CSUC CSCI-311 - Algorithms and Data Structures
Lab Assignment 3

Goal: 1) practice using BST.

In this lab we will begin the process of building our own binary search tree class by implementing methods for verifying that a tree is a binary search tree, inserting values into a binary search tree, searching for a given value in a binary search tree, and performing a pre-order traversal of a binary search tree.

Submission: C++ solutions to these problems should be written in a file called BST.cpp (a skeleton is available on Canvas along with BST.h, Node.h, Node.cpp, and BSTDriver.cpp as well as the test cases) and the file should be submitted on inginious (https://inginious.csuchico.edu/).

Coding Style: Note that your submission should use good C++ coding style and should include appropriate comments (for readability and maintainability). Specifically, your code must follow common C++ coding conventions for Naming, Indentation, and Comments. Points will be deducted if these are not present.

Collaboration: There will be time in lab to discuss these problems in small groups and I highly encourage you to collaborate with one another outside of class. However, you must write up your own solutions **independently** of one another. In addition, do not post solutions in any way. Also, please include a list of the people you work with in a comment section at the top of your submission.

Have fun!

Assignment Date:	**Feb 24, 2025**
Due Date:	**11:59pm on Mar 06, 2025**
Grace Due Date:	**11:59pm on Mar 09, 2025**

Grading Notes: 1) Assignment is due on the Due Date. 2) For each day late after the Due Date, there will be 10% penalty on the assignment's grades. 3) Submission is not accepted after the Grace Date. In other words, you will receive zero pts if your submission is not received by the Grace Date.

Special Nodes: Remember to keep a copy of your solution "BST.cpp" to use as a skeleton source code file for Lab 4 in the next week.

Grading: Coding Style 5 pts, Test Cases 95 pts, Total 100 pts.

Assignment Questions (95 pts)

1) Implement the insert method for binary search trees. Do not insert duplicate values.

   *NOTE: IT IS ASSUMED THAT WE DO NOT ALLOW THE INSERTION OF DUPLICATE VALUES INTO THE TREE.

   A potentially useful code snippet for creating an object using a shared pointer is given as follows.

   ```
   std::shared_ptr<Node> n;

   n = std::shared_ptr<Node>(new Node(val));
   ```

2) Implement the search method for binary search trees.
3) Implement isBST, a method for checking whether or not a tree is a binary search tree.
4) Implement the pre-order traversal method for binary search trees.