Linked Lists

By: Alex Roberts

The purpose of this assignment was to create some linked lists that the user can use to make a list of items and be able to rearrange, insert, and remove those items as needed. I incorporated housekeeping logic which basically just assigned names to the nodes that will be used so I can incorporate them into the code. Then made several linked lists throughout the code that made these lists changeable for the user. Honestly this was a pretty easy assignment and I enjoyed working on it. I didn’t really have any problems other than simple syntax issues which were easily fixed.

Pseudocode:

Initialize housekeeping logic into the construct

Assign head to null.

Assign tail to null.

Append to a linked list:

Create a new node, assign input to this new node.

IF the head of the list is empty, assign the new node to the head and tail.

ELSE make the current tail point to the new node, make the new node the new tail.

Prepend to a linked list:

Create a new node, assign input to this new node.

IF the head of the list is empty, assign the new node to the head and tail.

ELSE make the new node point to the head, make the new node the head.

Print a linked list:

Start at the head.

WHILE the current element exists, output the attributes of the element.

Increment the current element to the next pointed element in the list.

Remove from a linked list:

IF the selected item is the head node, remove the head.

Check if the element is the key, if it is, assign it to the temp value.

If it is not, check the next value.

WHILE the element in the list exists and does not equal the key.

Delete the temp value.

Search a linked list:

Assign a new node to the head.

WHILE the element exists.

IF the element equals the search key, return the element.

Increment to the next element.

If no element matches the search key, return null.