**Matching Tokens**

We define the following:

* There are *friends\_nodes* friends numbered from *1* to *friends\_nodes*.
* There are *friends\_edges* pairs of friends, where each *(xi, yi)* pair of friends is connected by a shared integer *token* described by *friends\_weighti*.
* Any two friends, *xi* and *yi*, can be connected by zero or more tokens because if friends *xi* and *yi* share token *ti* and friends *yi* and *zi* also share token *ti*, then *xi* and *zi* are also said to share token *ti*.

Find the maximal product of *xi* and *yi* for any directly or indirectly connected *(xi, yi)* pair of friends such that *xi* and *yi* share the maximal number of tokens with each other.

Complete the *maxTokens* function in the editor. It has four parameters:

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| friends\_nodes | integer | The number of friends. |
| friends\_from | integer array | Each *friends\_from[i]* (where *0 ≤ i < friends\_edges*) denotes the first friend in pair *(friends\_from[i], friends\_to[i])*. |
| friends\_to | integer array | Each *friends\_to[i]* (where *0 ≤ i < friends\_edges*) denotes the second friend in pair *(friends\_from[i], friends\_to[i])*. |
| friends\_weight | integer array | Each *friends\_weight[i]* (where *0 ≤ i < friends\_edges) denotes the ID number of a token shared by both friends\_from[i] and friends\_to[i].* |
| **Note:** *friends\_edges* is the number of pairs of friends that directly share a token. | | |

The function must return an integer denoting the maximal product of *xi* and *yi* such that *xi* and *yi* are a pair of friends that share the maximal number of tokens with each other.

**Input Format**

The first line contains two space-separated integers describing the respective values of *friends\_nodes* and *friends\_edges*.

Each line *i* of the *friends\_edges* subsequent lines (where *0 ≤ i < friends\_edges*) contains three space-separated integers describing the respective values of *friends\_fromi*, *friends\_toi*, and *friends\_weighti*.

**Constraints**

* *2 ≤ friends\_nodes ≤ 100*
* *1 ≤ friends\_edges ≤ min(200, (friends\_nodes × (friends\_nodes − 1)) / 2)*
* *1 ≤ friends\_weighti ≤ 100*
* *1 ≤ friends\_fromi, friends\_toi ≤ friends\_nodes*
* *1≤ friends\_weighti ≤ friends\_edges*
* *friends\_fromi ≠ friends\_toi*
* Each pair of friends can be connected by zero or more types of tokens.

**Output Format**

Return an integer denoting the maximal product of *xi* and *yi* such that *xi* and *yi* are a pair of friends that share the maximal number of tokens with each other.

**Sample Input 0**

4 5

1 2 1

1 2 2

2 3 1

2 3 3

2 4 3

**Sample Output 0**

6

**Explanation 0**

|  |  |
| --- | --- |
| Each pair of *n = 4* friends is connected by the following tokens:   * Pair *(1, 2)* shares *2* tokens (i.e., tokens *1* and *2*) * Pair *(1, 3)* shares *1* token (i.e., token *1*) * Pair *(1, 4)* shares *0* tokens * Pair *(2, 3)* shares *2* tokens (i.e., tokens *1* and *3*) * Pair *(2, 4)* shares *1* token (i.e., token *3*) * Pair *(3, 4)* shares *1* token (i.e., token *3*)   The pairs connected by the maximal number of tokens are *(1, 2)* and *(2, 3)*. Their respective products are *1 × 2 = 2* and *2 × 3 = 6*. We then return the largest of these values as our answer, which is *6*. |  |