Sorting Documentation

This program is for a templated doubly linked list that uses smart pointers.

It has the potential to display all values in the list, insert data to the end of the list, locate given data inside the list and delete data from the list.

make sure to #include “DoubleLL.h”.

**Node Class Attributes**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Definition** |
| data | genericType | Holds node data |
| nextNode | shared\_ptr<Node<genericType>> | Holds memory location of next Node |
| previousNode | shared\_ptr<Node<genericType>> | Holds memory location of previous Node |

**Node Class Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Return type** | **Parameter(s)** | **Definition** |
| Node() +2 overloads | Constructor | -  + genericType data  + shared\_ptr nextNode, shared\_ptr previousNode | Used during the construction of object |
| ~Node() | Deconstructor | - | Used during deletion of object |
| get\_Data() | genericType |  | Return data in Node |
| set\_Data() | void | genericType Data | Set data in node |
| get\_Next() | genericType |  | Return pointer to next node |
| set\_Next() | Void | Shared\_ptr<Node> | Set pointer to next node |
| get\_Prev() | genericType |  | Return pointer to previous node |
| set\_Prev() | void | Shared\_ptr<Node> | Set pointer to previous node |

**Doubly Linked List Class Attributes**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Definition** |
| head | shared\_ptr<Node<genericType>> | Holds memory location of head sentinel node |
| tail | shared\_ptr<Node<genericType>> | Holds memory location of tail sentinel node |

**Doubly Linked List Class Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Return type** | **Parameter(s)** | **Definition** |
| DoublyLinkedList() | Constructor | - | Used in construction of the list. Creates a head and tail sentinel node and sets pointers to each other. |
| Find() | Shared\_ptr  <Node<genericType>> | genericType dataToFind | Find a node that contains the data we send as a parameter and return its pointer |
| insertNode() | Void | genericType dataToFind | Dynamically create an object and set its data to the data passed in our parameter |
| deleteNode | Void | genericType dataToFind | Search for the data we are trying using the find function. |
| bubbleSort | Void | - | Bubble sort compares the current element to the next element. If the current is larger than the next it will swap the two values  If will then loop over the list to ensure that the data is sorted. |
| selectionSort | Void | - | Selection sort sets the first element to the minimum in the linked list then sets a new threshhold for searching.This will set each new element to the next lowest value until the Entire list has been searched |
| swap | Void | Shared\_ptr <Node<genericType>> current  Shared\_ptr <Node<genericType>> next | Swap the data in the two nodes |
| displayList() | Void | - | Output to console |