58131 Data Structures (Spring 2009)

Course Examination 1, 23.2.2009

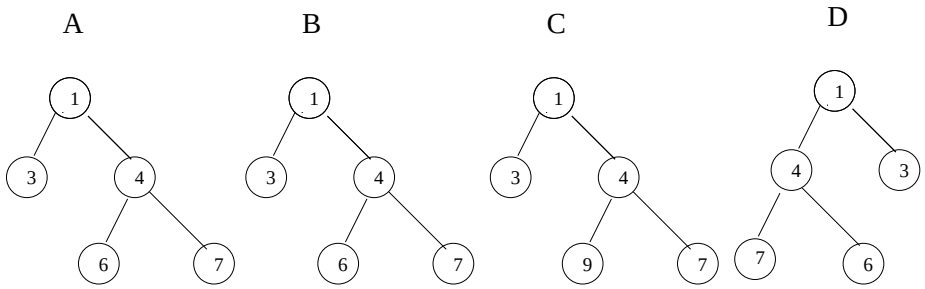
Give your answers to each problem (1, 2, 3) on separate sheets of paper. Write your name and student number on each paper.

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 8 points for each problem, in total at most 24 points.

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

1. Two binary trees are the same, if they have the same nodes (in the same order) and in the nodes the same data. Below, the trees A and B are the same, but for instance A and C are not the same, neither are A and D.



   (a) [*4 points*] Write a recursive function that checks if two binary trees are the same. The function gets as input pointers to both tree root nodes and it has to return TRUE if the trees are the same, otherwise FALSE.

   (b) [*2 points*] What is the time and space complexity of your algorithm, if the height of both trees is at most h and the number of nodes in both trees is at most n.

   (c) [*2 points*] How can you make your recursive algorithm non-recursive? What would in this case be the time and space complexity of the algorithm?

2. The Josephus problem is the following game: N persons are sitting in a circle. Starting at person 1 a hot potato is passed to person 2 and so on. After M passes, the person holding the hot potato is eliminated, the circle closes ranks, and the game continues with the person who was sitting after the eliminated person picking up the hot potato. The last remaining person wins. Thus, if $M = 0$ and $N = 5$, players are eliminated in order $1, 2, 3, 4$ and 5 wins. If $M = 1$ and $N = 5$, the order of elimination is $2, 4, 1, 5$ and 3 wins.

   (a) [*6 points*] Write a program that solves the problem of who is winning, given as input arbitrary integers $M \geq 0$ and $N > 0$.

   (b) [*2 points*] What is the time complexity of your program?

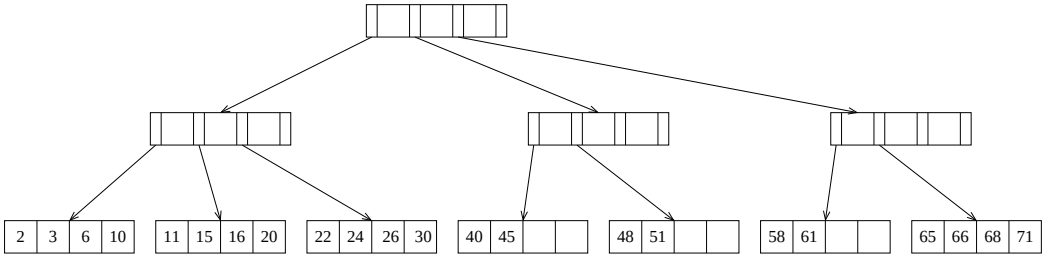
3. (a) [*2 points*] Complete the B^+ tree at the backside by inserting appropriate router values.

   (b) [*3 points*] The keys 27 and 7 are inserted in this order into the B^+ tree from part (a). Draw the main intermediate steps for each insertion.

   (c) [*3 points*] The keys 61 and 40 are deleted in this order from the B^+ tree resulting from the previous parts (a) and (b). Draw the main intermediate steps for each deletion.

   You do not need to redraw the parts of the tree that remain unchanged, as long as the changes are clearly drawn.

Please turn the page!



B^+ tree for Problem 3.