Michael Wood

09/22/2024

SNHU Analysis and Design 2024

Professor: Michael Rissover

3-2 Assignment: Linked Lists

**Purpose of code:**

This assignment involved information extraction from CSV files including bids for auction businesses. Using the LinkedList, virtual LinkedList, Append, Prepend, PrintList, Remove, and Bid Search methods, this assignment aimed to investigate singly linked lists. It was a real-world scenario that put our knowledge to the test. It was quite difficult even though part of the code was already written, and we only had to fix and insert parts of code to make the program to work.

I had to start from scratch many times over because I lost track of where I was and what I already had done. I tried to bite off more than I could chew and basically was choking. I took a step back and re-evaluated the situation and broke the parts that needed to be completed into smaller issues. Placed code within the program and then run and debug. I think I spent more time chasing errors than coding this time around. I found that it was minor mistakes such as forgetting signs {}; in most cases and sometimes placed in the wrong place creating my headaches. I learned to always double check that I have initialized my housekeeping variables correctly.

**Pseudocode:**

* LinkedList method: Initialize housekeeping variables
* SET head equal to null
* SET tail equal to null
* virtual ~LinkedList method:
* CREATE new node starting at head
* CREATE temp node
* WHILE current node is not null
* SET temp node to current
* SET current node to next node
* DELETE temp node
* Append method: Append a new bid to the end of the list
* CREATE new node
* IF head is null
* SET head and tail equal to new node
* ELSE
* SET tail equal to new node
* INCREASE sizePrepend method: Prepend a new bid to the start of the list
* Create new node
* IF head is not null
* New node points to head as the next nodeSET current node to point beyond next node
* DELETE temp node
* DECREASE size
* RETURN
* SET current node to next node
* Search method: Search for the specified bidId
* IF head is not null && head bidId is equal to input bidId
* SET head equal to next
* DECREASE size
* RETURN head bid
* CREATE new node starting at head
* WHILE current node is not null
* IF current bidId is equal to input bidId
* RETURN current bid
* Otherwise SET current node to next node
* RETURN bid