Singly Linked List

**Node Class:**

The **Node** class is a basic building block of the linked list. It holds an integer value and a pointer to the next node in the list.

**LinkedList Class:**

Constructor **LinkedList()**

* Creates an empty linked list by initializing **head**, **tail**, and **length**.

Function **isEmpty()**

* Checks if the linked list is empty.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **insertWhileEmpty(Node\* newNode)**

* Inserts a new node when the linked list is empty.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **deleteOnlyElement()**

* Deletes the only element from the linked list.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **findIndex(int index)**

* Finds the node at a given index.
* Time Complexity: O(index)
* Space Complexity: O(1)

Function **findValue(int value)**

* Finds the index of a given value in the linked list.
* Time Complexity: O(n)
* Space Complexity: O(1)

Function **swap(Node\* first, Node\* second)**

* Swaps the values of two nodes.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **pushBack(int value)**

* Adds an element to the end of the linked list.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **pushFront(int value)**

* Adds an element to the beginning of the linked list.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **insertAfterIndex(int index, int value)**

* Inserts an element after a specified index.
* Time Complexity: O(index)
* Space Complexity: O(1)

Function **insertBeforeIndex(int index, int value)**

* Inserts an element before a specified index.
* Time Complexity: O(index)
* Space Complexity: O(1)

Function **popFront()**

* Removes the first element from the linked list.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **popBack()**

* Removes the last element from the linked list.
* Time Complexity: O(n)
* Space Complexity: O(1)

Function **deleteNode(int index)**

* Deletes the element at the specified index.
* Time Complexity: O(index)
* Space Complexity: O(1)

Function **deleteLinkedList()**

* Deletes the entire linked list.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **display()**

* Displays the elements of the linked list.
* Time Complexity: O(n)
* Space Complexity: O(1)

Function **reverse()**

* Reverses the linked list.
* Time Complexity: O(n)
* Space Complexity: O(1)

Function **search(int value)**

* Searches for an element and returns its index.
* Time Complexity: O(n)
* Space Complexity: O(1)

Function **update(int index, int value)**

* Updates the value of a node at a given index.
* Time Complexity: O(index)
* Space Complexity: O(1)

Function **sort()**

* Sorts the linked list using Bubble Sort.
* Time Complexity: O(n^2)
* Space Complexity: O(1)

Function **headNode()**

* Returns the value of the head node.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **tailNode()**

* Returns the value of the tail node.
* Time Complexity: O(1)
* Space Complexity: O(1)

Function **linkedListLength()**

* Returns the length of the linked list.
* Time Complexity: O(1)
* Space Complexity: O(1)

Destructor **~LinkedList()**

* Frees memory by deleting all nodes in the linked list.
* Time Complexity: O(n)
* Space Complexity: O(1)

**Main Function:**

The **main** function demonstrates the usage of various linked list operations, including insertion, deletion, updating, searching, and sorting.Top of Form