Министерство высшего 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 выполненной лабораторной работе «ДВУХСВЯЗНЫЙ СПИСОК»**

Дисциплина: основы алгоритмизации и программирования

Вариант №1

Выполнил:

Студент группы ИВТ

Голубцов Никита

Проверил:

Доцент кафедры ИТАС

Полякова Ольга Андреевна

**Пермь, 2023**

**Постановка задачи**

Реализовать двухсвязный список.

Реализовать удаление элемента по значению, введённому пользователем.

Реализовать добавление элемента, введенного пользователем, в начало списка.

**Анализ задачи**

1. Создаем список. С помощью метода createList() создаем список

2. С помощью методы void addElement() добавляем новый элемент в список перед заданным номером элемента.

3. С помощью метода deleteKElement

**Блок – схема**

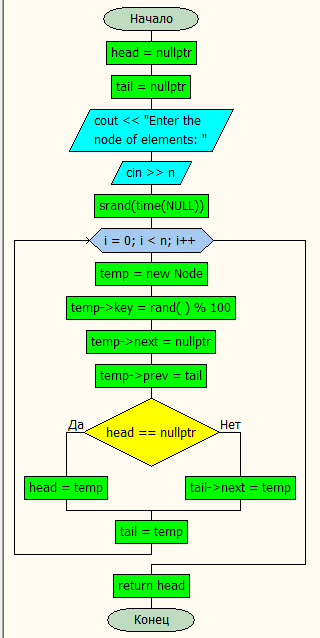
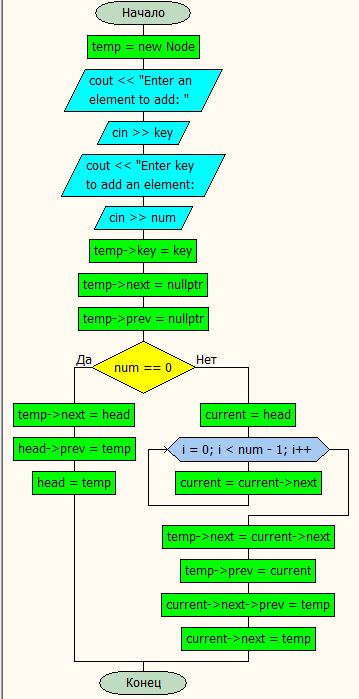


Рис 1,2 (addElement , createList)

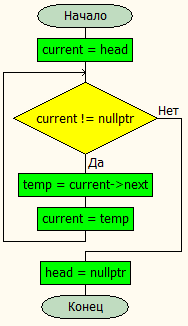
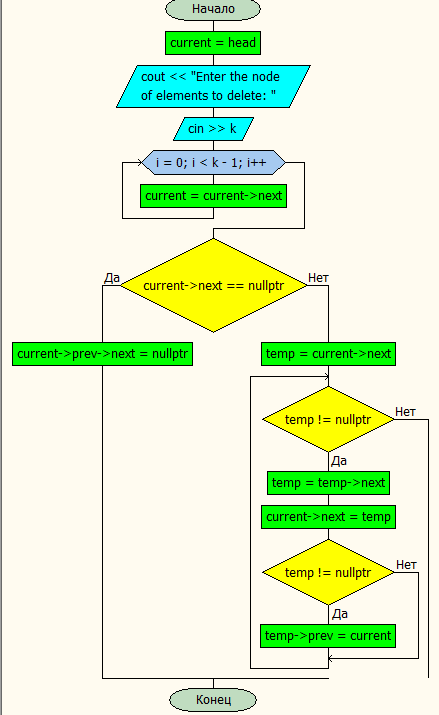


Рис3.4 (deleteElement, destroyList)

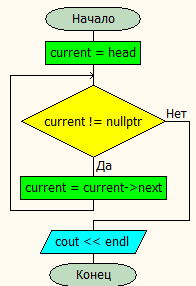
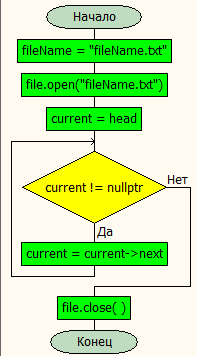
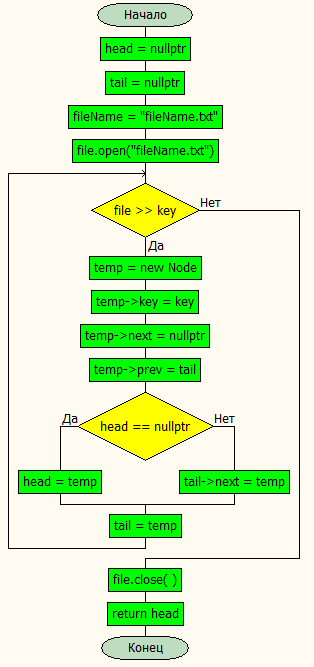
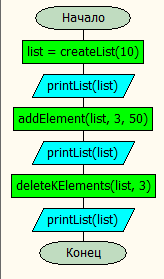


Рис 3,4,5(LoadListFromFile, savelistTpFile, printList)

рис 6(main).

**Код программы**

#include <iostream>

#include <fstream>

#include <ctime>

using namespace std;

// Ñòðóêòóðà çàïèñè ñïèñêà

struct Node {

int key;

Node\* next;

Node\* prev;

};

// Ôóíêöèÿ ñîçäàíèÿ ñïèñêà

Node\* createList(int n) {

Node\* head = nullptr;

Node\* tail = nullptr;

cout << "Enter the node of elements: ";

cin >> n;

srand(time(NULL));

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

Node\* temp = new Node;

temp->key = rand() % 100;

temp->next = nullptr;

temp->prev = tail;

if (head == nullptr) {

head = temp;

}

else {

tail->next = temp;

}

tail = temp;

}

return head;

}

// Ôóíêöèÿ äîáàâëåíèÿ ýëåìåíòà â ñïèñîê

void addElement(Node\*& head, int num, int key) {

Node\* temp = new Node;

cout << "Enter an element to add: ";

cin >> key;

cout << "Enter key to add an element: ";

cin >> num;

temp->key = key;

temp->next = nullptr;

temp->prev = nullptr;

if (num == 0) {

temp->next = head;

head->prev = temp;

head = temp;

}

else {

Node\* current = head;

for (int i = 0; i < num - 1; i++) {

current = current->next;

}

temp->next = current->next;

temp->prev = current;

current->next->prev = temp;

current->next = temp;

}

}

// Ôóíêöèÿ óäàëåíèÿ ýëåìåíòîâ èç êîíöà ñïèñêà

void deleteKElements(Node\*& head, int k) {

Node\* current = head;

cout << "Enter the node of elements to delete: ";

cin >> k;

for (int i = 0; i < k - 1; i++) {

current = current->next;

}

if (current->next == nullptr) {

// Óäàëÿåì ïîñëåäíèé ýëåìåíò

current->prev->next = nullptr;

delete current;

}

else {

// Óäàëÿåì íåñêîëüêî ýëåìåíòîâ èç êîíöà ñïèñêà

Node\* temp = current->next;

while (temp != nullptr) {

temp = temp->next;

delete current->next;

current->next = temp;

if (temp != nullptr) {

temp->prev = current;

}

}

}

}

// Ôóíêöèÿ ïå÷àòè ñïèñêà

void printList(Node\* head) {

Node\* current = head;

while (current != nullptr) {

cout << current->key << " ";

current = current->next;

}

cout << endl;

}

// Ôóíêöèÿ çàïèñè ñïèñêà â ôàéë

void saveListToFile(Node\* head, string fileName) {

ofstream file;

fileName = "fileName.txt";

file.open("fileName.txt");

Node\* current = head;

while (current != nullptr) {

file << current->key << " ";

current = current->next;

}

file.close();

}

// Ôóíêöèÿ âîññòàíîâëåíèÿ ñïèñêà èç ôàéëà

Node\* loadListFromFile(string fileName) {

Node\* head = nullptr;

Node\* tail = nullptr;

ifstream file;

fileName = "fileName.txt";

file.open("fileName.txt");

int key;

while (file >> key) {

Node\* temp = new Node;

temp->key = key;

temp->next = nullptr;

temp->prev = tail;

if (head == nullptr) {

head = temp;

}

else {

tail->next = temp;

}

tail = temp;

}

file.close();

return head;

}

// Ôóíêöèÿ óíè÷òîæåíèÿ ñïèñêà

void destroyList(Node\*& head) {

Node\* current = head;

while (current != nullptr) {

Node\* temp = current->next;

delete current;

current = temp;

}

head = nullptr;

}

int main() {

// Ñîçäàíèå ñïèñêà

Node\* list = createList(10);

// Ïå÷àòü ñïèñêà

printList(list);

// Äîáàâëåíèå ýëåìåíòà â ñïèñîê

addElement(list, 3, 50);

// Ïå÷àòü ñïèñêà

printList(list);

// Óäàëåíèå ýëåìåíòîâ èç êîíöà ñïèñêà

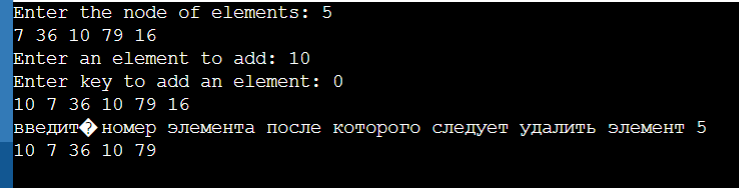
deleteKElements(list, 3);

// Ïå÷àòü ñïèñêà

printList(list);

}

**Результат работы программы**



**Анализ полученных результатов**

На вход подан список из пяти элементов.

Удален элемент со значением 4.

В начало списка добавлен элемент со значением 9999999