*Algorithms and programming*
École normale supérieure - Département d'informatique
TD 9: Depth-first search

---

1. Explain how a vertex $u$ can end up in a depth-first tree containing only $u$, even though $u$ has both incoming and outgoing edges in $G$.
2. Show that a depth-first search of an indirected graph can be used to identify the connected components of $G$, and that the depth-first forest contains as many trees as $G$ has connected components. More precisely, show how to modify depth-first search so that each vertex $v$ is assigned an integer label $cc[v] \in [1, k]$, where $k$ is the number of connected components of $G$, such that $cc[u] = cc[v]$ if and only if $u$ and $v$ are in the same component.
3. There is a new alien language which uses the Latin alphabet. However, the order among letters is unknown to you. You receive a list of non-empty words from the dictionary, where words are sorted lexicographically by the rules of this new language. Derive the order of letters in this language.
4. Prove or disprove : If a directed graph $G$ contains cycles, then TOPOLOGICAL-SORT produces a vertex ordering that minimizes the number of "bad" edges that are inconsistent with the ordering produced.
5. Another way to perform topological sorting on a directed acyclic graph $G = (V, E)$ is to repeatedly find a vertex of in-degree 0, output it, and remove it and all of its outgoing edges from the graph. Explain how to implement this idea so that it runs in time $O(|V| + |E|)$. What happens to this algorithm if $G$ has cycles ?
6. How can the number of strongly connected components of a graph change if a new edge is added ?
7. An Euler tour of a connected, directed graph $G = (V, E)$ is a cycle that traverses each edge of G exactly once, although it may visit a vertex more than once.
   (a) Show that $G$ has an Euler tour if and only if in-degree($v$) = out-degree($v$) for each vertex $v \in V$.
   (b) Describe an $O(|E|)$-time algorithm to find an Euler tour of $G$ if one exists.
8. Let $G = (V, E)$ be a directed graph in which each vertex $u \in V$ is labeled with a unique integer $L(u) \in \{1, 2, \ldots, |V|\}$. For each vertex $u \in V$, let $R(u)$ be the set of vertices reachable from $u$. Define $\min(u)$ to be the vertex in $R(u)$ whose label is minimum. Give an $O(|V| + |E|)$-time algorithm that computes $\min(u)$ for all $u \in V$.