Austin Frey

Professor Ling

CS-300

November 13, 2022

LinkedList Activity Code Reflection

The code for the LinkedList activity demonstrates proficiency with Linked List implementation. The program starts with a simple user interface in which users can load different bids from a separate CSV file. Then, there are multiple options for the user to interact with the program:

* Enter a bid
* Load bids from a separate file
* Display all bids
* Search for a specific bid
* Remove a specific bid

I did not have trouble developing most of the functionality of the program. The only aspect of the code I experienced issues with was loading in the bids. In addition, I consistently received an error message when executing the search function stating a memory error at a respective allocated memory location. I spent hours combing forums and stepping through the code to track down the bug. Eventually, I realized that when I changed the default case for the switch statement in the main file, I entered the wrong file name. The aforementioned bug stands as a reminder of the importance of careful coding.

LinkedList Pseudocode

Function LinkedList

Pass in: None

Set head and tail to null

Pass out: None

End function

Function Append

Pass in: The bid to append

Create temporary node

Set temp’s bid to the bid to append and set temp’s next pointer to null

IF head is null

Set head and tail to temp

ELSE

Set tail’s next pointer to temp and set tail to temp’s next pointer

END IF

Increment size of LinkedList

Pass out: None

End function

Function Prepend

Pass in: The bid to prepend

Create temporary node

Set temp’s bid to the bid to prepend and set temp’s next pointer to head

Set head’s next pointer to temp

IF head is not null

Set temp’s next pointer to head

Set head to temp

END IF

Increment size of LinkedList

Pass out: None

End function

Function PrintList

Pass in: None

Create currentNode equal to head node

Output a header detailing the format of the output

WHILE currentNode is not null

Output title, amount, and fund

END WHILE

Pass out: None

End function

Function Search

Pass in: bidId to find

Initialize currentNode to head node

WHILE currentNode is not null

IF currentNode’s bid’s bidId matches the bidId being searched

Return currentNode’s bid

END IF

Set currentNode to currentNode’s next node

END WHILE

Return empty bid

Pass out: found bid or empty bid if there’s not a matching bidId

End function

Function Remove

Pass in: bidId for the node to be removed

Call: Search

Set bidToRemove to the bid returned from search

Initialize currentNode and temp node as empty nodes

IF the head node is the node to be removed

Set temp to head

Set head to head’s next pointer

Delete temp

Decrement size of LinkedList

Return

END IF

Set currentNode to head

WHILE the currentNode’s next pointer is not null

IF currentNode’s next node’s bidId matches the bidId of bidToRemove

Set temp to currentNode’s next node

Set currentNode’s next node to temp’s next node

Delete temp

Decrement size of LinkedList

Return

END IF

Set currentNode to currentNode’s next node

END WHILE

Pass out: None

End function