Об’єктно орієнтоване програмування

**Лабораторна робота №5**

# Двозв’язний список

# Теоретичні відомості

Двозв’язний список являє собою розширення ідеї однозв'язного списку і також складається з елементів даних, кожен з яких містить посилання як на наступний, так і на попередній елементи. На рис. 5.1 показана організація посилань в двозв’язного списку.

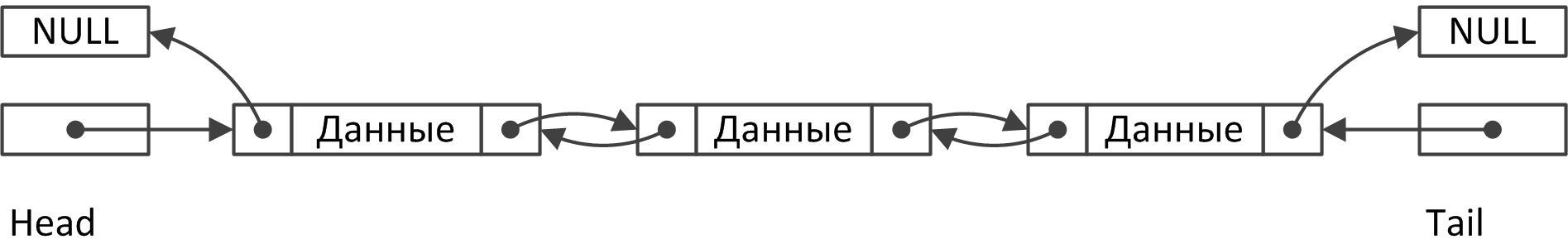


Рисунок 5.1 – Графічне зображення двозв’язаного списку

Наявність двох посилань замість одного надає кілька переваг. Найбільш важливе з них полягає в тому, що переміщення по списку можливо в обох напрямках. Це спрощує роботу зі списком, зокрема, вставку і видалення. Крім цього, користувач може переглядати список в будь-якому напрямку. Ще одна перевага має значення тільки при деяких збоях. Оскільки весь список можна пройти не лише за прямими, але і по зворотним посиланням, то в разі проблем з одним із посилань, цілісність списку можна відновити за іншим посиланням.

**Приклад**. Написати програму, в якій реалізований клас, що описує двозв’язний список для зберігання цілих чисел. Користувач має можливість додавати, редагувати та видаляти елементи списку (рис. 5.2).

Лістинг 5.1 – Файл linked2list.h

#ifndef LINKED2LIST\_H

#define LINKED2LIST\_H

#include <QTableWidget>

struct Elem // Елемент даних

**{**

int data**;** // Дані

Elem **\*** next**,** **\*** prev**;** // Адреса наступного та попередього елементу у списку

**};**

class Linked2list

**{**

Elem **\***Head**,** **\***Tail**;**// Голова, хвіст

int Count**;** // Кількість елементів списку

QTableWidget **\***qtable**;** // QTableWidget для відображення

public**:**

Linked2list**();** // Конструктор

**~**Linked2list**();** // Деструктор

int GetCount**();** // Кількість елементів у списку

void DelAll**();** // Видалення всього списку

void Del**(**int pos **=** 0**);** // Видалення елементу з списку

void AddHead**(**int n**);** // Дадавання елементу в початок списоку

void setQTable**(**QTableWidget **\***qtable**);**// Для встановлення посилання на QTableWidget

void PrintToQTable**();**// Вивід списку у QTableWidget

void SetValue**(**int index**,** int data**);** // Завдання значення i-го елементу

**};**

#endif // LINKED2LIST\_H

Лістинг 5.2 – Файл linked2list.cpp

#include "linked2list.h"

Linked2list**::**Linked2list**()**

**{**

// спочатку список порожній

Head **=** Tail **=** **nullptr;**

Count **=** 0**;**

**}**

Linked2list**::~**Linked2list**()**

**{**

qtable **=** **nullptr;**

DelAll**();** // Видалення всіх елементів списку

**}**

void Linked2list**::**AddHead**(**int n**)**

**{**

Elem **\*** temp **=** **new** Elem**;** // Створення нового елементу

temp**->**prev **=** **nullptr;** // Попередього немає

temp**->**data **=** n**;** // Заповнюємо дані

temp**->**next **=** Head**;** // Наступний - голова у минулому

**if** **(**Head **!=** **nullptr)** // Чи є елементи

Head**->**prev **=** temp**;**

// Якщо елемент перший, то він одночасно голова та хвіст

**if** **(**Count **==** 0**)**

Head **=** Tail **=** temp**;**

**else**

Head **=** temp**;** // інакше новий елемент - голова

Count**++;**

PrintToQTable**();**

**}**

void Linked2list**::**Del**(**int pos**)**

**{**

pos**++;**

**if(**pos **<** 1 **||** pos **>** Count**)**

**return;**

int i **=** 1**;** // Лічильник

Elem **\*** Del **=** Head**;**

**while(**i **<** pos**)**

**{**

Del **=** Del**->**next**;** // Знаходимо елемент, що видаляється

i**++;**

**}**

// Знаходимо елемент,

// попередній до видаляємого

Elem **\*** PrevDel **=** Del**->**prev**;**

// Елемент, наступний за видаляємим

Elem \* AfterDel = Del->next;

if (PrevDel != nullptr && Count != 1) // Якщо видаляємо не голову

PrevDel->next = AfterDel;

if(AfterDel != nullptr && Count != 1) // Якщо видаляємо не хвіст

AfterDel->prev = PrevDel;

if(pos == 1) // Удаляются крайние?

Head = AfterDel;

if(pos == Count)

Tail = PrevDel;

delete Del; // Освобождение памяти

Count--;

PrintToQTable();

}

void Linked2list::DelAll()

{

// Поки є елементи - видаляємо по одному з головы

while(Count != 0)

Del(0);

}

int Linked2list::GetCount()

{

return Count;

}

void Linked2list::SetValue(int index, int data)

{

if (index > GetCount())

return;

Elem \* temp = Head;

int i = 0;

while (temp != nullptr && i < index)

{

i++;

temp = temp->next;

}

temp->data = data;

PrintToQTable();

}

void Linked2list::setQTable(QTableWidget\* qtable)

{

this->qtable = qtable;

}

void Linked2list::PrintToQTable()

{

if (qtable == nullptr)

return;

Elem \* temp = Head; // Запам'ятовуємо адресу головного ел-ту

qtable->setRowCount(Count);

int i = 0;

while(temp != nullptr) // Поки є елементи

{

qtable->setItem(i, 0, new QTableWidgetItem(QString::number(temp->data)));

temp = temp->next; // Переходимо до наступного ел-ту списку

i++;

}

}

Лістинг 5.3 – Файл mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "linked2list.h"

#include <QInputDialog>

Linked2list list2**;**

MainWindow**::**MainWindow**(**QWidget **\***parent**)** **:**

QMainWindow**(**parent**),**

ui**(new** Ui**::**MainWindow**)**

**{**

ui**->**setupUi**(this);**

// Наші налаштування

list2**.**setQTable**(**ui**->**tableWidget**);**

ui**->**tableWidget**->**setColumnCount**(**1**);**

ui**->**tableWidget**->**setHorizontalHeaderItem**(**0**,** **new** QTableWidgetItem**(**"Дані"**));**

**}**

MainWindow**::~**MainWindow**()**

**{**

**delete** ui**;**

**}**

void MainWindow**::**on\_pushButton\_clicked**()**

**{**

list2**.**AddHead**(**ui**->**lineEdit**->**text**().**toInt**());**

**}**

void MainWindow**::**on\_pushButton\_2\_clicked**()**

**{**

list2**.**Del**();**

**}**

void MainWindow**::**on\_tableWidget\_cellDoubleClicked**(**int row**,** int column**)**

**{**

QString value **=** ui**->**tableWidget**->**item**(**row**,** column**)->**text**();**

bool ok**;**

QString text **=** QInputDialog**::**getText**(this,** "Значення"**,** "Введіть нове значення"**,** QLineEdit**::**Normal**,** value**,** **&**ok**);**

**if** **(**ok **&&** **!**text**.**isEmpty**())**

list2**.**SetValue**(**row**,** text**.**toInt**());**

**}**

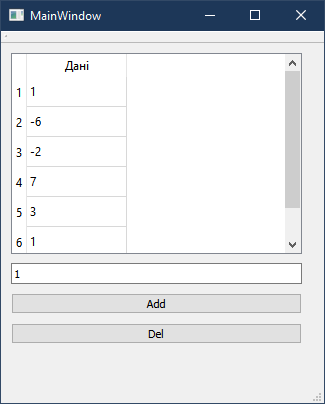


Рисунок 5.2 – Результат роботи програми

**Лабораторна робота №5 (3-й семестр)**

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення суми додатніх елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення мінімального значення елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення максимального значення елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення суми елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення добутку елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення мінімального додатного значення елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення максимального від'ємного значення елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення мінімального за модулем значення елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.

Написати програму, в якій

* 1. Реалізовано клас, що описує двозв’язний список для зберігання дійсних чисел;
  2. Користувач має можливість додавати (в голову, в середину і в хвіст), видаляти і змінювати довільний елемент списку;
  3. Вміст списку зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 4;
  4. У класі реалізований метод для обчислення максимального по модулю значення елементів списку;
  5. Представити алгоритм видалення довільного елемента списку у вигляді блок-схеми і словесного опису.