Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа**

**"** **Информационные динамические структуры "**

Выполнил:

студент группы ИВТ-23-1б

Машарипов.А

Проверила:

доцент кафедры ИТАС

Яруллин.Д.В

2024

1. **Постановка задачи.**

Написать программу, в которой создаются динамические структуры и выполнить их обработку в соответствии со своим вариантом.

1. **Вариант задания**

Записи в линейном списке содержат ключевое поле типа int. Сформировать однонаправленный список. Удалить из него элемент с заданным номером, добавить элемент с заданным номером;

1. **Текст программы.**

#include <iostream>

#include <fstream>

struct Node {

    int data;

    Node\* next;

};

class LinkedList {

private:

    Node\* head;

public:

    LinkedList() : head(nullptr) {}

    // Функция для добавления элемента в конец списка

    void add(int value) {

        Node\* newNode = new Node{value, nullptr};

        if (!head) {

            head = newNode;

        } else {

            Node\* temp = head;

            while (temp->next) {

                temp = temp->next;

            }

            temp->next = newNode;

        }

        print();

    }

    // Функция для удаления элемента по номеру

    void remove(int position) {

        if (!head) {

            std::cout << "Список пустой, удаление невозможно." << std::endl;

            return;

        }

        Node\* temp = head;

        if (position == 0) { // Удаление первого элемента

            head = temp->next;

            delete temp;

        } else {

            for (int i = 0; temp != nullptr && i < position - 1; i++) {

                temp = temp->next;

            }

            if (temp == nullptr || temp->next == nullptr) {

                std::cout << "Элемент с данным номером не найден." << std::endl;

                return;

            }

            Node\* next = temp->next->next;

            delete temp->next; // Удаляем элемент

            temp->next = next;  // Соединяем с последующим элементом

        }

        print();

    }

    // Функция для печати списка

    void print() {

        if (!head) {

            std::cout << "Список пустой." << std::endl;

            return;

        }

        Node\* temp = head;

        while (temp) {

            std::cout << temp->data << " ";

            temp = temp->next;

        }

        std::cout << std::endl;

    }

    // Функция для записи списка в файл

    void writeToFile(const std::string& filename) {

        std::ofstream outFile(filename);

        Node\* temp = head;

        while (temp) {

            outFile << temp->data << std::endl;

            temp = temp->next;

        }

        outFile.close();

    }

    // Функция для уничтожения списка

    void destroy() {

        Node\* current = head;

        Node\* nextNode;

        while (current != nullptr) {

            nextNode = current->next;

            delete current;

            current = nextNode;

        }

        head = nullptr;

    }

    // Функция для восстановления списка из файла

    void restoreFromFile(const std::string& filename) {

        std::ifstream inFile(filename);

        int value;

        while (inFile >> value) {

            add(value);

        }

        inFile.close();

    }

    ~LinkedList() {

        destroy();

    }

};

int main() {

    LinkedList list;

    // Пример добавления элементов

    list.add(10);

    list.add(20);

    list.add(30);

    // Пример удаления элемента по номеру

    list.remove(1); // Удаление элемента с индексом 1 (второй элемент)

    // Запись в файл

    list.writeToFile("list.txt");

    // Уничтожение списка

    list.destroy();

    list.print(); // Должно вывести "Список пустой."

    // Восстановление списка из файла

    list.restoreFromFile("list.txt");

    list.print(); // Должно вывести элементы из файла

    // Уничтожение списка перед завершением программы

    list.destroy();

    return 0;

}

1. **Результат решения конкретного варианта.**