Department of Computer Science and Software Engineering
Concordia University

COMP 352: Data Structure and Algorithms
Winter 2020
Assignment 3
Due date and time: Friday March 13th, 2020 by midnight

Written Questions (50 marks):

Question 1

A company is involved in shipping large number of containers at a dock. Containers are classified into three categories:
Category 1: containers need to be accessed by their indices: *lookup*, *set*, *add* and *remove* from Category 1.
Category 2: containers need to be *add* and *remove* before or after certain position including the first and last position of the Category 2.
Category 3: containers need to be *add* and *remove* in a sorted alphabetical order of their destination and any new added container need to follow that order of the Category 3.

From the three linear ADTs: List, Positional and Sequence covered in class. discuss which ADT to choose and its underlying implementation for the above containers' categories.

Question 2

a) Draw a single binary tree that gave the following traversals:

```
Inorder:    C  O  P  Y  R  I  G  H  T  A  B  L  E
Postorder:  C  P  O  R  G  I  T  H  B  A  E  L  Y
```

b) Assume that the binary tree from part (a) of this question is stored in an array-list as a complete binary tree as discussed in class. Specify the contents of such an array-list for this tree.

Question 3

a) Given a tree T, where *n* is the number of nodes of *T*.
Give an algorithm for computing the depths of all the nodes of a tree T. What is the complexity of your algorithm in terms of Big-O?
b) We say that a node in a binary search tree is full if it has both a left and a right child.
Write an algorithm called *Count-Full-Nodes(t)* that takes a binary search tree rooted at `node t`, and returns the number of full nodes in the tree. What is the complexity of your solution?

Question 4

a) Draw the min-heap that results from the bottom-up heap construction algorithm on the following list of values:

20, 12, 35, 19, 7, 10, 15, 24, 16, 39, 5, 19, 11, 3, 27.

Starting from the bottom layer, use the values from left to right as specified above. Show immediate steps and the final tree representing the min-heap. Afterwards perform the operation `removeMin` 6 times and show the resulting min-heap after each step.

b) Create again a min-heap using the list of values from the above part (a) of this question but this time you have to insert these values step by step using the order from left to right (i.e. insert 20, then insert 12, then 35, etc.) as shown in the above question. Show the tree after each step and the final tree representing the min-heap.

Note: You must submit the answers to all the questions above. However, only one or more questions, possibly chosen at random, will be corrected and will be evaluated to the full 50 marks.

Programming Questions (50 marks):

In class, we discussed evaluating arithmetic expressions using a binary tree, this binary tree has leaves that are associated with variables or constants, and whose internal nodes are associated with one of the operators: +,-,* and /. You can assume parentheses are well-matched, and the only binary operations allowed are +, -, *, and /. In this programming assignment, you will design, using pseudo code, and implementation in Java code, a new version of arithmetic calculators that uses expression trees. Your solution should perform the following:

1. read valid arithmetic expressions (parentheses are well-matched), where each expression is in a single line. Each arithmetic expression with operands (variables and/or constants) and operators.
2. then convert it to an arithmetic expression in a binary tree structure, where each, internal node represents an operator, and its subtrees represent operands. All operators are binary, so every operator has both a left and a right operand. Leaf nodes represent either integers or variables.
3. use a node in an expression tree. Nodes could be, a *leaf* node just contains a value (integer or variable); or an *internal* node has an operator and two Nodes representing its operands.
4. if the expression contains operand variables, it should prompt the user with those variables to set their values in order to evaluate the expressions.

a) Briefly explain the time and space complexity for the binary tree arithmetic expressions calculator. You can write your explanation in a separate file.
b) Provide test logs for at least 10 different and sufficiently complex arithmetic expressions that use any of +, -, *, and / operators (including parentheses) in varying combinations, and operand both as variables and constants.

you are required to submit:
- i. the pseudo code.
- ii. a fully commented Java source files (.java files), and the explanation text file.
- iii. and your experimental results.

Important Notes

The programming part can be done in groups of two students maximum.
For the Java programs, you must submit the source(.java) files. The solutions to all the questions should be zipped together into one .zip or .tar.gz file and submitted the EAS folder/ Moodle Dropbox. You must upload at most one file (even if working in a team, here is what you need to do:

1) Create **one** zip file, containing the necessary files (.java, .doc, .html, etc.). Please name your file following this convention:
If the work is done by 1 student: Your file should be called *a#_studentID*, where # is the number of the assignment *studentID* is your student ID number.

If the work is done by 2 students: The zip file should be called *a#_studentID1_studentID2*, where # is the number of the assignment, and *studentID1* and *studentID2* are the ID numbers of each student.
2) If working in a group, only one of the team members can submit the programming part. Do not upload 2 copies.

Important: for the programming part of the assignment, a demo is required (please refer to the courser outline for full details). The marker will inform you about the demo times. **Please notice that failing to demo your assignment will result in zero mark regardless of your submission.** If working in a team, both members of the team must be present during the demo.

Submission:

- The **written questions part** must be done **individually** and uploaded to "theory assignment 3" via EAS/ Moodle, name your file *A3_studentID. pdf*, where *studentID* is your ID number. For this third assignment, student 123456 would submit a file named A3_123456.pdf. All your answers to written questions must be in PDF (no scans of handwriting) or text formats only. **Please be concise and brief (about ¼ of a page for each question).**
- The **programming part** must be done **in a team of 2 students max,** For the Java programs, you must submit the source files together with the compiled files. The solutions to all the programming questions should be zipped together into one .zip or .tar.gz, name your file *A3_studentID1, studentID2. zip,* where *studentID1, studentID2, are the IDs of the three students who did the programming part together,* uploaded to "Programming assignment 3" via EAS/Moodle.

Note: **Assignment not submitted by the due date and in the correct format/ folder will not be graded – NO EXCEPTIONS!!!!**