

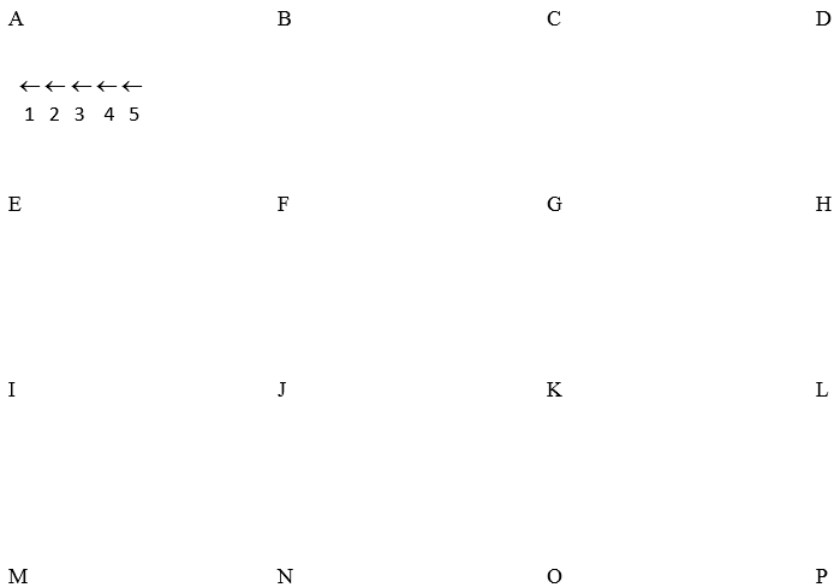
1. **Topological sort:** In a dag, the vertices can be linearly ordered so that every edge's starting vertex is listed before its ending vertex.
  
2. Two Topological sort algorithms (for a **dag**):
  - a. Based on DFS.
    - i. Do a DFS, keeping track of the order in which the nodes are popped off the stack.
    - ii. Reverse the order.
  
  - b. Source removal algorithm.

Repeatedly identify and remove a source node. If there are no cycles, there will always be a source.

Permutation generation: **We want to generate all permutations of the numbers 1, 2, ..., n.**

3. Bottom-up algorithm. Alternate the insertion orders.

4. Johnson-Trotter. Every element has an additional piece of info, its direction (right or left).
  - a. An element is *mobile* if the element it “points to” is smaller than itself.
  - b. Largest mobile element is swapped with the element it points to.
  - c. Then reverse the direction of all larger elements.



5. Which permutation follows each of these in lexicographic order?

183647520

471638520

6. Write an algorithm for generating the next permutation, with only  $N$  and the current permutation as input.

7. If the lexicographic permutations of the numbers  $[0, 1, 2, 3, 4]$  are numbered starting with 0, what is the number of the permutation 14023? How do you get this?

8. Write an algorithm which, given a permutation of the numbers  $0..N-1$ , calculates its (zero-based) position in the lexicographic ordering of all of the permutations of  $0..n-1$ .

9. In the lexicographic ordering of permutations of  $[0, 1, 2, 3, 4, 5]$ , which permutation is number 568? How do you get this?