CALIFORNIA STATE UNIVERSITY, FRESNO
DEPARTMENT OF COMPUTER SCIENCE

Class:	**Algorithms & Data Structures**	Semester:	**Fall 2023**

Points		Document author:	**Pratham Aggarwal**
		Author's email:	**Pratham_aggarwal@mail.fresnostate.edu**
		Laboratory number:	**Lab 8**

1. Statement of Objectives

The goal of this experiment was to create a Binary Search Tree (BST). Creating a functional BST class, inserting elements into the tree, printing the tree in post-order traversal, finding the largest element, and searching for specific numbers were all part of the scope. The purpose of this lab was to gain a hands-on understanding of BST data structures. This lab's major accomplishments were implementing the BST class and achieving the desired output as specified in the sample.

2. Experimental Procedure

I began by creating a Binary Search Tree (BST) class. The class included methods for performing various tree operations such as insertion, searching, finding the largest element, and printing the tree in post-order. The insertion process followed to BST principles, with smaller elements placed to the left and larger elements placed to the right, ensuring that the tree remained balanced.
Then, in the end the BST class's functionality was tested by inserting a series of integer elements into the tree and calling the defined functions.

3. Analysis

BST Construction and Element Insertion: By inserting a series of integer elements, the program successfully constructed a Binary Search Tree (BST). The structure remained consistent, with smaller elements on the left and larger elements on the right.

Post-Order Traversal and Printing: The program correctly implemented the tree's post-order traversal. This traversal order displayed the elements in a specific order (left, right, root), and the program printed the tree in this order correctly.

Finding the Largest Element: The program successfully identified and displayed the largest element in the BST. The algorithm correctly traversed the tree to identify the maximum value, which was appropriately presented as the tree's largest element.

Searching an element in the tree: When an element was searched for, the program correctly identified whether or not the element was present in the tree and provided output accordingly. If the element was present, it was displayed; otherwise, a "not found" message was displayed. This demonstrated that the search functionality had been successfully implemented.

Screenshot of the output in the end of the report.

4. Encountered Problems

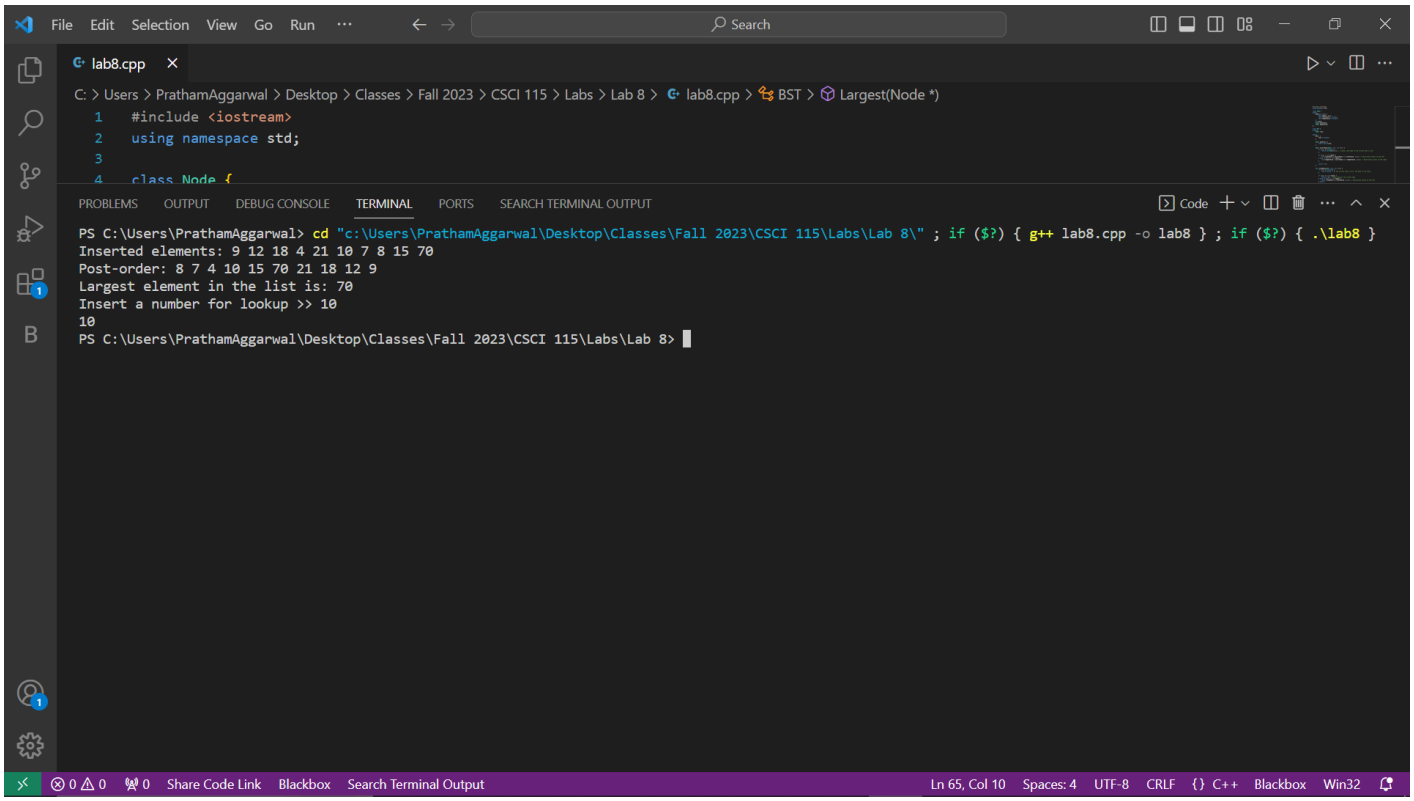
While working on this code, mostly I ran into a number of syntax-related issues, particularly when converting pseudo code into actual implementation. I figured them out with the help of some online resources and tutors.

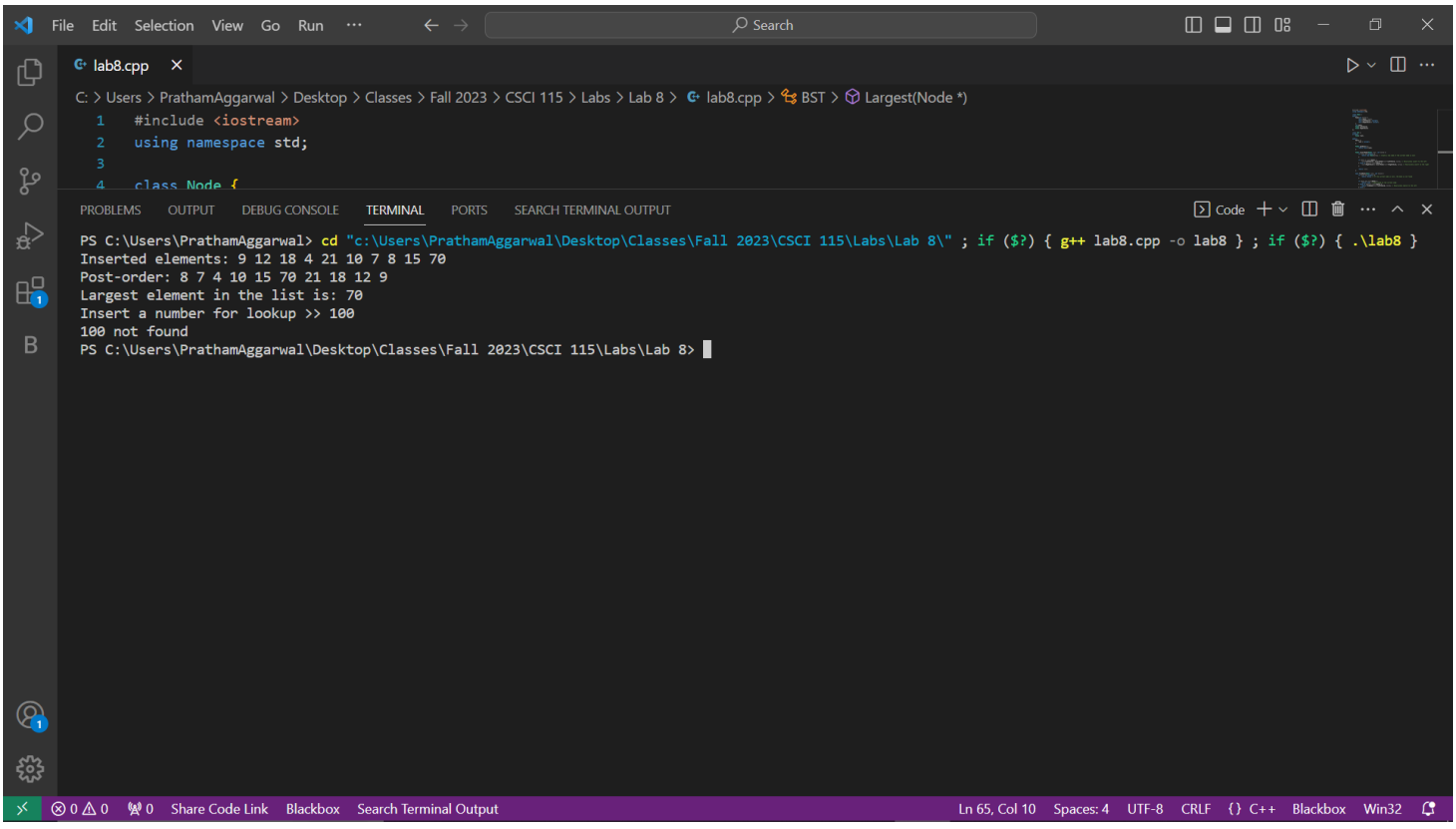
5. Conclusions

In conclusion, this lab provided important hands-on experience with the implementation of a Binary Search Tree (BST) data structure. I gained a thorough understanding of constructing and operating on BSTs through the successful execution of this experiment, including element insertion, post-order traversal, finding the largest element, and efficient search operations. In practice, implementing the pseudo code to see how the code works taught me more about how things work in BST.

6. References

N/A


File Edit Selection View Go Run ···
Search
lab8.cpp
C: > Users > PrathamAggarwal > Desktop > Classes > Fall 2023 > CSCI 115 > Labs > Lab 8 > lab8.cpp > BST > Largest(Node *)
1 #include <iostream>
2 using namespace std;
3
4 class Node {
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH TERMINAL OUTPUT
Code
PS C:\Users\PrathamAggarwal> cd "c:\Users\PrathamAggarwal\Desktop\Classes\Fall 2023\CSCI 115\Labs\Lab 8\" ; if ($?) { g++ lab8.cpp -o lab8 } ; if ($?) { .\lab8 }
Inserted elements: 9 12 18 4 21 10 7 8 15 70
Post-order: 8 7 4 10 15 70 21 18 12 9
Largest element in the list is: 70
Insert a number for lookup >> 10
10
PS C:\Users\PrathamAggarwal\Desktop\Classes\Fall 2023\CSCI 115\Labs\Lab 8>
Share Code Link Blackbox Search Terminal Output
Ln 65, Col 10 Spaces: 4 UTF-8 CRLF {} C++ Blackbox Win32


File Edit Selection View Go Run ···
Search
lab8.cpp
C: > Users > PrathamAggarwal > Desktop > Classes > Fall 2023 > CSCI 115 > Labs > Lab 8 > lab8.cpp > BST > Largest(Node *)
1 #include <iostream>
2 using namespace std;
3
4 class Node {
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH TERMINAL OUTPUT
Code
PS C:\Users\PrathamAggarwal> cd "c:\Users\PrathamAggarwal\Desktop\Classes\Fall 2023\CSCI 115\Labs\Lab 8\" ; if ($?) { g++ lab8.cpp -o lab8 } ; if ($?) { .\lab8 }
Inserted elements: 9 12 18 4 21 10 7 8 15 70
Post-order: 8 7 4 10 15 70 21 18 12 9
Largest element in the list is: 70
Insert a number for lookup >> 100
100 not found
PS C:\Users\PrathamAggarwal\Desktop\Classes\Fall 2023\CSCI 115\Labs\Lab 8>
Share Code Link Blackbox Search Terminal Output
Ln 65, Col 10 Spaces: 4 UTF-8 CRLF {} C++ Blackbox Win32