Tentamensdatum: 2020-05-26, 09:00 - 13:00
Tillåtna hjälpmedel: Engelsk-svensk ordbok
Totalt antal uppgifter: 11 st, 100p
Jourhavande lärare: 0920 / 49 20 44

1. Sort numbers 954, 323, 667, 254, 234 using *radixsort step by step* (starting with the least significant digit). **(5p)**

2. It is known that *radixsort* sorts n d-digits numbers (in which each digit is an integer between 0 and 9) in time $\Theta(d \cdot n)$ in the worst case. What is the best-case complexity of *radixsort* on n d-digits numbers ($d > 1$)? Why? **(5p)**

3. Construct hash tables of size 11 by inserting $20, 7, 9, 18, 21, 0$ *in turn* into an initially empty hash table

   (a) with collisions resolved by (separate) chaining. What is your hash function? **(5p)**

   (b) using double hashing. What is your hash function? **(5p)**

4. Draw the binary search tree obtained by inserting $20, 7, 9, 18, 21, 0$ *in turn* into an initially empty binary search tree. **(5p)**

5. Draw the binary tree that represents the min-heap obtained by inserting $20, 7, 9, 18, 21, 0$ *in turn* into an initially empty min-heap. **(5p)**

6. Given a binary tree B of size n, design linear-time algorithms to determine

   (a) if B is an AVL tree. **(5p)**

   (b) if B represents a max-heap. **(5p)**

7. For each of the following statements, indicate whether the statement is TRUE or FALSE. **Justify your answers**. *That is, if the statement is correct, state why; and if the statement is wrong, give a counter-example.* No credit will be given without justifications.

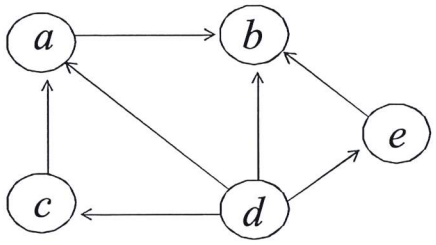
   (a) If $f(n) = O(g(n)$ and $f(n) = \Omega(g(n)$, then $f(n) = g(n)$. **(2p)**

   (b) $2^{2n} = O(2^n)$. **(2p)**

   (c) If the running time of an algorithm A is $\Theta(n^2)$ in the worst case, then the average-case complexity of A is also $\Theta(n^2)$. **(3p)**

   (d) If we need a data structure to support *insert* and *delete-min* operations, then a min-heap is the better choice than a balanced binary search tree. **(3p)**

8. Consider the following graph G:



   (a) Represent G by its adjacency matrix and adjacency list, respectively. **(5p)**

   (b) Draw a depth-first search *tree* and a breadth-first search *tree* of this graph starting at the node d, respectively. **(5p)**

   (c) Compute all the topological orderings of G. **(5p)**

9. Design a divide-and-conquer algorithm for computing the smallest and the largest element in an array of n (≥ 2) elements. Find the recurrence that computes the number of comparisons used by your algorithm in the worst case. Your algorithm should run in $O(n)$ time in the worst case. You may assume that $n = 2^k$. **(15p)**

10. Given two red-black trees T_1 and T_2 each of size $n \geq 2$, design an $O(\log n)$-time algorithm to determine whether or not there exist three keys a, b, and x such that

    - $x \in T_1$;
    - $a, b \in T_2$; and
    - $a < x < b$

    You may assume that all the keys are distinct. **(10p)**

11. Let $G = (V, E)$ be a connected, weighted, undirected graph (with n vertices and m edges) and T a spanning tree of G. Design an algorithm to determine whether or not T is a single-source shortest-paths tree rooted at s, where $s \in V$. Your algorithm should run in $O(m + n)$ time in the worst case. **(10p)**