ЗВІТ

про виконання лабораторної роботи № 12.3(rec)

«Опрацювання лінійного двонаправленого списку»

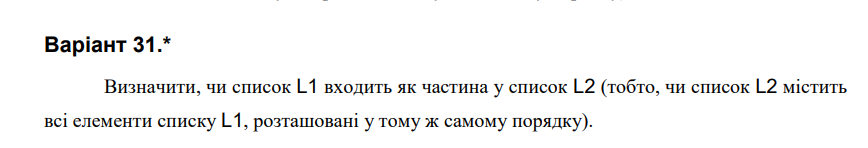
з дисципліни

«Алгоритмізація та програмування»

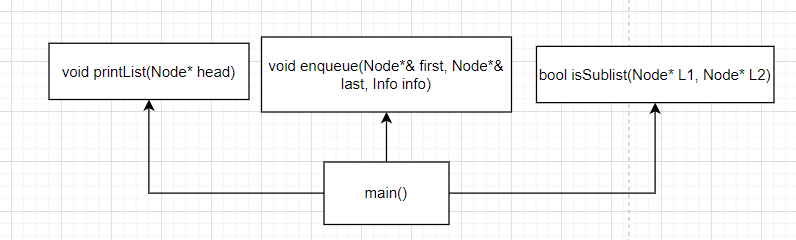
студента групи: KH-2226Б

**Кав’яка Назарія Івановича**

Мета: Навчитися опрацьовувати лінійні двонаправлені списки



Структурна схема програми:



Текст програми:

#include<iostream>

#include<iomanip>

using namespace std;

typedef int Info;

struct Node

{

Node\* next,

\* prev;

Info info;

};

void enqueue(Node\*& first, Node\*& last, Info info)

{

Node\* tmp = new Node;

tmp->info = info;

tmp->next = NULL;

if (last != NULL)

last->next = tmp;

tmp->prev = last;

last = tmp;

if (first == NULL)

first = tmp;

}

bool isSublist(Node\* L1, Node\* L2)

{

// Якщо L1 стає NULL, це означає, що весь L1 був знайдений у L2

if (L1 == NULL)

return true;

// Якщо L2 стає NULL, а L1 ще не досягнув кінця, то L1 не є підсписком L2

if (L2 == NULL)

return false;

// Якщо поточні елементи L1 та L2 збігаються, перевіряємо наступні елементи

if (L1->info == L2->info)

return isSublist(L1->next, L2->next);

// Якщо поточні елементи L1 та L2 не збігаються, перевіряємо наступний елемент L2

return isSublist(L1, L2->next);

}

void printList(Node\* head) {

if (head == NULL) {

cout << endl;

return;

}

cout << head->info << " ";

printList(head->next);

}

int main()

{

Node\* FirstL1 = NULL; // Список L1

Node\* LastL1 = NULL;

Node\* FirstL2 = NULL; // Список L2

Node\* LastL2 = NULL;

// Заповнення списків L1 та L2

enqueue(FirstL1, LastL1, 1);

enqueue(FirstL1, LastL1, 2);

enqueue(FirstL1, LastL1, 3);

enqueue(FirstL2, LastL2, 1);

enqueue(FirstL2, LastL2, 2);

enqueue(FirstL2, LastL2, 3);

enqueue(FirstL2, LastL2, 4);

enqueue(FirstL2, LastL2, 5);

//Друкування списків L1 і L2

cout << "List 1: ";

printList(FirstL1);

cout << "List 2: ";

printList(FirstL2);

// Перевірка, чи є L1 частиною L2

if (isSublist(FirstL1, FirstL2))

cout << "L1 is a part of L2" << endl;

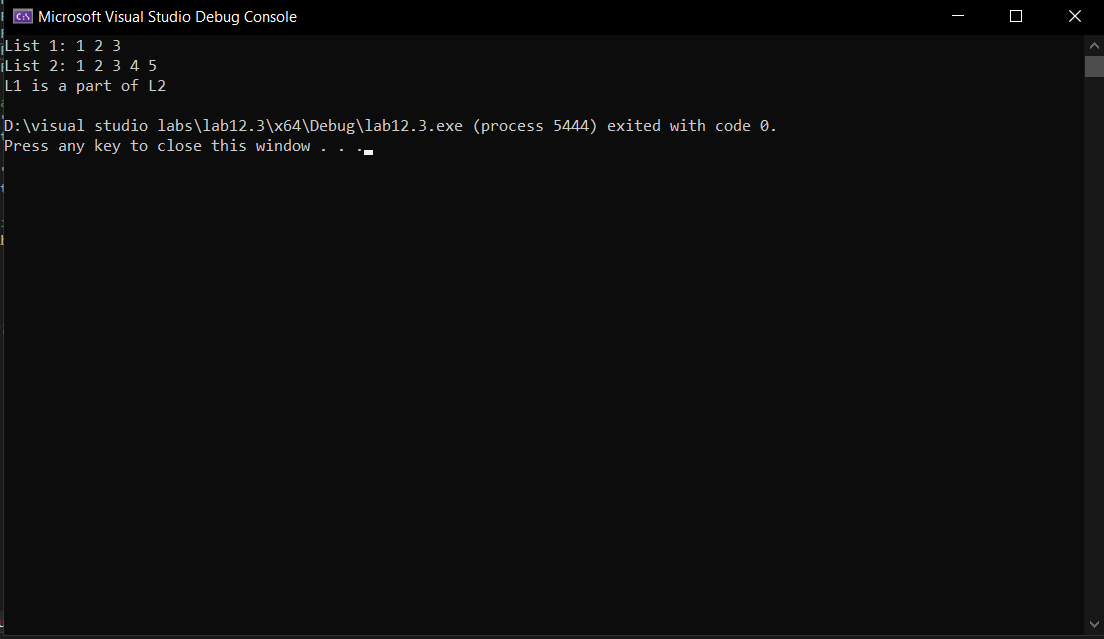
else

cout << "L1 is not a part of L2" << endl;

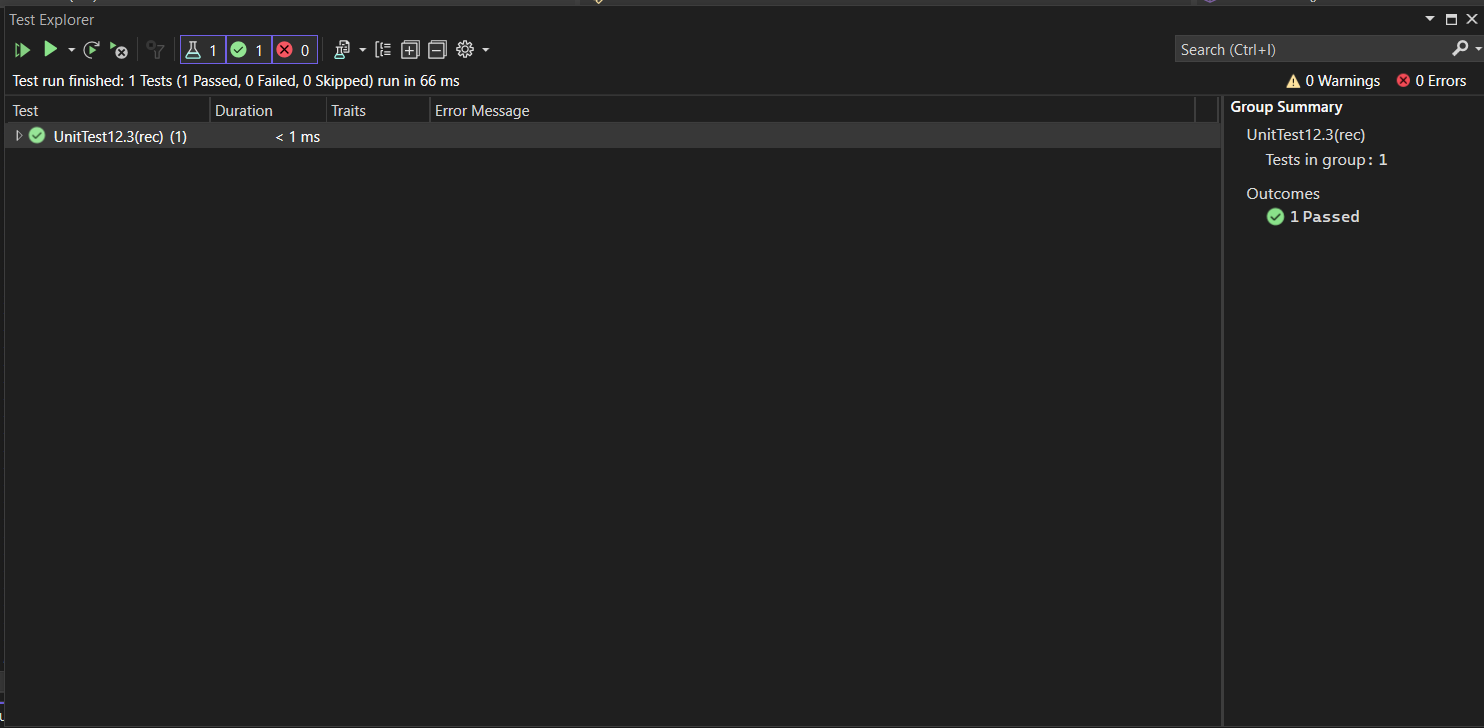
return 0;

}

Результат:



Результат юніт-тесту:



Текст програми юніт-тесту:

#include "pch.h"

#include "CppUnitTest.h"

#include "../Source.cpp"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace UnitTest123rec

{

TEST\_CLASS(UnitTest123rec)

{

public:

TEST\_METHOD(TestMethod1)

{

// Створюємо списки L1 і L2

Node\* FirstL1 = NULL;

Node\* LastL1 = NULL;

Node\* FirstL2 = NULL;

Node\* LastL2 = NULL;

// Заповнюємо списки L1 і L2 згідно макету

enqueue(FirstL1, LastL1, 22);

enqueue(FirstL1, LastL1, 44);

enqueue(FirstL2, LastL2, 14);

enqueue(FirstL2, LastL2, 423);

enqueue(FirstL2, LastL2, 22);

enqueue(FirstL2, LastL2, 44);

enqueue(FirstL2, LastL2, 55);

// Викликаємо функцію isSublist і перевіряємо результат

bool result = isSublist(FirstL1, FirstL2);

Assert::IsTrue(result, L"L1 is a part of L2");

// Звільняємо пам'ять списків L1 і L2

Node\* curr = LastL1;

while (curr != nullptr)

{

Node\* temp = curr;

curr = curr->next;

delete temp;

}

curr = LastL2;

while (curr != nullptr)

{

Node\* temp = curr;

curr = curr->next;

delete temp;

}

}

};

}

Посилання на git-репозиторій:

https://github.com/sDiablo/Lab12.3-rec-

Висновок: На цій лабораторній роботі я навчився опрацьовувати лінійні двонаправлені списки