CPSC 331 26. November 2015

University of Calgary

Homework Assignment # 3

Due: 8. December 2015, **11:59pm** via the Dropbox on D2L (This time, we accept only electronic submissions.)

Late assignments will not be accepted.

For this assignment you are NOT allowed to collaborate with other students!

You are allowed to use literature, as long as you cite that literature in a scientific way (including page numbers, URLs, etc.). In any case, your solutions must be self-contained. For example, if you use a theorem or lemma that was not covered in the lecture and that is not "well-known", then you have to provide a proof of that lemma or theorem. All your work, including source code, **must be written by yourself**.

If you are in doubt whether a certain form of aid is allowed, ask your instructor!

Academic misconduct (cheating, plagiarism, or any other form) is a very serious offense that will be dealt with rigorously in all cases. A single offense may lead to disciplinary probation or suspension or expulsion. The Faculty of Science follows a zero tolerance policy regarding dishonesty. Please read the sections of the University Calendar under the heading "Student Misconduct".

Submission Information

Additional submission information will be posted on D2L. Please note that revisions to the provided files may also be posted on D2L, if necessary.

Question 1

Use pseudocode to describe a sorting algorithm using **one stack**, **one queue**, and a constant number of variables. The integers to sort are initially stored in unsorted order on the stack; once the algorithm terminates, they should appear in sorted order on the stack, with the largest element at the bottom. Your pseudocode can use the following data items:

- A stack S which can store elements of type `int`, and which supports the operations `push`, `pop`, `peek`, and `isEmpty`;
- a queue Q which can store elements of type `int`, and which supports the operations `insert`, `remove`, `element`, and `isEmpty`;
- a constant number of variables of type `int`.

No other data structures can be used by your algorithm.

Your algorithm should satisfy the following pre- and postconditions:

Precondition: The stack S contains n distinct integers $a_1, \ldots, a_n$ and the queue Q is empty.

Postcondition: The stack S contains n distinct integers $b_1, \ldots, b_n$ in the order from bottom to top, such that $b_1 > b_2 > \cdots > b_n$ and $\{a_1, \ldots, a_n\} = \{b_1, \ldots, b_n\}$.

Ideally, your algorithm has a worst-case running time of $O(n^2)$, assuming each stack and queue operation takes only constant time. You will get partial marks for an implementation that does not achieve this running time.

Argue informally why your algorithm is correct and state its running time.

Question 2

With this assignment you receive a file `Assignment3.zip`. Unzip it obtain several Java files. You don't have to change any of the following files:

- `Main.java`: Contains the `Main` class to test your code.
- `DouplicateKeyException.java` and `KeyNotFoundException.java`: Contains classes for exception handling that you can (and should) use in your implementations.
- `HashFunction.java`: Contains the interface `HashFunction`. The classes `DoubleHashing`, `LinearProbing`, and `QuadraticProbing` (provided in the corresponding Java-files), extend this interface.

Implement a dictionary based on open addressing, using linear probing, quadratic probing, and double hashing. To do that, implement the methods of the class `OpenAddressing`, as well as the `HashFunction` classes `LinearProbing`, `QuadraticProbing`, and `DoubleHashing`, provided in the corresponding Java-files.

You can and should test your code using `Main.java`.

Question 3

On the following pages you find eight figures, each of them labeled with "Group x" for some number $x \in \{0, \ldots, 7\}$. Let y be your UofC Student ID. Your group is $x = y \bmod 8$.

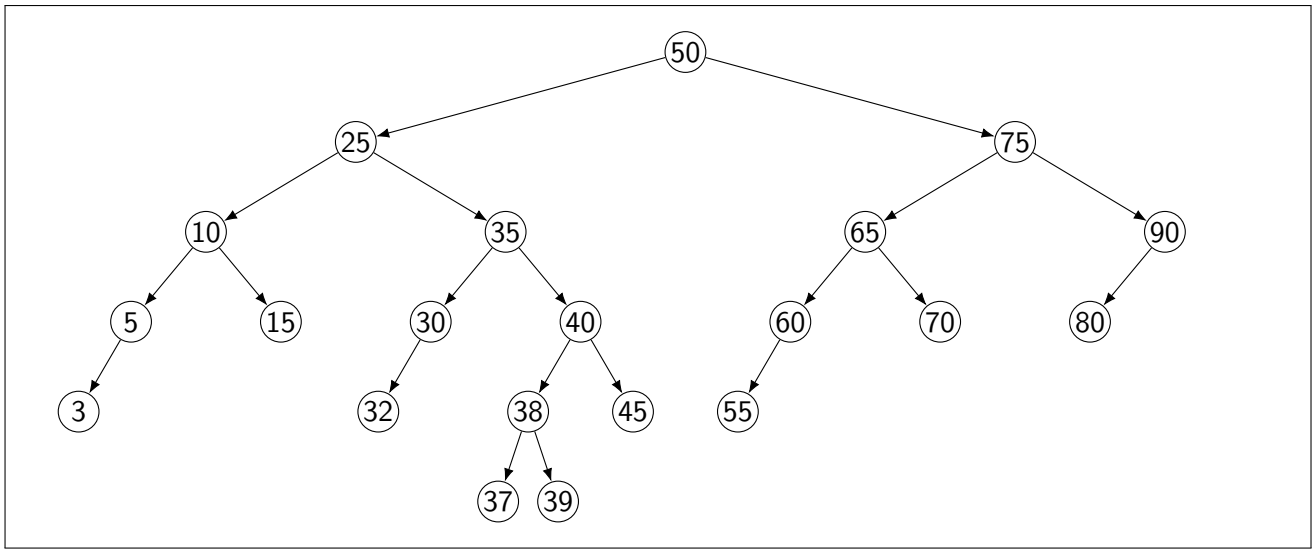
Consider the figure labeled Group x, where x is your group number. This question refers to that figure. Please make sure that you use the correct figure, as otherwise you may receive 0 points for this question.

(a) Write down "Student ID:" followed by the value of y (your UofC Student ID), and "Group:" followed by the value of x (your Student ID mod 8).

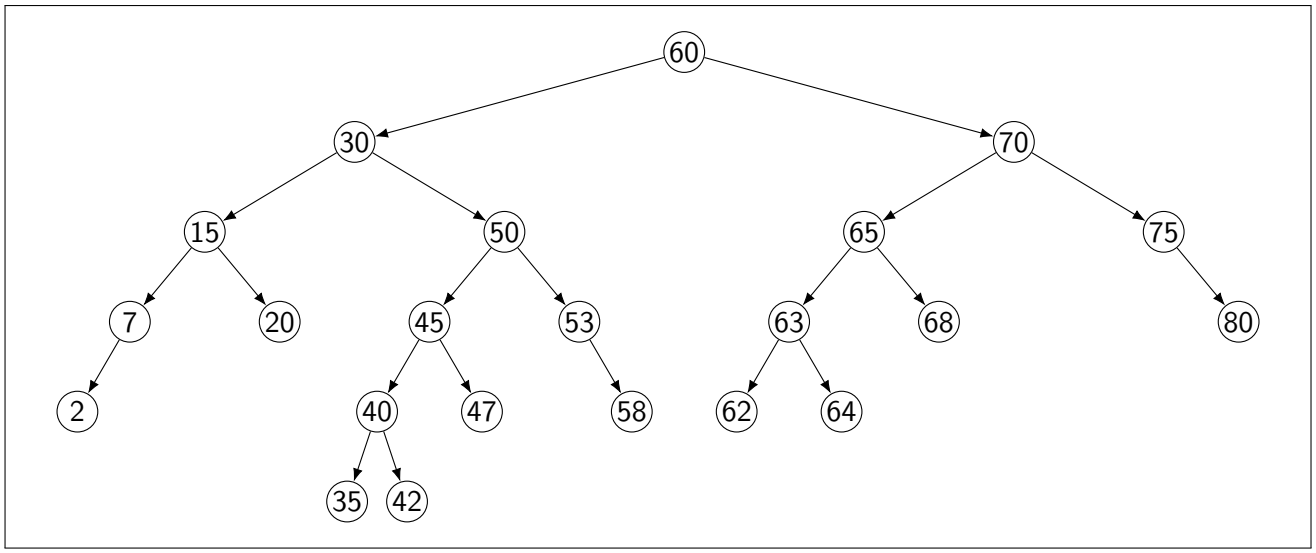
(b) The tree given in the figure with label "Group x" is an AVL Tree T. Label each node v of the tree with a pair "$h(v)/\varphi(v)$", where $h(v)$ is the height of node v, and $\varphi(v)$ is its balance factor.

(c) Perform the AVL Tree *insert* operation on T that is specified in the label of the figure. Draw the tree obtained after a node has been added to T and before any rotations have been performed, as well as each tree obtained after a rotation or double rotation that is being performed during the insert operation.

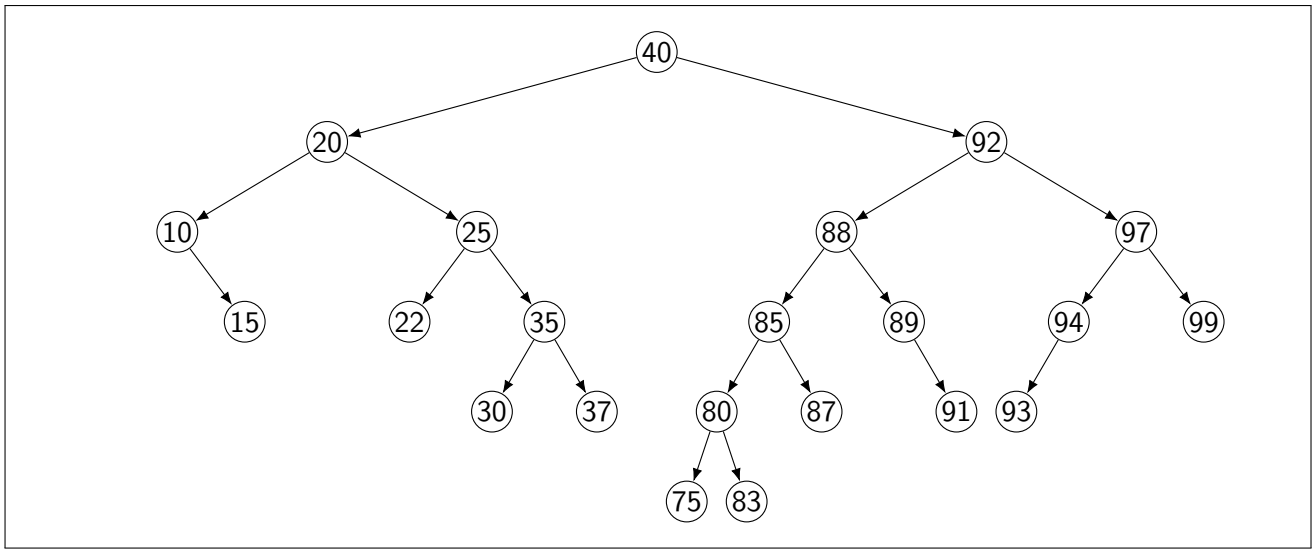
(d) Perform the AVL Tree *delete* operation on T that is specified in the label of the figure. (Use the original tree depicted in the figure, and **not** the one obtained as a result of the insertion above.) Draw the tree obtained after a node has been deleted from T and before any rotations have been performed, as well as each tree obtained after a rotation or double rotation that is being performed during the delete operation.



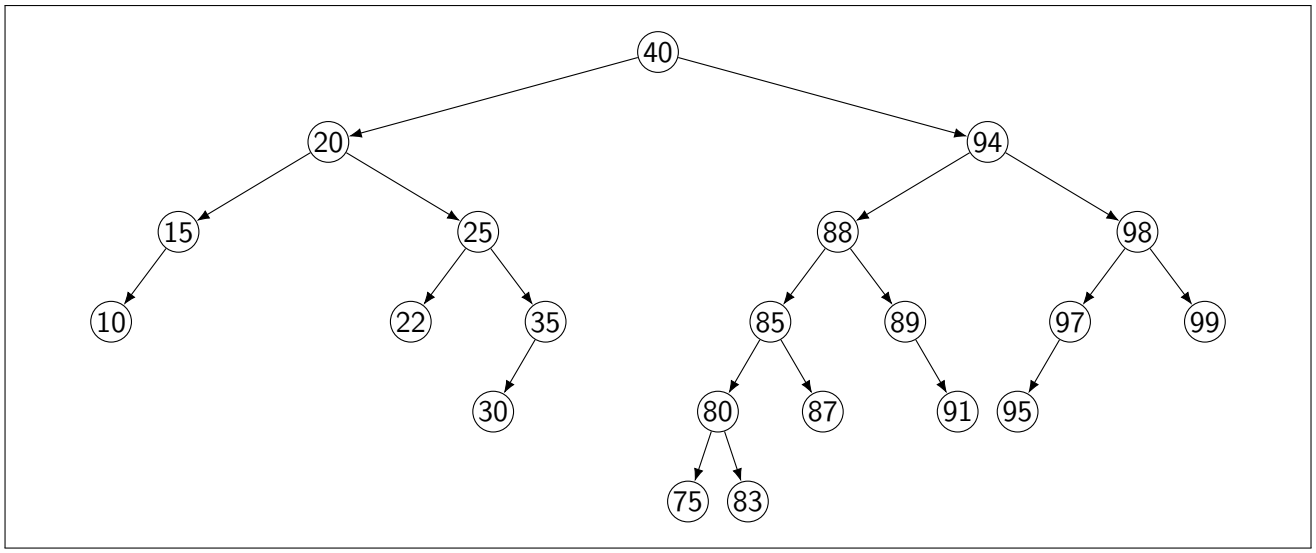
Group 0: Part (c): insert(36). Part (d): delete (75).



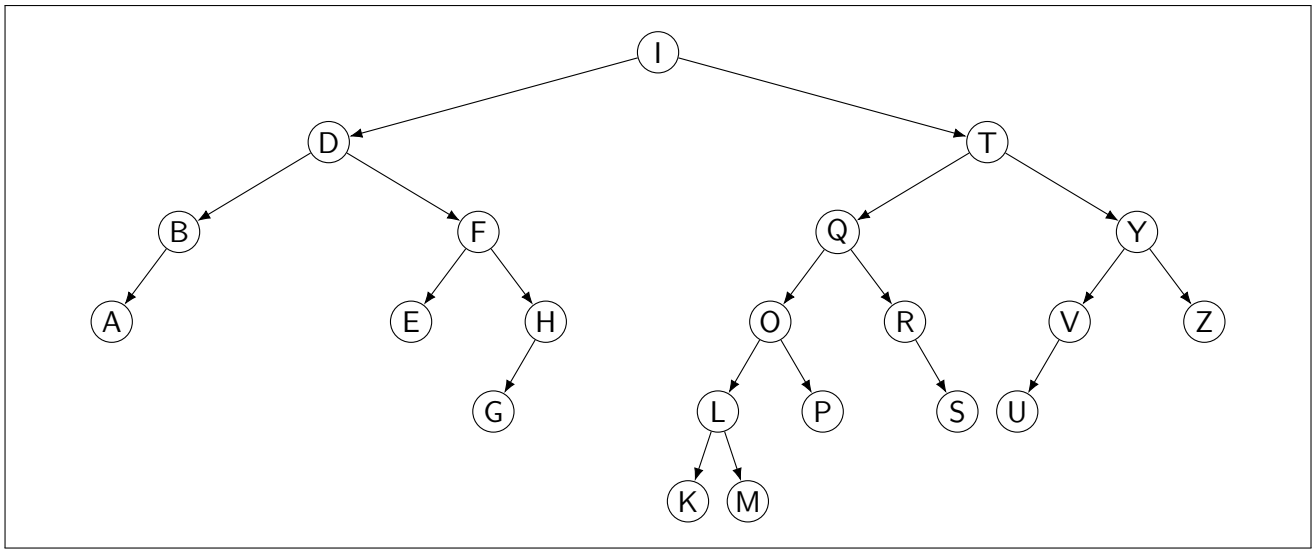
Group 1: Part (c): insert(36). Part (d): delete (70).



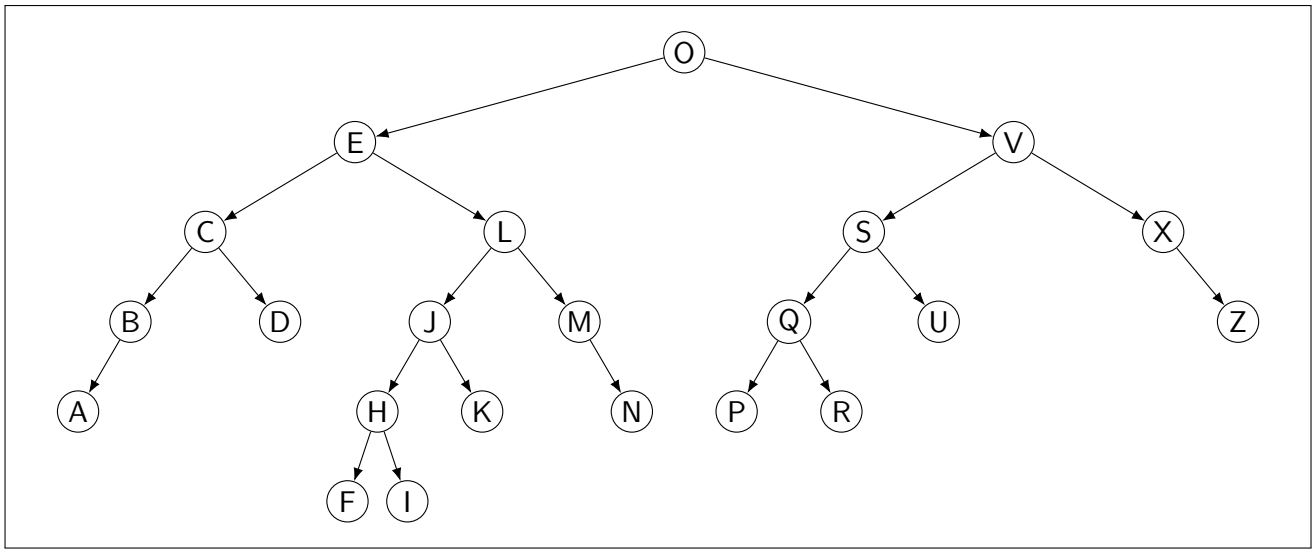
Group 2: Part (c): insert(78). Part (d): delete (20).



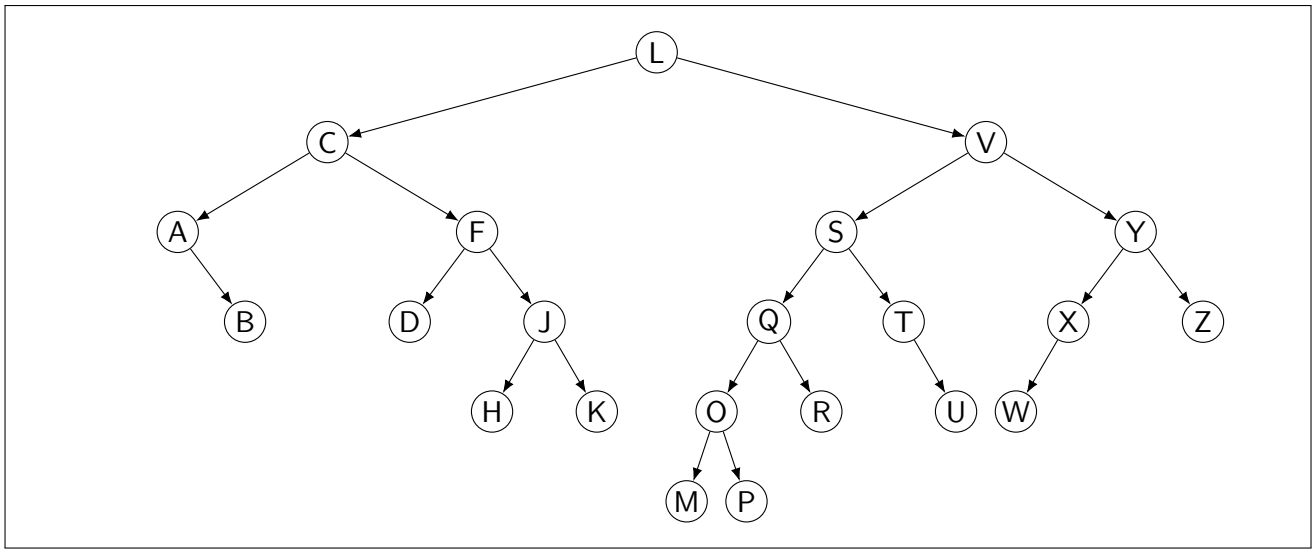
Group 3: Part (c): insert(70). Part (d): delete (20).



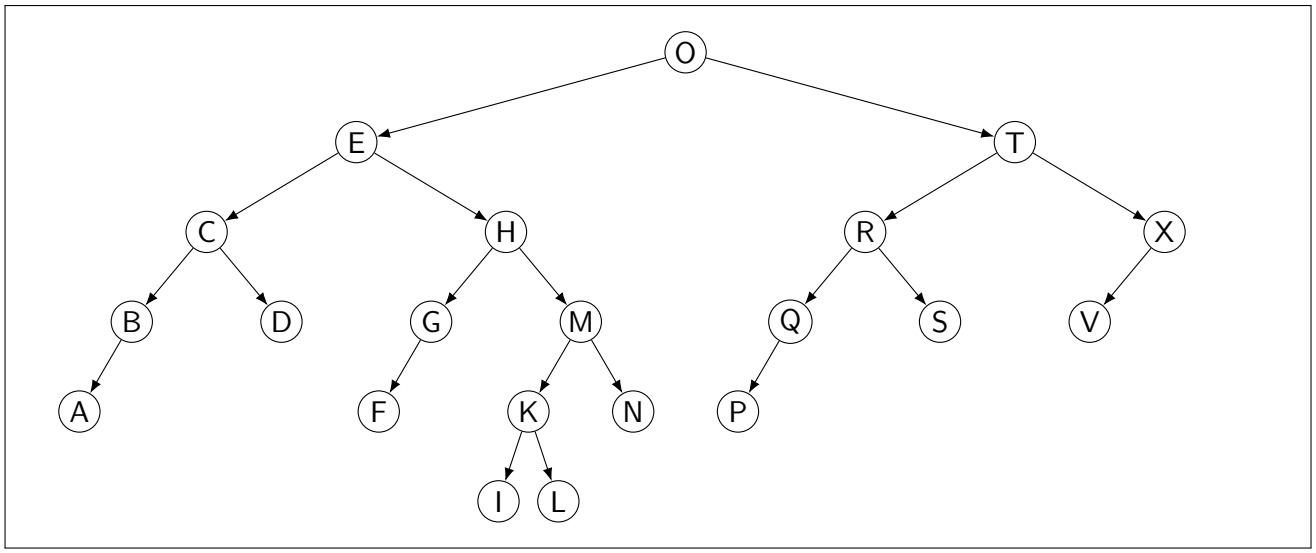
Group 4: Part (b): insert(J). Part (c): delete (D).



Group 5: Part (b): insert(G). Part (c): delete (V).



Group 6: Part (b): insert(N). Part (c): delete (C).



Group 7: Part (b): insert(J). Part (c): delete (T).