COSC 3333 - Data Structures & Algorithms II Spring 2022

Programming Assignment 4 **Due Date: Sunday, April 3 @ 11:59 p.m**

Download the code provided by the textbook in order to complete this program.
This is a two part assignment. **Part II is a BONUS part.**

Part I: An experimental Algorithm for mapping a Heap Array into a Binary Tree

You are asked to write a program that maps a Heap that is stored in an array to a Binary Tree. The following is the list of steps to take:

1. Instantiate an array of 10 Node objects. You will find the class Node in the book. You need the Node class with the integer key attribute as well as the left and the right child.

2. Each Node object need to have an integer attribute that you randomly create in the range 10 - 50. Make sure you do not have any duplicates. Modify the Node class if needed.

3. Use the code in the book to build a Heap out of this array. Make sure you modify the book's code properly. The Heap will be built based on the integer key attribute of the Node.

4. Our algorithm approaches the task of mapping this Heap Array into a Binary Tree in a bottom up fashion. This means that we will start with the last parent, find its possible children, attach them to the parent and move to the next parent (which will be the one before last, etc) in line and do the same.

Following this algorithm the original parent nodes that are the occupants of our Heap Array may grow into subtrees as we connect them with their possible children.

When you are done with this step of the algorithm, in order to show your work, iterate through the array and display its contents. Keep in mind that each cell will have a subtree or a single node which can be considered a one-node subtree. Use the display method in the book that is used to display trees, that method needs to have access to the root of a given tree, you can pass the node in each cell of this array as the root, and have this method display it as a tree/subtree/possibly a single node. Feel free to make any changes in the display method in the book if there is a need.

Would you be done building the tree when you reach to the first element of the array?

Now you must provide answers to the following questions in a report file similar to what we used to have in some previous assignments. Questions are numbered and appear in italics:

Q1. *When you are done with this algorithm, what do you find in the first element of the Heap Array?*

Q2. *Discuss this mapping algorithm in terms of:*

1. *Running time in Big O notation*

2. *Memory cost*

Q3. *Discuss the advantages and disadvantages of this algorithm. Be specific.*

Part II. ************************** BONUS **************************

This is a bonus option for those who are interested in both the challenge and the extra credit. I strongly recommend that you try this part. This part will earn you potentially **50 bonus points**.

A side note: Please note that these are purely exercises with data structures and strategies and by no means offer/claim the most efficient or feasible solutions to proposed problems.

For this part we would like to experiment with a different approach in mapping a Heap Array into a Binary Tree. This time we will use a Queue to help us with this process.
Here are some guidelines about how to implement this algorithm.

- We again start with the last parent, when we find their possible children, we connect them with the parents but this time we enqueue this subtree in a Queue of proper type.
- We look for the children of the parents that are part of the Heap Array as long as their possible children are not parents themselves.
- When we realize that we are processing parents whose children are parents themselves, we will look for the children in our Queue, by dequeueing the first two subtrees/children off the Queue and connect them with the parent and enqueue this "family" back in the queue. I let you decide which will be attached as the left child and which will be the right child.
- We continue with this process until we are done with the "root" of the Heap.

What you need to display for this part is the content of the Queue at every step of this algorithm. Again use the Tree display method provided by the book.
Note that you need to modify the Queue class and any code that you use from the textbook to work for our problem here.

Now the report for this part:

Bonus Q. *Compare this algorithm with the one in Part I. Frame your argument in terms of advantages and disadvantages in terms of running time, memory cost and programming convenience. Be specific, clear and organized in making your case.*

- *The program must be written in Java.*
- *You must upload all relevant .java files, including the ones from the book, on blackboard to ensure successful execution of your code, by the due date.*
- *The submission that is missing required files/classes will be considered void.*
- *Programs that do not compile will be considered void.*
- *This is an individual assignment. If students submit fully or partially identical programs, all individuals involved will earn a grade of zero.*