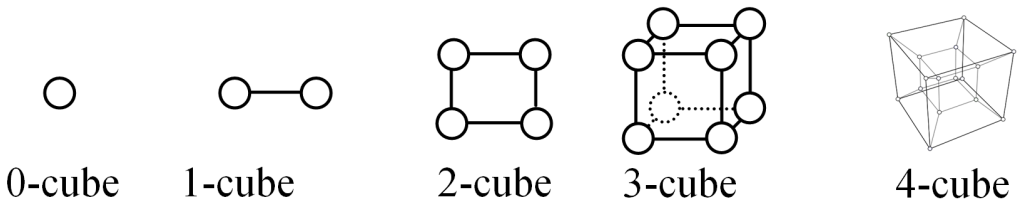
Problem F
Hypercube

Source file name: `hypercube.c`, `hypercube.cpp` *or* `hypercube.java`

In geometry, a hypercube is an $n-$dimensional analogue of a square ($n = 2$) and a cube ($n = 3$). It consists in groups of opposite parallel line segments aligned in each of the space's dimensions, at right angles to each other and of the same length. An $n-$dimensional hypercube is also called an $n-$cube.



In parallel computing, the vertexes are processors, and the line segments (edges) represent connections. The $n-$cube architecture has the following properties:

- Each node has n connections with different processors.
- Each processor has a unique identifier, between 0 and $2^n - 1$.
- Two processors are directly connected if and only if their identifiers differ in just one bit. For instance, in a $3-$cube, processors 3 (011 in binary) and 7 (111 in binary) are directly connected.
- The number of processors is 2^n

The new company WEFAIL is designing hypercubes, but they are always contracting new people, whose do not know all the hypercube properties, and sometimes they fail; thus these properties are not satisfied in all cases.

Given an arbitrary graph, your task is to write a program that determines whether the graph is a hypercube or not.

Input

The input consists in several problem instances. Each instance contains one graph, which starts with a line with two positive integers: K and M, representing the number of vertexes ($0 < K \leq 1024$) and the number of edges respectively. It follows ($0 \leq M \leq 5130$) lines, representing the edges. Each edge is given by two 32 bits integers, representing the processors connected by the edge.

The end of input is indicated by a test case with $K = 0$.

The input must be read from standard input.

Output

For each problem instance, the output is a single line, with the word "YES" if the corresponding graph is a hypercube, and "NO" otherwise (quotes for clarity).

The output must be written to standard output.

Sample input	**Output for the sample input**
4 4 0 1 1 3 2 0 3 2 2 1 1 4 3 2 0 1 1 2 0 0	YES NO NO