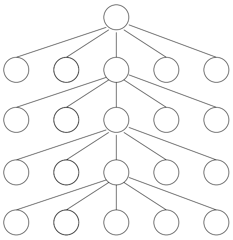
Data Structures and Algorithms I

Thu 23.2.2017

Exercise 5

25. Write an algorithm that finds the least deep (shortest path from root) leaf node of a binary tree. Hint: traversal by level. What is the time complexity of your algorithm?

26. Write an algorithm that creates and returns a general tree of height k. The tree has shape of a thick spruce tree. In a thick spruce of height k, there are k trunk nodes, all of which have five children. The middle child is again a trunk node having five children. Other nodes have no children. Deepest (depth $k-1$) trunk node has middle child that is a base node having no children. Attached figure presents a thick spruce of height 4. You can name the nodes as you wish.



27. Write an algorithm that gets as a parameter a collection (*Collection<E>*) and which creates and returns a mapping (function) (*Map<E, Integer>*) having as keys all different elements of the collection and as a value the number of each elements found in the collection. Hint: all collections provide *foreach* iteration. What is the time complexity of your algorithm?

28. Write an algorithm that gets as parameter two unsorted lists (A and B) and which creates and returns a new list which has all the elements that occur in both lists at least twice (at least twice in A *and* at least twice in B). Each element will, however, be in the result list only once. This differs from task X1 by having unsorted lists. Using algorithm of task 27 as helper, this can be done easily in linear time. Keep the elements in the result list in the same order as they were in input list A (hint: *LinkedHashMap*).

The following task X2 is obligatory for all students. X-tasks must be done **oneself** by each student. Copies/versions of the same answer won't be accepted. Answers must be sent by Wed 22.2. 21:00 using the instructions below. You'll receive an automatic reply by email soon after successful submission. If you won't get the email reply, something went wrong. If the reply contains compiler errors, there is something wrong in the file. Then **resend a fixed version**. The answer must contain a short **self evaluation** where you evaluate the functionality, correctness, time complexity, and possible points of improvement of your solution. A correct self evaluation (for a full answer) is worth one point. The points of these tasks form a part of course evaluation.

Send your solution using a www-form, address and credentials of which you got by email. The solution should be a compilable Java source code file of name `userid.java` where `userid` is the first part of your email address. Also the **class name** of your solution must be `userid`. As the submission is Java source code, the self evaluation must be in comments of the program.

Take a skeleton from course www-page. Do not change the header (name, parameters) of the X-task method. Please make sure that the program is compilable as such, i.e., have whole answer in the same class and do not use a package.

X2. Write two algorithms that build a sorted array of an in-ordered binary tree, and vice versa. Using these algorithms and algorithm like in task 11, we can implement set difference, union, etc. in linear time.

First algorithm gets as parameter an in-ordered binary tree (*BTree*) and it creates and returns an array (*ArrayList*) to which it has stored the elements of the tree in the same order as the elements are in the input tree.

Second algorithm gets as parameter a sorted array (*ArrayList*) and creates and returns a balanced (as low as possible) in-ordered binary tree. Neither of algorithms may change the input.

For full points, both algorithms must have linear time complexity. Building a balanced tree in linear time is possible by starting from the middle element of the input array and setting it as the root of the new tree, and creating recursively left and right sub-tree from remaining start and end halves of the array. Building can be done also using other methods, but then time complexity and/or tree height may increase. Take a skeleton from course www-page. Do not change the method names or parameters.