CSE 2383: Data Structures and Algorithm Analysis Challenge 4 – Linked Lists + BST

Submission Window Opens:
Friday, November 15

Points Available:
90 points for a working demonstration
10 points for correct submission & understandable code

Objectives:

- Demonstrate working BST and Linked List implementations
- Calculate runtime for searching a given value in both BST and Linked List
- Demonstrate sufficient knowledge to implement data generation and runtime measurement for both data structures

Assignment:

For this challenge, you must write a program that compares the runtime performance of searching for a given value in both a Binary Search Tree (BST) and a Linked List, derived from your existing code. There are two stages of code development (additional details are below): code completed prior to your demonstration and code completed during your demonstration. You must demonstrate working BST and Linked List implementations to the TA.

Prior to demonstrating your code:

1. Use your existing BST and Linked List code implemented in previous challenges.
2. Write code to generate random numbers to insert into the Linked List and the BST in for the following quantities:
    - 500,000 thousand random numbers (between 1 and 1000)
    - 1,000,000 million random numbers (between 1 and 1000)
    - 2,000,000 million random numbers (between 1 and 1000)
    - 1 through 10k backwards (i.e., 10,000 down to 1)
    - 1 through 10k forwards (i.e., 1 up to 10,000)

3. Populate both the BST and Linked List with the generated random numbers for each scenario.
4. Write code to search for a given value in both the BST and Linked List and record the runtime of each operation (Total time to insert all elements, total time to find search value). The search function must return a Boolean value of true if the value is found and a Boolean value of false if it is not found.

To be completed DURING YOUR DEMO:

1. Demonstrate the code you have written for generating random numbers, populating the BST and Linked List, and measuring the runtime for searching a given value.
2. Delete the random number generation code in front of the TA.
3. Reimplement the random number generation code from memory, and explain the logic behind it.
4. Explain how the code works and discuss the performance differences between the BST and Linked List implementations based on your runtime measurements.

As with the previous challenge, you must fully pass the demonstration before a grade will be assigned to your submission. Grades are assigned based on the day of submission (see syllabus). Once you have demonstrated your submission, you must upload it to Canvas within 48 hours.

You will need to bring your laptop to the TA for demonstration. If you cannot bring your laptop, make arrangements ahead of time with the TA to demonstrate it some other way. Arrangements must be made at least a day ahead of your planned demonstration.

Your class's TAs are:

Jacob Adams, jra457@msstate.edu, T Th F 12:00pm – 2:00pm
Robert Dilworth M W 3:00pm – 4:30pm, F 12:30pm – 3:30pm

Deliverables

1. Demo your working code to the class TA before uploading it to Canvas. You cannot proceed to step 2 before doing this.
2. Once your code is working and you've demoed it to the TA, upload all your .cpp and .h files to Canvas <netID>_2383_Ch<challenge_number>.<file_type>, where <netID> is your MSU Net ID, <challenge_number> is the number assigned to the Challenge, and <file_type> is either cpp, h, or hpp.