

**58131 Data Structures**  
**Separate Exam 15.9.2009**

Write your name and student number on each paper.

In problems asking for an algorithm, you may use another pseudocode style than the one used in the course material, and you may also use e.g. Java.

You can get at most 12 points for each problem.

You are allowed to have an A4-sized sheet of paper of personal notes with you at the exam.

1. Consider the Catalan numbers  $C_0 = 1$ ,  $C_{n+1} = \frac{4n+2}{n+2}C_n$ , when  $n \geq 0$ . The Catalan numbers tell in how many different ways you can correctly nest  $n$  pairs of parentheses. For instance, when  $n = 2$ , the parentheses can be in the forms  $()()$  and  $(())$ , so there are two ways:  $C_2 = 2$ . The Catalan numbers also tell how many binary search trees there are with  $n$  nodes.
  - (a) Write a recursive function for calculating the Catalan numbers. What are the time and space complexities of the function?
  - (b) Write an iterative function for calculating the Catalan numbers. What are the time and space complexities of the function?

Note: There is a simple formula for the Catalan numbers,  $C_n = \binom{2n}{n}/(n+1)$ , but you are not allowed to use it here.

2. Explain the main principles of hashing: What is hasing needed for? What is a collision? Chaining vs. open addressing: what it is and when is one better than the other? You do not need to write down formulas — a general explanation is enough. A suitable amount of text is 1 – 2 pages with mid-size handwriting on each row of the paper.
3. Assume that we choose the median of  $A[p]$ ,  $A[\lfloor \frac{p+r}{2} \rfloor]$  and  $A[r]$  as pivot for the function  $\text{PARTITION}(A, p, r)$  in quicksort. What kind (easiest, hardest, something in between) are the following cases for this version of quickort. Explain why.
  - (a) [6 points] All are different numbers in ascending order, e.g., [1, 2, 3, 4].
  - (b) [6 points] All are different numbers in descending order, e.g., [4, 3, 2, 1].
4. A  $k$ -tree is a generalisation of a binary tree: the nodes have  $0, 1, \dots, k$  children. A full  $k$ -tree is one whose nodes have either 0 or  $k$  children. Deduce a formula for the number of nodes in a full  $k$ -tree as a function of the height.

Note: The answer is a lower bound and an upper bound.

5. Suppose we are given an undirected graph  $G = (V, E)$  and two nodes  $v$  and  $w$  in  $G$ . Give an algorithm that computes the *number* of shortest paths between  $v$  and  $w$ . Note that the algorithm needs not list the paths, it only gives the number of paths. The running time of your algorithm should be  $O(n + m)$  for a graph with  $n$  nodes and  $m$  edges.