## Perfect Matching

Credit Suisse organizes a private banking career information session for its potential future private bankers and currently employed private bankers. There are n private bankers and m participants.

Assume for each participant, they want to meet a number of private bankers and similarly, for each private banker they want to recruit a number of participants. However, only one-on-one meetings are possible. So for each session, one participant can only meet one banker.

If Credit Suisse has a list of preferences from participants and private bankers, how many sessions are needed in order to fulfil everyone's preferences?

# Constraints

- Every banker and participant must have at least one preference.

# Input format

The first line relates to the private bankers, and the second line relates to the participants.

The first integer in each line is the number of bankers/participants.

The subsequent integer input is the preference of bankers/participants, the preference of each person is separated by `,`.

For example:

```
2 1&2,2
2 1,2
```

The first line of input means that there are two private bankers. The preference of banker 1 is to meet participants 1 & 2, and the preference of banker 2 is to meet participant 2 only.

The second line of input means that there are two participants. The preference of participant 1 is to meet banker 1 only, and the preference of participant 2 is to meet banker 2 only.

# Output format

An integer that is the minimum number of sessions required to fulfil everyone's preferences.

# Examples

## Example 1

### Input

```
2 1,2&3
3 1,2,2
```

### Output

```
2
```

i.e. In the first session, banker 1 will meet with participant 1 (fulfilling banker 1's preference and participant 1's preference) and banker 2 will meet with participant 2 (fulfilling banker 2's first preference and participant 2's preference). Another session is needed to fulfil banker 2's second preference and participant 3's only preference. In the second session, banker 2 will meet with participant 3 (fulfilling banker 2's second preference and banker 3's preference) and banker 1 will meet with participant 2. After two sessions, everyone's preferences have been fulfilled.

## Example 2

### Input

```
3 1,1,1
3 3,1,1
```

### Output

```
3
```

i.e. There are many ways to arrive at this solution. This is just one of them. In the first session, banker 1 will meet with participant 1 (fulfilling banker 1's preference), banker 2 will meet with participant 2 and banker 3 will meet with participant 3. In the second session, banker 2 will meet with participant 1 (fulfilling banker 2's preference), banker 3 will meet with participant 2, and banker 1 will meet with participant 3 (fulfilling participant 3's preference). This leaves banker 3, participant 1, and participant 2's preferences. So in the third session, banker 3 will meet with participant 1 (fulfilling banker 3's preference and participant 1's preference), banker 1 will meet with participant 2 (fulfilling participant 2's preference), and banker 2 will meet with participant 3. After three sessions, everyone's preferences have been fulfilled.


```cpp
#include <bits/stdc++.h>
#define forr(i, p, n) for (int i = p; i < (int)n; i++)
using namespace std;

int calculateMinimumSession(int nb, int np, vector<vector<int>> bp, vector<vector<int>> pp)
{
    vector<set<int>> b2p(nb + 1);
    vector<set<int>> p2b(np + 1);
    b2p.clear();
    p2b.clear();
```

```
11      forr(i, 0, bp.size())
12      {
13          for (int j = 0; j < (int)bp[i].size(); j++)
14          {
15              b2p[i + 1].insert(bp[i][j]);
16              p2b[bp[i][j]].insert(i + 1);
17          }
18      }
19      // int base1 = (int)b2p.size() * (int)p2b.size();
20      forr(i, 0, pp.size())
21      {
22          for (int j = 0; j < (int)pp[i].size(); j++)
23          {
24              p2b[i + 1].insert(pp[i][j]);
```

## Your submission History for Question 5

| | Timestamp | Commit ID | Language | # Tests Passed | # Tests Failed | # Tests Timed out | Build Status |
|---|---|---|---|---|---|---|---|
| + | 10/20/2020, 1:16:32 AM | 233ef64 | CPLUSPLUS | 90 | 0 | 0 | Perfect |
| + | 10/20/2020, 12:50:33 AM | 70c3a04 | CPLUSPLUS | 56 | 34 | 0 | Failed |
| + | 10/20/2020, 12:13:28 AM | cce64df | CPLUSPLUS | 22 | 68 | 0 | Failed |
| + | 10/20/2020, 12:11:01 AM | 9443272 | CPLUSPLUS | 22 | 68 | 0 | Failed |
| + | 10/20/2020, 12:08:11 AM | 0a9c3e6 | CPLUSPLUS | 0 | 90 | 0 | Failed |