Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы» направление подготовки: 09.03.04 – «Программная инженерия»

**Лабораторная работа №11.**

**«Информационные динамические структуры.**

**Односвязные списки»**

Выполнил студент гр. РИС-24-1б

Чижов Денис Николаевич

Проверил:

Доц. Каф. ИТАС 

Ольга Андреевна Полякова

(оценка) (подпись)

(дата)

г. Пермь, 2025

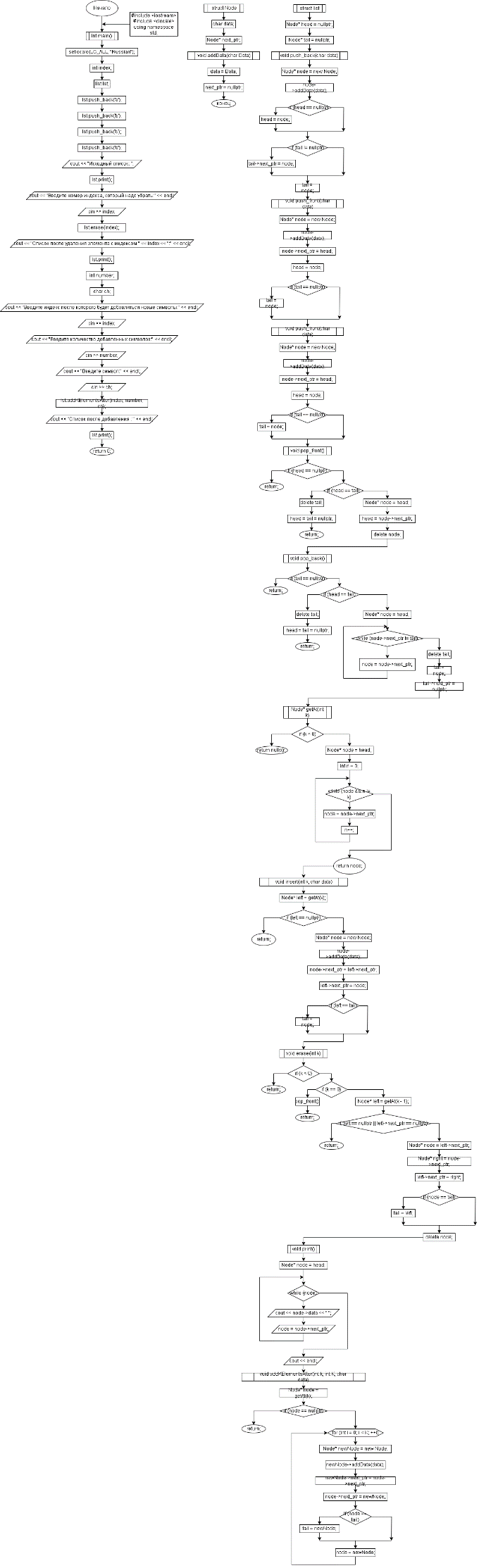
**Задача:**

23.Записи в линейном списке содержат ключевое поле типа \*char(строка символов). Сформировать односвязный список. Удалить элемент с заданным ключом. Добавить K элементов после элемента с заданным ключом.

**Анализ и визуализация:**

Мы формируем односвязный список, в котором записаны наши элементы типа char и затем удаляем элемент с заданным индексом и добавляем K элементов после элемента с выбранным индексом.

**Блок-схема:**



**Код:**

#include <iostream>

#include <clocale>

using namespace std;

struct Node {

char data;

Node\* next\_ptr;

void addData(char Data) {

data = Data;

next\_ptr = nullptr;

}

};

struct list {

Node\* head = nullptr;

Node\* tail = nullptr;

void push\_back(char data) {

Node\* node = new Node;

node->addData(data);

if (head == nullptr) head = node;

if (tail != nullptr) tail->next\_ptr = node;

tail = node;

}

void push\_front(char data) {

Node\* node = new Node;

node->addData(data);

node->next\_ptr = head;

head = node;

if (tail == nullptr) tail = node;

}

void pop\_front() {

if (head == nullptr) return;

if (head == tail) {

delete tail;

head = tail = nullptr;

return;

}

Node\* node = head;

head = node->next\_ptr;

delete node;

}

void pop\_back() {

if (tail == nullptr) return;

if (head == tail) {

delete tail;

head = tail = nullptr;

return;

}

Node\* node = head;

while (node->next\_ptr != tail) {

node = node->next\_ptr;

}

delete tail;

tail = node; // Теперь tail указывает на предпоследний узел

tail->next\_ptr = nullptr;

}

Node\* getAt(int k) {

if (k < 0) return nullptr;

Node\* node = head;

int n = 0;

while (node && n != k) {

node = node->next\_ptr;

n++;

}

return node;

}

void insert(int k, char data) {

Node\* left = getAt(k);

if (left == nullptr) return;

Node\* node = new Node;

node->addData(data);

node->next\_ptr = left->next\_ptr;

left->next\_ptr = node;

if (left == tail) tail = node;

}

void erase(int k) {

if (k < 0) return;

if (k == 0) {

pop\_front();

return;

}

Node\* left = getAt(k - 1);

if (left == nullptr || left->next\_ptr == nullptr) return;

Node\* node = left->next\_ptr;

Node\* right = node->next\_ptr;

left->next\_ptr = right;

if (node == tail) tail = left;

delete node;

}

void print() {

Node\* node = head;

while (node) {

cout << node->data << " ";

node = node->next\_ptr;

}

cout << endl;

}

void addKElementsAfter(int k, int K, char data) {

Node\* node = getAt(k);

if (node == nullptr) return;

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

Node\* newNode = new Node;

newNode->addData(data);

newNode->next\_ptr = node->next\_ptr;

node->next\_ptr = newNode;

if (node == tail) tail = newNode;

node = newNode;

}

}

};

int main(){

setlocale(LC\_ALL, "Russian");

int index;

list lst;

lst.push\_back('a');

lst.push\_back('b');

lst.push\_back('c');

lst.push\_back('d');

cout << "Исходный список: ";

lst.print();

// Удаляем элемент с индексом 2

cout << "Введите номер индекса, который надо убрать:" << endl;

cin >> index;

lst.erase(index);

cout << "Список после удаления элемента с индексом " << index << ":" << endl;

lst.print();

// Добавляем 3 элемента 'x' после элемента с индексом 1

int number;

char ch;

cout << "Введите индекс после которого будет добавляться новые символы:" << endl;

cin >> index;

cout << "Введите количество добавленных символов:" << endl;

cin >> number;

cout << "Введите символ:" << endl;

cin >> ch;

lst.addKElementsAfter(index, number, ch);

cout << "Список после добавления : " << endl;

lst.print();

return 0;

}

**Вывод:**

Программа работает. Задача решена.