Brandon Slusser

March 9, 2023

Node Class: a class that represents a node in a linked list.

LinkedList Class: represents a linked list.

PUBLIC MEMBER VARIABLES

string fullStudentName; - stores the full name of a student.

int RedID; - stores the Red ID number of a student.

int value; - stores the value of a node.

Node* next; - a pointer that points to the next node in the linked list.

CONSTRUCTORS

The Node class has a constructor that takes an integer and initializes the value of the node with that integer value. This constructor also sets the pointer of the node to nullptr.

The LinkedList class also has a constructor that takes an integer value and initializes a linked list with a new node that is that integer value. It also sets the head and the tail pointers of the linked list to point to the new node, and sets the length of the linked list to 1.

The LinkedList class has a destructor that deallocates all of the nodes of the linked list.

OTHER PUBLIC MEMBER FUNCTIONS

printList(): prints all of the integer values that are stored in the nodes of the linked list. Precondition: The linked list is not empty. Postcondition: The linked list remains the same and is printed.

printListName(): prints all of the characters that are stored in the nodes of the linked list. Precondition: The linked list is not empty and it only contains characters. Postcondition: the linked list remains the same and is printed.

getHead(): prints the value that is stored in the first node of the linked list. Precondition: The linked list is not empty. Postcondition: The linked list remains the same and the head is printed.

getTail(): prints the value that is stored in the last node of the linked list. Precondition: The linked list is not empty. Postcondition: The linked list remains the same and the tail is printed.

getLength(): prints the length of the linked list. Precondition: None. Postcondition: The linked list remains the same and the length is printed.

Node* get(int index): User inputs the index that they would like printed and it returns a pointer to the node at the specified index. Precondition: The linked list is not empty and is within the bounds of the linked list. Postcondition: The linked list remains the same.

bool set(int index, int value): User enters the index they would like to modify and the value they would like that index to be, this function then sets the value of the node at the specified index to the value that the user entered. Precondition: The index input is not nullptr. Postcondition: The value of the node at the specified index is set to the value that was entered by the user.

void append(int value): User inputs the value they would like added to the end of the linked list and this creates a new node with the given value and adds it to the end of the linked list. Precondition: None. Postcondition: A new node with the given value is added to the end of the linked list.

void prepend(int value): User inputs the value they would like added to the beginning of the linked list and this creates a new node with the given value and adds it to the beginning of the linked list. Precondition: None. Postcondition: A new node with the value inputted by the user is added to the beginning of the linked list.

bool insert(int index, int value): User inputs the index they would like the node to be at and enters the value that they want the index to hold and this creates a new node with the given value and inserts it at the given index in the linked list. Precondition: The index is within the bounds of the linked list. Postcondition: A new node with the given value is inserted at the specified index of the linked list.

void deleteFirst(): deletes the first node of the linked list. Precondition: The linked list is not empty. Postcondition: The first node of the linked list is deleted.

void deleteLast(): deletes the last node of the linked list. Precondition: the linked list is not empty. Postcondition: the last node of the linked list is deleted.

void deleteNode(int index): User inputs the index that they want deleted from the linked list and the function deletes the node at the inputted index. Precondition: The index is within the bounds of the linked list. Postcondition: The node at the index inputted by the user is deleted.

void reverseLinkedList():reverses the order of the linked list. Precondition: The linked list is not empty. Postcondition: The linked list is in reverse order.

void selectionSort(): Sorts the linked list using the selection sort method. Precondition:None Postcondition: The linked list is sorted.

void insertionSort(): Sorts the linked list using the insertion sort method. Precondition:None Postcondition: The linked list is sorted.