There are 100 points in total.

1. **(15 points)** Show how the algorithm **Recursive DFS** (along with **To-Start-DFS**) in the Lecture Note 4 works on the graph figure shown below. Assume for the graph, the adjacency list of every vertex is ordered alphabetically. Assume that the for loop of lines 4-5 of algorithm **To-Start-DFS** considers the vertices in the alphabetical order. Show the resulted depth-first-search tree (or forest).

2. **(25 points)** Consider the following **Constrained Reachability** problem: the input consists of a directed graph $G(V,E)$, two vertices $s,t \in V$, and a subset $U \subseteq V$, where $U \cap \{s,t\} = \emptyset$. The output is “yes” if and only if there is a path from $s$ to $t$ in the graph **without** using any vertex in $U$. Apparently, the problem is the commonly known **Reachability** problem when $U = \emptyset$.

   (1) Design a recursive DFS type of algorithm to **Constrained Reachability** solve problem. You may assume that the input graph is already stored in adjacency lists and $v.\mu$ indicates if vertex $v$ is in the set $U$ or not. The information about $\mu$ is also given along with the graph.
(2) Argue that your algorithm still runs in linear time (i.e., \(O(|E| + |V|)\)-time) though the set \(U\) may contain a large number of vertices.

3. (20 points) It is known that depth-first-search strategy can be used to determine if a given undirected graph contains a cycle. Argue that the algorithm runs in time \(O(|V|)\), independent of \(|E|\) before it answers the question correctly. Note that you need to prove that time \(O(|V|)\) is enough for the algorithm to answer both “yes” and “no”.

4. (15 points) Show how the procedure Strongly Connected Components (page page 617) works on the graph given for question 1. Specifically show the finishing times computed in line 1 and the forest produced in line 3. Assume the for loop of lines 5-7 of DFS (page 604) considers vertices in alphabetical order and that the vertices in each adjacency list are in alphabetical order.

5. (25 points) Prove that the Topological Sorting problem can be solved with the reversed order of finished time produced by the DFS algorithm. Note that you will need to argue that, for any two vertices \(u\) and \(v\) in the graph, \(u.f > v.f\) if and only if there is a directed path from \(u\) to \(v\).

The answers must be word-processed or typed. You may substitute formulae and figures with hand-writings. Your submitted algorithms should be in the pseudo-code, not in any specific programming language. Answers deviating from these requirements will be returned without grading.

The answers must be the student’s own work. Idea sharing and referencing to others’ work are not allowed. Plagiarism and other forms of academic dishonesty will be handled within the guidelines of the Student Handbook and reported to the University.