Об’єктно орієнтоване програмування

**Лабораторна робота №3**

# Черга

# Теоретичні відомості

Зв’язний список – це структура даних, що складається з вузлів, кожен з яких має як власне дані, так і одне або два посилання («зв'язки») на наступний і / або попередній вузол списку. Важлива перевага перед масивом є структурна гнучкість: порядок елементів зв'язкового списку може не збігатися з порядком розташування елементів даних в пам'яті комп'ютера, а порядок обходу списку завжди явно задається його внутрішніми зв'язками.

Однозв'язний список складається з покажчика на перший елемент списку (голову, head) і самих даних, причому кожен елемент списку містить вказівник на наступний. Останній елемент списку містить вказівник на NULL (рис. 1).



Рисунок 4.1 – Графічне зображення зв’язаного списку

**Приклад.** Написати програму, в якій реалізований клас, що описує пов'язаний список для зберігання цілих чисел. Користувач має можливість додавати, редагує-вать і видаляти елементи (рис. 4).

Лістинг 3.1 – Файл linkedlist.h

#ifndef LINKEDLIST\_H

#define LINKEDLIST\_H

#include <QTableWidget>

struct Element // Елемент даних

{

int data; // Дані

Element \* Next; // Адреса наступного елементу у списку

};

class LinkedList

{

Element \* Head; // Вказівник на голову списка

int Count; // Кількість елементів списку

QTableWidget \*qtable; // QTableWidget для відображення

public:

LinkedList(); // Конструктор

~LinkedList(); // Деструктор

void Add(int data);// Дадавання елементу в список

void Del();// Видалення елементу з списку

void DelAll(); // Видалення всього списку

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

int GetCount(); // Кількість елементів у списку

public:

};

#endif // LINKEDLIST\_H

Лістинг 3.2 – Файл linkedlist.cpp

#include "linkedlist.h"

LinkedList::LinkedList()

{

Head = nullptr; // спочатку список порожній

Count = 0;

}

LinkedList::~LinkedList()

{

qtable = nullptr;

DelAll(); // Видалення всіх елементів списку

}

int LinkedList::GetCount()

{

return Count; // Кількість елементів у списку

}

void LinkedList::Add(int data)

{

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

temp->data = data; // заповнення даними

temp->Next = Head; // наступний елемент - це теперешня голова

Head = temp; // новий елемент стає головним елементом

Count++; // Збільшуємо кількість ел-тів

PrintToQTable();

}

void LinkedList::Del()

{

if (Head == nullptr)

return;

Element \* temp = Head; // запам'ятовуємо адресу головного ел-та

Head = Head->Next; // перекидаємо голову на наступний ел-т

delete temp; // видаляжмо минулий головний ел-т

Count--; // зменшуємо кількість на 1

PrintToQTable(); // друкуємо список

}

void LinkedList::DelAll()

{

while (Head != nullptr) // Пока еще есть элементы

Del(); // Удаляем элементы по одному

}

void LinkedList::setQTable(QTableWidget\* qtable)

{

this->qtable = qtable;

}

void LinkedList::PrintToQTable()

{

if (qtable == nullptr)

return;

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

qtable->setRowCount(Count);

int i = 0;

while(temp != nullptr) // Поки є ел-ти

{

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

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

i++;

}

}

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

{

if (index > GetCount())

return;

Element \* temp = Head;

int i = 0;

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

{

i++;

temp = temp->Next;

}

temp->data = data;

PrintToQTable();

}

Лістинг 3.3 – Файл mainwindow.cpp

#include <QInputDialog>

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "linkedlist.h"

LinkedList list;

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

// Наші налаштування

list.setQTable(ui->tableWidget);

ui->tableWidget->setColumnCount(1);

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

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_pushButton\_clicked()

{

list.Add(ui->lineEdit->text().toInt());

}

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

{

list.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())

list.SetValue(row, text.toInt());

}

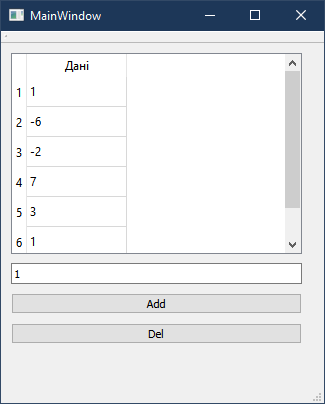


Рисунок 4.2 – Результат роботи програми

**Лабораторная работа №4 (3-й семестр)**

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть его актуальное состояние. Пользователю также выводится результат выполнения п. 3;
  4. В классе реализован метод для вычисления суммы положительных элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть его актуальное состояние. Пользователю также выводится результат выполнения п. 4;
  4. В классе реализован метод для вычисления минимального значения элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть ее актуальное состояние. Пользователю также выводится результат выполнения п. 4;
  4. В классе реализован метод для вычисления максимального значения элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть его актуальное состояние. Пользователю также выводится результат выполнения п. 4;
  4. В классе реализован метод для вычисления суммы элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть ее актуальное состояние. Пользователю также выводится результат выполнения п. 4;
  4. В классе реализован метод для вычисления произведения элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть его актуальное состояние. Пользователю также выводится результат выполнения п. 4;
  4. В классе реализован метод для вычисления минимального положительного значения элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть ее актуальное состояние. Пользователю также выводится результат выполнения п. 4;
  4. В классе реализован метод для вычисления максимального отрицательного значения элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть его актуальное состояние. Пользователю также выводится результат выполнения п. 4;
  4. В классе реализован метод для вычисления минимального по модулю значения элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.

Написать программу, в которой

* 1. Реализован класс, описывающий односвязный список для хранения вещественных чисел;
  2. Пользователь имеет возможность добавлять, а так же удалять и изменять произвольный элемент списка;
  3. Содержимое списка связывается с некоторым компонентом таким образом, чтобы пользователь мог видеть его актуальное состояние. Пользователю также выводится результат выполнения п. 4;
  4. В классе реализован метод для вычисления максимального по модулю значения элементов списка;
  5. Представить алгоритм удаления произвольного элемента списка в виде блок-схемы и словесного описания.