EDAF05 example questions 2021

There will be no programming questions. There may be a problem solving question but nothing complex or ”tricky” if you understand the course contents.

1. Explain what $O(n)$, $\Omega(n)$, and $\Theta(n)$ mean.
2. Suppose you have invented a greedy algorithm that finds an optimal solution to a problem. Explain two approaches to prove its output really is optimal.
3. Explain what is meant by a divide-and-conquer algorithm (söndra-och-härska).
4. Explain what is meant by dynamic programming (dynamisk programmering).
5. Explain what the Master theorem is about.
6. Explain how hollow heaps work. Focus on the simplest version with multiple root nodes.
7. What can make the naïve version of union-find slow?
8. Explain how union-find can be made faster than as in the naïve version.
9. What does it mean that a directed graph is strongly connected (starkt sammankopplad), and how can you use BFS to determine if a graph is strongly connected?
10. Explain how Tarjan’s algorithm can find the strongly connected components in a directed graph.
11. What is a bipartite graph (bipartit graf), and how can you determine if a graph is bipartite?
12. Explain how Dijkstra’s algorithm works and why it is correct.
13. Explain what can happen if there are negative edge weights (negativa kanter).
14. Explain how the Bellman-Ford algorithm works and why it is correct.
15. Explain what a minimum spanning tree (minimalt uppspännande träd) is and how it can be found using Prim’s and Kruskal’s algorithms.
16. What is a safe edge (säker kant) for miminum spanning trees?
17. What is network flow (nätverksflöde) about? Give an example of when it can be used.
18. Explain the Ford-Fulkerson algorithm and why it is correct. What is its time complexity, and why?
19. Explain the Goldberg-Tarjan (preflow-push) algorithm and why it is correct.
20. Explain why the Gale-Shapley algorithm finds a stable matching (stabil matchning)?
21. Explain the time complexity of Gale-Shapley.
22. What is sequence alignment and how can it be done?
23. What does it mean that a problem is NP-complete (NP-fullständigt) ?
24. If you want to prove that a new problem is NP-complete, how would you do?
25. Explain how the first NP-complete problem was shown to be NP-complete.
26. Explain how it can be shown that Hamiltonian cycle (Hamiltonsk cykel) is NP-complete.
27. Explain how it can be shown that the Traveling salesman problem (Handelsresandeproblemet) is NP-complete.
28. Explain how it can be shown that graph coloring (graffärgning) is NP-complete.
29. Explain what the simplex algorithm can do (but not why it works).
30. Explain what the branch-and-bound paradigm (förgrena-och-begränsa) is and can used exploited in integer linear programming (heltalsprogrammering).