Maira Soomro | Section 1003 | Assignment 6

This program will implement a Linked List class and will be able to perform different functions such as insert, delete a node, search for a node and copy a list onto a another one.

|  |
| --- |
| Linked List |
| -count: int  -\*first: <type>  -\*last: <type> |
| +LinkedList()  +initializeList(): void  +isEmptyList (): bool  +print(): void  +length(): void  +destroyList(): void  +search(Type&): bool  +insert(Type&): void  +deleteNote(Type&): void  +isSortedList(): bool  +sortList(): void  +LinkedList(LinkedList<Type>&)  +~LinkedList  +copyList(LinkedList<Type>&): void |

Linked List Class Specification:

const LinkedList<Type>& operator=(const LinkedList<Type>&);

//Overload the assignment operator.

void initializeList();

//Initialize the list to an empty state.

//Postcondition: first = NULL, last = NULL, count = 0;

bool isEmptyList() const;

//Function to determine whether the list is empty.

//Postcondition: Returns true if the list is empty,

// otherwise it returns false.

void print() const;

//Function to output the data contained in each node.

//Postcondition: none

int length() const;

//Function to return the number of nodes in the list.

//Postcondition: The value of count is returned.

void destroyList();

//Function to delete all the nodes from the list.

//Postcondition: first = NULL, last = NULL, count = 0;

bool search(const Type& searchItem) const;

//Function to determine whether searchItem is in the list.

//Postcondition: Returns true if searchItem is in the

// list, otherwise returns false.

void insert(const Type& newItem);

//Function to insert newItem at the end of the list.

//Postcondition: first points to the new list, newItem

// is inserted at the end of the list,

// last points to the last node in the

// list, and count is incremented by 1.

void deleteNode(const Type& deleteItem);

//Function to delete deleteItem from the list.

//Postcondition: If found, the node containing

// deleteItem is deleted from the list.

// first points to the first node, last

// points to the last node of the updated

// list, and count is decremented by 1.

bool isSortedList() const;

//Function to determine whether the list is sorted.

//Postcondition: Returns true if the list is sorted in

// ascending order, otherwise returns false.

void sortList();

//Function to sort the list in ascending order.

//Postcondition: List is sorted in ascending order.

LinkedList();

//default constructor

//Initializes the list to an empty state.

//Postcondition: first = NULL, last = NULL, count = 0;

LinkedList(const LinkedList<Type>& otherList);

//copy constructor

~LinkedList();

//destructor

//Deletes all the nodes from the list.

//Postcondition: The list object is destroyed.

void copyList(const LinkedList<Type>& otherList);

//Function to make a copy of otherList.

//Postcondition: A copy of otherList is created and

// assigned to this list.