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

## Project 7: Binary Tree: Checking for Complete Binary Tree and Binary Search Tree

**Due:** November 3, 1 PM (submission through Canvas)

Q1 - 50 pts) A binary tree is a complete binary tree if all the internal nodes (including the root node) have exactly two child nodes and all the leaf nodes are at level 'h' corresponding to the height of the tree.

Consider the code for the binary tree given to you for this question. Add code in the blank space provided for the member function checkCompleteBinaryTree( ) in the BinaryTree class. This member function should check whether the binary tree input by the user (in the form of the edge information stored in a file) is a complete binary tree.

To test your code, come up with two binary trees of at least 12 vertices: one, a complete binary tree and another, a binary tree that is not a complete tree.

Prepare the input file for the two binary trees and input them to the code for this question. Capture the screenshots of the outputs.

Q2 - 50 pts) A binary search tree (BST) is a binary tree in which the data for an internal node is greater than or equal to the data of its left child node and lower than or equal to the data of its right child node.

Consider the code for the binary tree given to you for this question. Add code in the blank space provided for the member function checkBST( ) in the BinaryTree class. This member function should check whether the binary tree input by the user (in the form of the edge and data information stored in two separate files) is a binary search tree.

You are provided a sample of two files that constitute the edges and data for a binary tree that is also a binary search tree. 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 checkBST() function and test its working through files that represent the edges and data for a binary tree.

To test your code, come up with two binary tree of at least 12 nodes as follows:
(i) A binary tree that is also a BST with the data of the nodes setup in such a way that the data for an internal node is greater than or equal to its left child and lower than or equal to its right child.
(ii) In the binary tree that you came up with for (i), change the data of the nodes in such a way that for at least one internal node, the BST requirement is violated so that the binary tree is not a BST.

Prepare the input files (edges and data files) for the two scenarios above and test your code with these files.

**Submission:** Submit all the following in a single word document.
(1) For Q1, draw the two binary trees (along with their node ids) that you have come up with, include the text files (corresponding to these two binary trees) that are input to the program and the complete C++ or Java code to test your program as well as screenshots of the outputs.
(2) For Q2, draw the two binary search trees (along with their node ids and data) that you have come up with, include the text files (the edges and data files for the corresponding two binary trees) that are input to the program and the complete C++ or Java code to test your program as well as the screenshots of the outputs.