In a directed graph, we start at some node and every turn, walk along a directed edge of the graph.  If we reach a node that is terminal (that is, it has no outgoing directed edges), we stop.

Now, say our starting node is *eventually safe*if and only if we must eventually walk to a terminal node.  More specifically, there exists a natural number K so that for any choice of where to walk, we must have stopped at a terminal node in less than K steps.

Which nodes are eventually safe?  Return them as an array in sorted order.

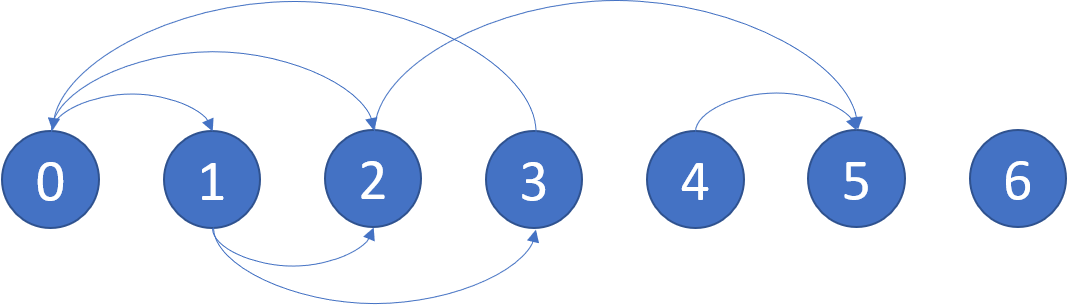
The directed graph has N nodes with labels 0, 1, ..., N-1, where N is the length of graph.  The graph is given in the following form: graph[i] is a list of labels j such that (i, j) is a directed edge of the graph.

**Example:**

**Input:** graph = [[1,2],[2,3],[5],[0],[5],[],[]]

**Output:** [2,4,5,6]

Here is a diagram of the above graph.



**Note:**

* graph will have length at most 10000.
* The number of edges in the graph will not exceed 32000.
* Each graph[i] will be a sorted list of different integers, chosen within the range [0, graph.length - 1].