Brian Engel

Module Three Linked List

The purpose of this assignment was to create and manipulate a singularly linked list with an append method, a prepend method, a print method, a remove method, and a search method. Since we are using a singularly linked list, we have to make sure we stay on a node before the one we actually want to modify, so we use current->next quite a bit. If we were using a doubly linked list, we would be able to use the previous node to go backward, but that is not available in the list we are using.

Append pseudocode

Return void method append(parameter Bid bid)

Node pointer newNode = new Node(bid)

If head is null

head = newNode

tail = newNode

Else

tail->next = newNode

tail = newNode

Increment size

Prepend pseudocode

Return void method prepend(parameter Bid bid)

Node pointer newNode = new Node(bid)

If head is not null

newNode = head

Head = newNode

Increment size

Print pseudocode

Return void method Print (no parameters)

Node pointer current = head

While current is not null

output current bid information

current = current->next

Remove Pseudocode

Return void method Remove (parameter string bidId)

If head node’s bidId = bidId

head = head->next

decrease list size

return

Node pointer current = head

Initialize node pointer temp

While current->next is not null

if current->next->bid’s bidId = bidId

temp = current->next

current->next = temp->next

delete temp

decrease list size

return

Search Pseudocode

Return Bid method Search(parameter string bidId)

If head node’s bidId = bidId

return head

Node pointer current = head

While current->next is not null

if current bidId = bidId

return current

else

current = current->next

Create a empty bid

Return the empty bid