S2-A. Duo Companion

Mr. Dengklek is the CEO of Bebek Dengklek Inc, one of the biggest companies in Singanesia. There are N employees at his company, each of them has a unique ID numbered between 1 and N inclusive. As the company has a very good culture, the employees tend to get along with each other really well.

After observing how some pairs of workers are able to achieve better performance upon being paired up together, Mr. Dengklek plans to revise the structural organization of his company to run things more efficiently. He asks his secretary to make a report of the best pairings that can be made among the employees.

After some time, his secretary comes back with a list of M possible pairings, where the i-th item ($1 \leq i \leq M$) basically says that the employee with ID U_i works very well with the employee with ID V_i ($U_i < V_i$). Interestingly, it comes with a special property: if an employee with ID x is paired with another one with ID y, their difference is either A or B ($0 < A < B < N$). More formally, it is guaranteed that $V_i - U_i = A$ or $V_i - U_i = B$. Also, no pair of workers is listed more than once in the list.

Mr. Dengklek can make matchings using the pairings he got. A matching is defined as a set of pairings such that no employees are paired twice. Two matchings are different if there exists a pairing that is included in one but not the other.

Mr. Dengklek wants to determine the **number of different matchings** that can be made, modulo 998244353, given the M possible pairings. Note that the matchings **do not** need to have maximum number of pairings. He is looking for someone that can solve the problem and offers a huge sum of money as a prize. You definitely don't want to miss this once in a lifetime opportunity.

Implementation Details

You need to implement the following function:

```
int count_matching(int N, int M, int A, int B, vector U, vector V)
```

- N: the number of employees.
- M: the number of possible pairings.
- A, B: integers denoting the only possible differences of each pair of employee ID.
- U, V: arrays of size M each that describe the possible pairing of the employees.
- This function must return the number of different pairings modulo 998244353.
- This function is called exactly once by the grader.

Examples

Example 1

Consider the following call:

```
count_matching(4, 3, 1, 2, [1, 1, 3], [2, 3, 4])
```

This means that Mr. Dengklek has 4 employees and 3 possible pairings, each pair has a difference in value of either 1 or 2. The three possible pairings are $(1, 2)$, $(1, 3)$ and $(3, 4)$. This function should return 5, as there are 5 possible different pairings.

Example 2

Consider the following call:

```
count_matching(10, 14, 2, 4,
    [5, 7, 2, 6, 1, 3, 4, 1, 4, 8, 3, 5, 2, 6],
    [7, 9, 6, 8, 5, 7, 8, 3, 6, 10, 5, 9, 4, 10])
```

This function should return 225.

Constraints

- $3 \leq N \leq 200$
- $1 \leq M \leq 400$
- $1 \leq A < B \leq N - 1$
- $1 \leq U_i < V_i \leq N$
- $V_i - U_i = A$ or $V_i - U_i = B$
- For all $1 \leq i < j \leq M$, $U_i \neq U_j$ or $V_i \neq V_j$

Subtasks

Subtask 1 (7 points)

- $M \leq 20$

Subtask 2 (12 points)

- $A = 1$
- $B = 2$

Subtask 3 (23 points)

- $B \leq 20$

Subtask 4 (37 points)

- $gcd(A, B) = 1$

Subtask 5 (21 points)

- No additional constraints

Example Grader

The provided grader reads the input in the following format:

- The first line consists of four integers N, M, A and B
- The next M lines consist of U_i and V_i, where each line represents a possible pair.

The provided grader prints output in the following format:

- The first line consists of an integer returned by the `count_matching` function.