Assignment 0 – CSC 300 Review

Spring 2026

Due: Monday, Feb 2 11:59 pm

To-do

Implement a Binary Search Tree (BST) (taken from your previous CSC300 class) in C++.

Task 1: Include the basic operations ***printBST***, ***insert***, ***delete (handle all three cases)***, and ***search***.

Task 2: Include additional operations for ***traversal (inorder, preorder, and post order)***, ***check if tree is Balanced***, and ***find the height of a given node***.

Task 3:

- Implement a detection mechanism to identify when a BST has reached a worst-case configuration.
- Experiment with and generate friendly and adversarial inputs, then measure the resulting cost of operations.

Task 4: Use header files to organize your code.

- Use three files bst.h, bst.cpp, and mainbst.cpp

Task 5: Use the provided exception handling to manage exceptions and edge cases

- Ensure that you are handling edge cases like searching in an empty tree, inserting duplicate values, …

Task 5: Use make files to automate compiling and running your program (use version 5)

Total Points (50)

- Code runs and works as expected – Task 1: 20 points
- Code runs and works as expected – Task 2: 10 points
- Used header files – Task 3: 5 points
- Used exception handling – Task 4: 5 points
- Used make files – Task 5: 5 points
- Available on GitHub: 5 points

Deliverables:

- A zipped folder named A0 that contains all the files
    - C++ files and header files
    - README.txt file [similar to the README file included in the array code]
    - A screenshot showing the functions work.
- Published to D2L dropbox and GitHub under the folder where you added me as a collaborator.