You are given a list of preferences for n friends, where n is always **even**.

For each person i, preferences[i] contains a list of friends **sorted** in the **order of preference**. In other words, a friend earlier in the list is more preferred than a friend later in the list. Friends in each list are denoted by integers from 0 to n-1.

All the friends are divided into pairs. The pairings are given in a list pairs, where pairs[i] = [xi, yi] denotes xi is paired with yi and yi is paired with xi.

However, this pairing may cause some of the friends to be unhappy. A friend x is unhappy if x is paired with y and there exists a friend u who is paired with v but:

* x prefers u over y, and
* u prefers x over v.

Return *the number of unhappy friends*.

**Example 1:**

Input: n = 4, preferences = [[1, 2, 3], [3, 2, 0], [3, 1, 0], [1, 2, 0]], pairs = [[0, 1], [2, 3]]  
Output: 2  
Explanation:  
Friend 1 is unhappy because:  
- 1 is paired with 0 but prefers 3 over 0, and  
- 3 prefers 1 over 2.  
Friend 3 is unhappy because:  
- 3 is paired with 2 but prefers 1 over 2, and  
- 1 prefers 3 over 0.  
Friends 0 and 2 are happy.

**Example 2:**

Input: n = 2, preferences = [[1], [0]], pairs = [[1, 0]]  
Output: 0  
Explanation: Both friends 0 and 1 are happy.

**Example 3:**

Input: n = 4, preferences = [[1, 3, 2], [2, 3, 0], [1, 3, 0], [0, 2, 1]], pairs = [[1, 3], [0, 2]]  
Output: 4

**Constraints:**

* 2 <= n <= 500
* n is even.
* preferences.length == n
* preferences[i].length == n - 1
* 0 <= preferences[i][j] <= n - 1
* preferences[i] does not contain i.
* All values in preferences[i] are unique.
* pairs.length == n/2
* pairs[i].length == 2
* xi != yi
* 0 <= xi, yi <= n - 1
* Each person is contained in **exactly one** pair.