**Week 7 - Assignment: Implement a Binary Tree Search**

Patrick Swafford

TIM-6110: Programming Languages and Algorithms

Jeevan D’Souza

12 Feb 2023

Table of Contents

BST.cs ............................................................................................................................................ 3

Node.cs ........................................................................................................................................... 9

Program.cs ...................................................................................................................................... 9

Works Cited .................................................................................................................................. 11

BST.cs

The BST class has an interface consisting of two constructors, an insert method, a delete method, a search method, an inorder method, a preorder method, and a postorder method, and each of these methods have private methods in which the actual algorithms are implemented. It should be noted that (Binary Search Tree, 2023) was used as a resource when implementing this class.

The first constructor takes no arguments and simply sets the head Node to null. The second constructor takes a key, creates a new node, and sets that new node as the head node. The second constructor ended up not being used.

Text

Description automatically generated

The public insert interface takes a key as an argument. The interface then passes the head node and key to the private insert implementation method.

The private method is a recursive method. The base case is whether or not the current node is null or not. If the current node is null, the program has recursed its way down to the bottom of the tree. At this point, a new node is created with the key that was passed as a parameter, the counter which tracks the number of times this specific key has been encountered is incremented, and the new node is returned.

If the current node is not null, the String.Compare() method is called on the key and on the current node’s key and the result is stored in a variable for readability. There are three possibilities for this test variable: The test is less than zero, the test is greater than zero, or the test is equal to zero. If the test is less than zero, the key is less than the current node’s key and the current node’s left child is set to the result of the insert function when the left child and the key is passed as arguments. Same concept with test being greater than zero but right instead of left. If the test is equal to zero, this means that the key is the same as the current node’s key. The current node’s counter is incremented, and the current node is returned.

Text

Description automatically generated

As with the insert interface, the delete interface takes a single parameter, the key, and calls a private method which implements the algorithm.

The private delete method takes two parameters, the current node and the key to be deleted. As before, the base case is whether the current node is null or not. If the current node is null, the program has traversed down the tree and the key to be deleted is not in the key. Null is returned.

As with the insert method, a variable stores the result of calling the String.Compare() method on the key to be deleted and the key of the current node, and depending on that variable, the left or right child is set to the result of the delete method called using the left or right child and the key to be deleted as parameters.

If the test variable is not less than or greater than 0, the key to be deleted is equal to the key of the current node, meaning the current node must be deleted. If at least one of the current node’s children is null, this is straightforward. If one child is null, we return the other child. If however neither child is null, this will not work. If we return the left child, the right child would be orphaned, and vice versa. To get around this an auxiliary get\_min() method fetches the minimum child on the current node’s right side which will be referred to as the min Node. The current node will steal the identity of the min node and the delete method deletes the min node instead.

A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated

The search interface and implementation methods are very similar to the insert and delete methods. The interface takes the key as a parameter, calls the private method, and returns a Boolean corresponding to if the key was found or not.

The private method works the same way as the other private methods: by recursing down to the bottom of the tree, and returning the correct node or null depending on if the key was found or not.

Text

Description automatically generated

The inorder, preorder, and postorder methods each print the tree to console. The inorder method prints the tree in alphabetical order, the preorder prints first the root, then the left subtree, then the right subtree, and the postorder prints the left subtree, the right subtree, then the root.

Text

Description automatically generated

Node.cs

There’s not much to say about the Node class. The class has four class members: the left child, the right child, a counter, and the key, as well as getters and setters for each. There are two constructor methods. The first constructor method takes no parameters and initializes everything to null and the second constructor method takes the key as a parameter, initializes the key according to the passed value, and initializes everything else to null. Finally, there’s a method to increment the counter.

Program.cs

Program.cs drives the BST demonstration and contains the Main method. The main method starts by getting the filename from the user and then instantiates a new BST object. Next it reads in each line of the file into a String array called “lines”. A foreach loop loops each line in lines, splits the lines into words, and another foreach loop adds each word to the tree. After all words have been added to the tree, the inorder, preorder, and postorder methods are all called. Should the user fail to pass in a valid filename, or anything at all, the errors are caught and an error message is written to stderr.

Background pattern

Description automatically generated with medium confidence

Text

Description automatically generated

Works Cited

*Binary Search Tree.* (2023). Retrieved from Geeks for Geeks: https://www.geeksforgeeks.org/binary-search-tree-data-structure/