# Problem 3 – Graph Cycles

You’ll be given a **directed graph**; your task is to **find all cycles of length 3** and **print them in lexicographical order**.

The graph’s vertices will be **natural numbers**. Cycles are considered different if the direction of movement is different, even if they contain the same 3 vertices. For example, the cycle 0 -> 1 -> 2 is the same as 1 -> 2 -> 0 and 2 -> 0 -> 1, but is different from 2 -> 1 -> 0.

There may be multiple edges connecting the same two vertices in the same direction, take into consideration only one of them – e.g. if there are two edges connecting 0 -> 1, there is only one cycle 0 -> 1 -> 2 because the two edges between 0 and 1 are considered identical. Check out the examples to understand the task better.

### Input

* The input data should be read from the console.
* On the first input line you’ll be given the **number of vertices N**.
* On the next N lines, you’ll be given the edges in format **{source} -> {child1} {child2} … {childN}**.
* The input data will always be valid and in the format described. There is no need to check it explicitly.

### Output

* The output should be printed on the console.
* On each line, print a cycle of length 3. Cycles should be printed in format **{{vertex1} -> {vertex2} -> {vertex3}}** in **lexicographical order**.

### Constraints

* The graph will contain between 1 and 50 vertices.
* Vertices will be natural numbers in the range [0 … N].
* Allowed working time for your program: 0.1 seconds. Allowed memory: 16 MB.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 3  1 -> 2 2  0 -> 1  2 -> 0 | {0 -> 1 -> 2} | One cycle (ignore the second edge connecting 1 -> 2): |
| 4  0 -> 2 3 1  1 -> 3 0  2 -> 0 3  3 -> 2 0 1 | {0 -> 1 -> 3}  {0 -> 2 -> 3}  {0 -> 3 -> 1}  {0 -> 3 -> 2} | The cycle 0 -> 1 -> 3 is different from 0 -> 3 -> 1 (different direction). Same applies for 0 -> 2 -> 3 and 0 -> 3 -> 2: |
| 14  12 -> 7  7 -> 12  5 -> 13  13 -> 9 2  9 -> 1 0  2 -> 0 11  0 -> 13 1 11  1 -> 6  11 -> 8  6 -> 0 11 10  8 -> 6  10 -> 10 4  4 -> 6 3  3 -> 4 | {0 -> 1 -> 6}  {0 -> 13 -> 2}  {0 -> 13 -> 9}  {4 -> 6 -> 10}  {6 -> 11 -> 8} |  |
| 4  0 -> 1  1 -> 2  2 -> 3 2 1  3 -> | No cycles found |  |