CSC 228 Data Structures and Algorithms, Fall 2017
Instructor: Dr. Natarajan Meghanathan

Project 10: Checking whether a Binary Tree is a Max Heap

Due: November 29, 1 PM (submission through Canvas)

A binary tree is a max heap if it satisfies the following two properties: (i) the binary tree is essentially complete and (ii) the data for every internal node is greater than or equal to the data of its two child nodes.

In class, we saw the BFS-based algorithm (pseudo code is given in the slides) to determine whether a binary tree is essentially complete or not. The idea is briefly as follows: We do a BFS of the nodes starting from the root node. We keep track of whether we come across a state wherein an internal node does not have a child node. The moment we come across an internal node with a missing Child node (left node or right node), we set the boolean 'noChildZoneStarts' to true. If we come across an internal node with a child node when the boolean noChildZoneStarts is true, we declare the tree is not essentially complete! If we are done with BFS and do not come across any internal node with a child node after the boolean noChildZoneStarts is set to true, we say the binary tree is essentially complete!

To implement the isMaxHeap( ) function, you could go over the arrayOfBTNodes[...] from index (n/2)-1 and check whether for each internal node, the data at the internal node is greater than or equal to the data of its child node(s).

Consider the code for the binary tree given to you for this question. Add code in the blank space provided for the member function isEssentiallyComplete( ) and isMaxHeap( ) in the BinaryTree class. These member functions should check whether the binary tree satisfies the two requirements to serve as a Max Heap. The input by the user is a binary tree (in the form of the edges and data information stored in two separate files; similar to Project 7 wherein you will have to input two files: one file for the edge information and the other file for the data associated with the nodes).

You are provided a sample of two files that constitute the edges and data for a binary tree that is also a max heap. You could validate your code with these two files. The main function of the code given to you is already updated to input the above two files. Your task is only to implement the isEssentiallyComplete() and isMaxHeap() functions and test their working through files that represent the edges and data for a binary tree.

Your main function should be able to test for three scenarios: (a) Binary tree that is essentially complete and a max-heap; (a) Binary tree that is essentially complete, but not a max-heap; (c) Binary tee that is not essentially complete (in this case, we need not check for the max-heap property).

Make small changes to the input text files (that currently reflect scenario-a) given to you in such a way that the corresponding binary trees reflect scenarios-b and c. Capture the screenshots of the outputs.

What to submit (in a word document, through Canvas):
The complete code, including the implementation of the isEssentiallyComplete() and isMaxHeap() functions in the BinaryTree class as well as the main function.
Draw the binary trees corresponding to the three scenarios (a), (b) and (c) and include screenshots of the outputs obtained by running the code with the three scenarios of input files.