Problem A: Building Roads

Advanced Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 256 Mb

Problem description

After finishing your studies, you moved to the little known island kingdom of *Tsu-Nami*, whose infrastructure gets obliterated by a flood every couple of years. Every time this happens, the king orders his subjects to rebuild the network of roads between the island's villages by themselves and report to him whenever they finish a road. Not to waste too much of their time on this, he wants them to stop once any village can be reached from any other village via the new roads. This hasn't worked very well in the past though, since the mathematicians he ordered to oversee the process were having a hard time keeping track of the network's connectivity.

As he recently learned that you, a computer scientist, had moved to *Tsu-Nami*, he contacted you and ordered you to write a program that reads the reports of newly built roads and instantly informs the king when the desired connectivity is reached.

Input

The first line contains N ($2 \leq N \leq 10^5$), the number of villages to be connected, and M ($1 \leq M \leq 10 \cdot N$), the number of roads that the king's subjects are *willing* to build following his initial order. Then there are M lines containing descriptions of the new roads in the order in which they are (or *would be*) finished.

Both cities and roads are numerated $1, ..., N$ and $1, ..., M$ respectively. The i-th road description consists of two integers a_i and b_i ($1 \leq a_i, b_i \leq N$), describing that road i connects villages a_i and b_i (bidirectionally). Note that due to lack of organization the subjects may occasionally build roads between a city and itself or between cities that were already connected by another road.

Output

If road i makes the last connections that were missing, output i. If there are still cities not connected by the M roads you are given, output "Build some more!".

Sample input and output

Input	**Output**
4 6 1 2 2 3 3 1 4 2 3 4 4 1	4
6 6 1 3 6 4 3 5 4 2 5 1 2 6	Build some more!