The code implements a Binary Search Tree (BST) to efficiently store, search for, and manage bids extracted from a CSV file. Each bid includes essential information such as a unique `bidId`, `title`, `fund`, and `amount`. The BST plays a key role in facilitating efficient insertions, deletions, and lookups, by utilizing the properties of a binary tree, where each node’s left child contains smaller values and the correct child contains larger ones. This design enables rapid retrieval of bid information. It supports various tree traversal methods, including in-order, pre-order, and post-order, allowing for the display of bids in different sequences.

During the development process, several challenges were addressed, particularly concerning memory management and data input handling. Dynamic memory allocation for tree nodes required careful oversight to ensure proper deallocation and avoid memory leaks, although the destructor could be enhanced for more thorough cleanup. Additionally, parsing the CSV file presented challenges due to inconsistent or missing data, which could lead to runtime errors. However, this issue was mitigated by implementing a `try-catch` block to handle parsing exceptions gracefully, reassuring the robustness of the system. Another challenge involved converting monetary values in the CSV, which often contained dollar signs and commas. This was resolved by removing these undesired characters prior to converting the values to doubles, ensuring accurate handling of the data.

Pseudocode

Start Program

The program begins by setting up some basic variables and checking the command-line input to get the path of the CSV file and an optional specific bid ID to search for.

Define Structures

Bid: Each bid holds a unique bidId, a title, the fund associated with it, and an amount.

Node: Each node in the tree holds a Bid and pointers to two other nodes: one to the left (for smaller values) and one to the right (for larger values).

Define the Binary Search Tree (BST) Operations

The Binary Search Tree (BST) manages the nodes where each node contains a Bid. The tree helps us store bids in an ordered way, allowing for fast searching, inserting, and deleting.

Insert: When a new bid is added, the tree finds the correct place for the new bid by comparing it to existing bids. Smaller bids go to the left, and larger ones go to the right.

Search: To find a bid by its ID, the tree searches through the nodes, starting at the root. It compares the search ID to the current node’s ID and decides whether to move left or right, repeating this process until it finds the bid or reaches the end.

Remove: When deleting a bid, the tree searches for the node, and depending on how many children the node has (none, one, or two), it rearranges the tree to maintain its structure.

Traversal: The tree can be traversed in three ways to display the bids:

In-Order: Visit the left subtree, then the current node, then the right subtree.

Pre-Order: Visit the current node, then the left subtree, and finally the right subtree.

Post-Order: Visit the left subtree, then the right subtree, and finally the current node.

Load Bids from CSV File

The program opens the CSV file and reads each line of data, extracting the bidId, title, fund, and amount for each bid.

For each bid, it converts the amount (which is a string with a dollar sign) into a number and inserts the bid into the BST.

User Interaction

The program shows a menu with options:

Load Bids: Load bids from the CSV file into the tree.

Display All Bids: Show all the bids in the tree using one of the traversal methods.

Find a Bid: Search for a specific bid by its ID and display its details if found.

Remove a Bid: Delete a bid from the tree by its ID.

Exit: End the program.

Perform Operations

Based on the user’s choice, the program performs one of the operations:

Load the bids from the CSV file into the tree and display how long the process took.

Display all bids by traversing the tree in the selected order (in-order, pre-order, or post-order).

Search for a bid by its bidId and show the result.

Remove a bid from the tree by its bidId.

End Program

The program finishes by printing a goodbye message and exits.