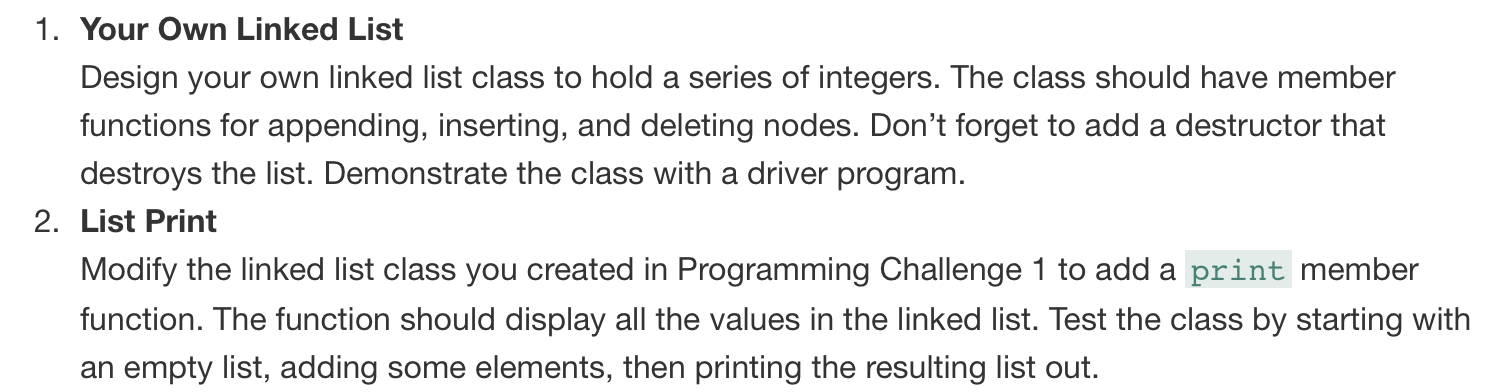
COMSC-165 Lab-5 Due: July-22 11:59 PM



#ifndef LINKEDLIST\_H

#define LINKEDLIST\_H

#include "ListNode.h"

class LinkedList{

public:

// pointer to linked list header

ListNode \*head;

// constructor

LinkedList();

// deconstructor

~LinkedList();

// linked list operations

void appendNode(int);

void insertNode(int);

void deleteNode(int);

void display() const;

};

#endif // LINKEDLIST\_H

#ifndef LISTNODE\_H

#define LISTNODE\_H

struct ListNode{

int value;

struct ListNode \*next;

};

#endif // LISTNODE\_H

#include <iostream>

#include "ListNode.h"

#include "LinkedList.h"

using namespace std;

LinkedList::LinkedList(){

head = NULL;

}

LinkedList::~LinkedList(){

ListNode \*nodePointer;

ListNode \*nextNode;

nodePointer = head;

while(nodePointer != NULL){

nextNode = nodePointer->next;

delete nodePointer;

nodePointer = nextNode;

}

}

void LinkedList::appendNode(int x){

ListNode \*newNode;

ListNode \*nodePointer;

newNode = new ListNode;

newNode->value = x;

newNode->next = NULL;

if (!head){

head = newNode;

}else{

nodePointer = head;

while(nodePointer->next){

nodePointer = nodePointer->next;

nodePointer->next = newNode;

}

}

display();

}

void LinkedList::insertNode(int x){

ListNode \*newNode;

ListNode \*nodePointer;

ListNode \*previousNode = NULL;

newNode = new ListNode;

newNode->value = x;

newNode->next = NULL;

if (!head){

head = newNode;

}else{

nodePointer = head;

previousNode = NULL;

while(nodePointer != NULL && nodePointer->value < x){

previousNode = nodePointer;

nodePointer = nodePointer->next;

}

if (previousNode == NULL){

head = newNode;

newNode->next = nodePointer;

}else{

previousNode->next = newNode;

newNode->next = nodePointer;

}

}

display();

}

void LinkedList::deleteNode(int x){

ListNode \*nodePointer;

ListNode \*previousNode;

if (!head){

cout << "\nFailed to delete, list is empty.";

return;

}

if (head->value == x){

nodePointer = head->next;

delete head;

head = nodePointer;

}else{

nodePointer = head;

while(nodePointer != NULL && nodePointer->value != x){

previousNode = nodePointer;

nodePointer = nodePointer->next;

}

}

if (nodePointer){

previousNode->next = nodePointer->next;

delete nodePointer;

}else{

cout << "\nFailed to delete, " << x << " is not found in the list.";

}

display();

}

void LinkedList::display() const{

ListNode \*nodePointer;

if (!head){

cout << "\nThe list is empty." << endl;

return;

}else{

nodePointer = head;

cout << "\nThe elements in the list are:\n";

while(nodePointer){

cout << nodePointer->value << "->";

nodePointer = nodePointer->next;

}

cout << "Null" << endl <<endl;

}

}

#include <iostream>

#include "ListNode.h"

#include "LinkedList.h"

using namespace std;

int main(){

int choice;

int num;

LinkedList llo;

do{

cout << "\tMENU";

cout << "\n1 - Append node";

cout << "\n2 - Insert node";

cout << "\n3 - Delete node";

cout << "\n4 - Exit";

cout << "\n\nEnter a menu option: ";

cin >> choice;

switch(choice){

case 1:{

cout << "Enter an integer: ";

cin >> num;

llo.appendNode(num);

break;

}

case 2:{

cout << "Enter an integer: ";

cin >> num;

llo.insertNode(num);

break;

}

case 3:{

cout << "Enter an integer: ";

cin >> num;

llo.deleteNode(num);

break;

}

}

}while(choice != 4);

return 0;

}

