ESCAPE

Problem

It is the year 3018, and Vim is as popular an editor as it has ever been. Recently, you've actually been trying to teach Granny how to use the latest release of Vim, version 124.0. To help her escape even the most esoteric of Vim states, you want to teach her a sequence of keypresses (an "escape sequence"), so that regardless of the initial state of her editor, the keypresses will bring her editor to the same familiar ending state.

Given a description of Vim 124.0, which has nnn states labeled 111 through nnn, your task is to determine whether such an escape sequence exists. Because Granny's keyboard has kkk keys, the state transitions are characterized by kkk functions f1,…,fkf_1,\ldots,f_kf1,…,fk, where fp(s)f_p(s)fp(s) is the state obtained by pressing the ppp-th key while in state sss. An escape sequence is then a sequence of keypresses p1,…,ptp_1,\ldots,p_tp1,…,pt such that (fpt∘fpt−1∘⋯∘fp1)(s) (f_{p_t}\circ f_{p_{t-1}}\circ\cdots\circ f_{p_1})(s) (fpt∘fpt−1∘⋯∘fp1)(s) is the same for all s∈{1,…,n}s\in \{1,\ldots,n\}s∈{1,…,n}. (∘\circ∘ denotes function composition.) Equivalently, for all s,s′∈{1,…,n}s,s'\in\{1,\ldots,n\}s,s′∈{1,…,n}, fpt(fpt−1(⋯(fp1(s))⋯))=fpt(fpt−1(⋯(fp1(s′))⋯)). f_{p^t}(f_{p^{t-1}}(\cdots(f_{p^1}(s))\cdots)) = f_{p^t}(f_{p^{t-1}}(\cdots(f_{p^1}(s'))\cdots)). fpt(fpt−1(⋯(fp1(s))⋯))=fpt(fpt−1(⋯(fp1(s′))⋯)). (You may find this latter characterization helpful for fully solving this problem!) You should output `YES` if an escape sequence exists for the given description of Vim 124.0 and `NO` otherwise.

Hint: How can you construct escape sequences by merging together pairs of states? Try to find an "if and only if" condition for an escape sequence to exist.

Constraints

There is one sample input, worth 0 points. (See "Sample Input" below.) This test serves to verify that the server is assessing your code correctly, in the event that all of the other test cases are reported as Incorrect Output.

For all other test cases, worth a total of 100 points, it is guaranteed that n≤250 n\le 250 n≤250 and k≤50 k\le 50 k≤50. These test cases are further divided into five batches, with the following additional constraints:

- in the first batch of test cases, worth 15 points, it is guaranteed that n≤10 n\le 10 n≤10 and k=1 k = 1 k=1;
- in the second batch, worth 20 points, it is guaranteed that n≤5 n\le 5 n≤5 and k≤10 k\le 10 k≤10;
- in the third batch, worth 25 points, it is guaranteed that n≤10 n\le 10 n≤10 and k≤10 k\le 10 k≤10;
- in the fourth batch, worth 25 points, it is guaranteed that n≤30 n\le 30 n≤30 and k≤30 k\le 30 k≤30;
- the fifth batch, with no additional constraints, is worth the remaining 15 points.

For each of these batches, there will be one input file containing TTT test cases (see below). It is guaranteed that 1≤T≤101\le T\le 101≤T≤10.

Time Limit

The time limit is 100ms for the first input file (worth a total of 15 points), 500ms for the second and third input files (worth a total of 20 and 25 points, respectively), 800ms for the third input file (worth a total of 25 points), and 3000ms for the fourth input file (worth a total of 15 points).

Input Format

To reduce the effectiveness of randomly outputting `YES` or `NO`, we are placing multiple test cases in the same input file. Each input file begins with an integer TTT, the number of tests in the file. The remainder of the file contains the TTT test cases, separated by newlines. Each test case consists of k+1k+1k+1 lines as follows:

- line 1 consists of two space-separated integers, nnn and kkk (the number of states and the number of keys, respectively);
- lines 2 through k+1k+1k+1 each consist of nnn space-separated integers giving the values of fp(1),…,fp(n) f_p(1),\ldots,f_p(n) fp(1),…,fp(n) for p=1,2,…,kp = 1,2,\ldots,kp=1,2,…,k.

Output Format

For each test case, output `YES` if an escape sequence exists and `NO` otherwise, followed by a newline.

Sample Cases

Sample Input

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

Sample Output

```
NO
YES
```

Details

In the first test case, it is clear that pressing the only available key cycles the states. Thus, after a sequence of key presses, any two distinct states will still be distinct, making it impossible for an escape sequence to exist.

In the second test case, pressing the sequence 1-2-1 brings all starting states to state 1, so there exists an escape sequence for this specification of Vim 124.0.

View submissions

Test cases

Input	Output	Points	Timeout
2 3 1 2 3 1 3 2 1 1 3	NO YES	0	100 ms

Input	Output	Points	Timeout
`1 3 2`			
Hidden	*Hidden*	15	100 ms
Hidden	*Hidden*	15	3000 ms
Hidden	*Hidden*	20	500 ms
Hidden	*Hidden*	25	500 ms
Hidden	*Hidden*	25	800 ms

Download

Inspired by the "Ultra Cool Programming Contest Control Centre" by Sonny Chan.
Modified for CS 124 by Neal Wu, with design help from Martin Camacho.
Further refined by Nikhil Benesch.