Fisal Ikhmayes

CS 300

02/02/2024

Module 4 code reflection

Code Reflection:

The code uses a hash table with chaining to manage bid information read from a CSV file. The program uses bid insertion, removal, and search operations, incorporating a basic menu. During development, I addressed issues such as incomplete constructor initialization and corrected memory management in the destructor. There was a confusion in accessing the Node structure in the Print All method, which I fixed by adjusting the code to use the address of the node at the specified index in the vector. Additionally, I simplified the logic in the Insert method for better readability. Overall, the code now serves as a structured foundation for managing bids in a hash table, with improved memory handling and corrected access patterns.

Pseudocode or Flowchart:

constructor Hash Table

table size = Size

Nodes resize

Calculate hash value using modulo operator

Return key % table size

HashTable Insert

Create key for the given bid

Retrieve node using key

Previous node = nodes at key

If previous Nodes == null

Nodes = key position

Else

assigne old node key to UNIT\_MAX,

set to key,

set old node to bid

old node next to null pointer

else

while previous node -> next is not null

previousNode = previousNode->next

HashTable Print All

For loop iterate through each bucket

Nodes = node.at(i)

While node is not null

Print key, bidID, title, amount, and fund

Move to next node

Node = node-> next

HashTable Remove

Erase nodes begin and key using nodes.erase

HashTable Search

Nodes at key

If node is not null and node is not equal to uint max and node.bid compare == 0

Return node bid

If node is null and node key = uint max

Return bid

Looking for a match

While node doesn’t equal null

If node key is not equal to uint max and node bid compare == 0

Return node bid.

Node = node next.