Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**Отчет по лабораторной работе №11.1**

Дисциплина: «Основы алгоритмизации и программирования»

Тема : "Однонаправленный список"

Выполнил работу:

Студент группы РИС-22-1Б

Батин В. В.

Проверил:

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

**Постановка задачи.**

Создать однонаправленный список, добавить в него k элементов с заданного номера, удалить из него k элементов с заданного номера, вывести список.

**Анализ задачи.**

Структура Node представляет узел списка, который содержит значение ключа (поле) и указатель на следующий узел (поле next).

Класс LinkedList представляет сам список и содержит методы для добавления, удаления и вставки элементов, а также для вывода списка на экран.

Метод add добавляет новый элемент в конец списка. Если список пустой, то новый элемент становится его головой (head), иначе происходит перебор элементов до тех пор, пока не будет найден последний элемент, и новый элемент добавляется после него.

Метод remove удаляет элементы из списка, начиная с заданного номера (start) и до количества элементов (count). Если start больше или равен размеру списка, то метод ничего не делает. Если count равен нулю, то метод также ничего не делает.

Метод insert добавляет новые элементы в список, начиная с заданного номера (start) и до количестваов (count). Если start больше размера списка, то метод ничего не делает.

Метод print выводит элементы списка на экран.

В функции main создается объект класса LinkedList, заполняется 10 элементами с помощью метода add, затем выводится список на экран. Далее происходит удаление 4 элементов, начиная с третьего, и вывод списка на экран. Затем происходит вставка 2 новых элементов, начиная с четвертого, и вывод списка на экран.

**Код.**

#include <iostream>

struct Node {

int key;

Node\* next;

Node(int key) : key(key), next(nullptr) {}

};

class LinkedList {

public:

LinkedList() : head(nullptr), size(0) {}

// Добавление элемента в конец списка

void add(int key) {

Node\* newNode = new Node(key);

if (head == nullptr) {

head = newNode;

}

else {

Node\* curr = head;

while (curr->next != nullptr) {

curr = curr->next;

}

curr->next = newNode;

}

size++;

}

// Удаление элементов из списка, начиная с заданного номера

void remove(int start, int count) {

if (start >= size || count == 0) {

return;

}

Node\* curr = head;

Node\* prev = nullptr;

for (int i = 0; i < start; i++) {

prev = curr;

curr = curr->next;

}

for (int i = 0; i < count && curr != nullptr; i++) {

Node\* next = curr->next;

delete curr;

curr = next;

size--;

}

if (prev == nullptr) {

head = curr;

}

else {

prev->next = curr;

}

}

// Добавление элементов в список, начиная с заданного номера

void insert(int start, int count) {

if (start > size) {

return;

}

Node\* curr = head;

Node\* prev = nullptr;

for (int i = 0; i < start; i++) {

prev = curr;

curr = curr->next;

}

for (int i = 0; i < count; i++) {

Node\* newNode = new Node(rand() % 100);

newNode->next = curr;

if (prev == nullptr) {

head = newNode;

}

else {

prev->next = newNode;

}

prev = newNode;

size++;

}

}

// Вывод элементов списка на экран

void print() const {

Node\* curr = head;

while (curr != nullptr) {

std::cout << curr->key << " ";

curr = curr->next;

}

std::cout << std::endl;

}

private:

Node\* head;

int size;

};

int main() {

LinkedList list;

for (int i = 0; i < 10; i++) {

list.add(i);

}

list.print(); // 0 1 2 3 4 5 6 7 8 9

list.remove(3, 4);

list.print(); // 0 1 2 7 8 9

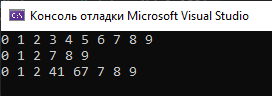
list.insert(3, 2);

list.print(); // 0 1 2 78 52 7 8 9

return 0;

}

**Результат выполнения.**

****