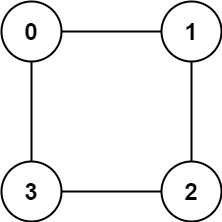
Given an undirected graph, return true if and only if it is bipartite.

Recall that a graph is *bipartite* if we can split its set of nodes into two independent subsets A and B, such that every edge in the graph has one node in A and another node in B.

The graph is given in the following form: graph[i] is a list of indexes j for which the edge between nodes i and j exists.  Each node is an integer between 0 and graph.length - 1.  There are no self edges or parallel edges: graph[i] does not contain i, and it doesn't contain any element twice.

**Example 1:**

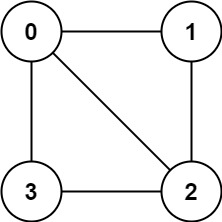


**Input:** graph = [[1,3],[0,2],[1,3],[0,2]]

**Output:** true

**Explanation:** We can divide the vertices into two groups: {0, 2} and {1, 3}.

**Example 2:**



**Input:** graph = [[1,2,3],[0,2],[0,1,3],[0,2]]

**Output:** false

**Explanation:** We cannot find a way to divide the set of nodes into two independent subsets.

**Constraints:**

* 1 <= graph.length <= 100
* 0 <= graphp[i].length < 100
* 0 <= graph[i][j] <= graph.length - 1
* graph[i][j] != i
* All the values of graph[i] are **unique**.
* The graph is **guaranteed** to be **undirected**.