

Big Homework 3

Data Structures and Algorithms

BST & Heaps

General mentions

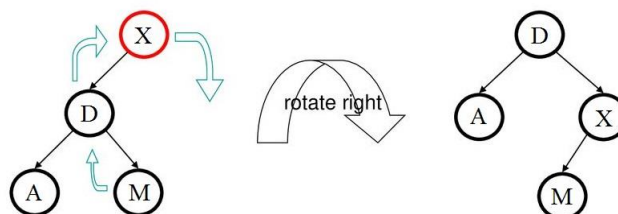
- For this project you will work either in teams of 2 people or alone.
- The homework will be uploaded on the Moodle platform (fils.curs.pub.ro), for Hw3 assignment. If you encounter problems with the platform, contact by email your lab assistant.
- The homework must be submitted by 19.05.2019, at 08:00 AM. No late submissions will be accepted.
- The evaluation of the project will take place during the first lab after the submission deadline. Presence at the lab will be mandatory for presenting your projects.
- The final submission will contain an archive named Student1FamilyName_Student1Name_Student2FamilyName_Student2Name_HW1 with:
 - the source files of your project (.cpp and .h) and not the object files (.o) or executables (.exe)
 - a README file in which you will specify all the functional sections of the project, together with instructions for the user; additionally, if you have parts of the homework that don't work, you may offer solution ideas for a partial score on these sections.
- For all questions regarding the project, communicate by email with your lab assistant.
- Warning: we will use plagiarism detection software on your submissions. Copied homework will be marked with 0 points.

1) BST (5p)

- a) Read letters from the keyboard and insert them in a BST. The reading process stops when the user inserts the character 0 (zero) (0.5p)
- b) Find the common ancestor of 2 given nodes (0.75p)
- c) Write a function to check if a node is a grand-parent of a given node. (0.75p)
- d) Design an algorithm which creates a linked list of all the nodes at each depth (i.e., if the tree has depth D, you'll have D linked lists). (1p)
- e) Check if a BST is perfect or not (all leaves are at the same distance from the root) (1p)
- f) Perform a right rotation of the BST. (1p)

More info: <https://slideplayer.com/slide/14553828/>

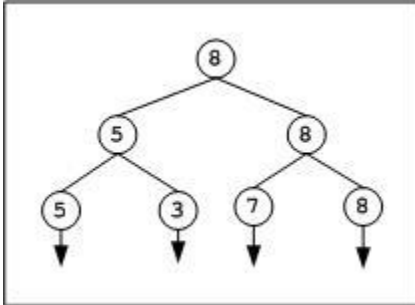
Ex:



2) Heap (5p)

A tournament tree is a form of max (min) heap which is a complete binary tree. Every external node (leaf) represents a player and an internal node represents a winner. We will thus have $N-1$ internal nodes in a binary tree with N leaves.

Ex:



The above (max) tree contains 4 leaf nodes that represent players (5, 3, 7, and 8). We have 3 levels: 0, 1 and 2. Initially 2 games are played at level 2, one between 5 and 3 and another one between 7 and 8. In the next move (level 1), one more game is conducted between 5 and 8 to conclude the final winner. Overall we need 3 comparisons to find the winner.

- a) Check if an input set can form a tournament tree after being inserted in a heap and display an appropriate message. (1p)

Ex: 3, 5, 5, 6, 6, 2, 5 – it can form a tournament tree

3, 5, 5, 6, 2 – it cannot form a tournament tree

Pick a valid set and insert it in your tournament tree.

- b) Determine who is the “second best” player. This is the person who lost in the first round the match against the final winner (here the second best is: 7) (0.5p)
- c) Display the history of the matches played by the winner at each level. (1p)

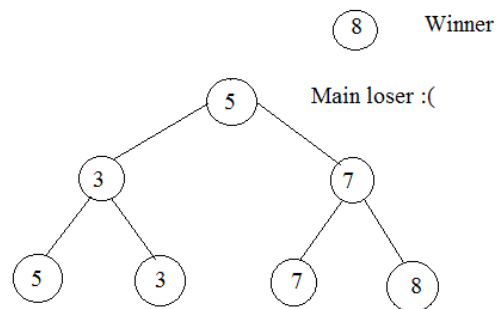
Ex:

In the first round, the winner defeated 7.

In the second round, the winner defeated 5 etc.

- d) Construct a “loser tree” using the previous info from the tournament tree (2p).

Ex:



In round 1, between 5 and 3, 3 is the loser; between 7 and 8, 7 is the loser. In round 2, we have the match between the 2 previous winners (5 and 8): the loser is 5. The winner is 8!

e) Determine the winner and the main loser from the loser tree (0.5p).