Topological Sort (Kahn's Algorithm) Pack — Cheat Sheet

Topological Sort (Kahn's Algorithm) Pack — Cheat Sheet
==========

Files:
1) can_finish.py — cycle detection for course schedule.
2) topo_order.py — produce a valid topological ordering.
3) min_semesters.py — layered BFS to count semesters/phases.
4) kahn_template.py — reusable scaffold for toposort problems.

Kahn's Algorithm Core:
- Build indegree[] and adjacency list for directed edges u->v.
- Queue all nodes with indegree 0.
- Pop, record/process node, decrement neighbors' indegree, push new zeros.
- If all nodes are popped, graph is acyclic; otherwise, there is a cycle.

Patterns:
- To detect cycle: check if popped count == n.
- To build order: append popped nodes into a list.
- To count layers (semesters): process queue level-by-level.

Complexity:
- O(n + m) time, O(n + m) space.

Tips:
- Edges direction matters: (u, v) means u precedes v.
- If multiple valid orders exist, queue ordering determines which one.
- For critical path length (longest semesters when each course takes 1),
  use DP on topo order instead of BFS layering.