Jarrod Helmers

CS-300

Dr. Webb

3 April, 2024

Module Five Assignment

The provided code implements a binary search tree (BST) data structure in C++, with methods for insertion, removal, search, and traversal of the tree nodes. The tree holds bids with unique identifiers, titles, funds, and amounts. It is designed to facilitate efficient search, insertion, and deletion operations. Bids are loaded from a CSV file using the CSVparser.hpp library.

I faced an error when developing due to unresolved symbols in the CSVparser.hpp header file. I resolved it by correctly setting the include path and linking necessary files. Debugging the binary search tree (BST) implementation required stepping through the code and examining variables. Explicit comments throughout the code helped with understanding and debugging.

Pseudocode:

Structure Bid:

String bidId

String title

String fund

Double amount

Structure Node:

Bid bid

Node\* left

Node\* right

Class BinarySearchTree:

Node\* root

Constructor BinarySearchTree():

Initialize root to nullptr

Method Insert(Bid bid):

If tree is empty:

Create a new node with bid and assign it to root

Else:

Recursively insert the bid into the appropriate subtree

Method Remove(string bidId):

Recursively remove the node with the given bidId from the tree

Method Search(string bidId):

Start from the root

While the current node is not nullptr:

If current node's bidId matches the given bidId:

Return the bid associated with the node

Else if given bidId is less than current node's bidId:

Move to the left subtree

Else:

Move to the right subtree

Return an empty bid if bidId is not found

Method InOrder():

Recursively traverse the tree in order, starting from the root

Method inOrder(Node\* node):

[Recursive implementation to traverse the tree in order]