2 is 1, and the distance between vertices 1 and 3 is 2.

Therefore, the total distance between vertices 1 and 2 over all $T(P)$ is 2, and the total distance between vertices 1 and 3 over all $T(P)$ is 3.

Sample Input 2

```
2 1
100
1 2
```

Sample Output 2

```
100
```

Sample Input 3

```
9 6
765689282 93267307 563699854 951829154 801512848 389123318 924504746 596035433
3 8
2 5
5 8
2 9
8 9
5 7
```

Sample Output 3

```
55973424
496202632
903509579
343265517
550981449
68482696
```

Remember to take the sum modulo 998244353.