Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Пермский национальный исследовательский политехнический университет» (ПНИПУ)

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

**Лабораторная работа № 11.2**

**по дисциплине**

**«Основы алгоритмизации и программирования»**

**(Семестр** 2)

Выполнил студент гр. ИВТ-21-1б

Гребнев Алексей Дмитриевич

Проверил:

Яруллин Денис Владимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

г.Пермь -2022

**Постановка задачи**

1. Сформировать двусвязный список. Тип информационного поля указан в варианте.

2. Распечатать полученную структуру.

3. Выполнить обработку структуры в соответствии с заданием.

4. Распечатать полученный результат.

5. Удалить соответствующую структуру из памяти.

Исходные данные

1. Создаем структуру Nodeс информационным полем char data и с указателем типа структуры на следующий и предыдущий элементы;
2. Создаем список через отдельную функцию, в которую передаем количество элементов, создаем указатель на начало списка и два вспомогательных, задаем первый узел, заходим в цикл и создаем следующие узлы, присваивая в их адресные поля адрес на следующий и предыдущий элемент от каждого;
3. В другой функции мы будем проверять элементы списка условие меньше ли информационное поле нуля если да, то меняем указатели и удаляем элемент, если нет, то переходим на следующий элемент:
4. Выводим список.

#include <iostream>

using namespace std;

struct Node

{

char data;

Node\* ptr\_prev = nullptr;

Node\* ptr\_next = nullptr;

};

struct List

{

Node\* headnode = nullptr;

Node\* tailnode = nullptr;

};

void push(List& list, char& data)

{

Node\* newnode = new Node;

newnode->data = data;

if (list.headnode == nullptr)

{

list.headnode = newnode;

list.tailnode = newnode;

return;

}

Node\* currentnode = list.headnode;

newnode->ptr\_prev = currentnode;

if (currentnode->ptr\_next != nullptr)

{

newnode->ptr\_next = currentnode->ptr\_next;

currentnode->ptr\_next->ptr\_prev = newnode;

}

currentnode->ptr\_next = newnode;

list.tailnode = newnode;

}

void printList(List& list)

{

Node\* currentnode = list.headnode;

while (currentnode != nullptr)

{

cout << currentnode->data << " ";

currentnode = currentnode->ptr\_next;

}

}

void add(List& list, char& value, char& add\_value)

{

Node\* newnode = new Node;

newnode->data = add\_value;

Node\* currentnode;

if (list.headnode == nullptr)

{

list.headnode = newnode;

list.tailnode = newnode;

return;

}

for (currentnode = list.headnode; currentnode->data != value; currentnode = currentnode->ptr\_next);

newnode->ptr\_next = currentnode->ptr\_next;

currentnode->ptr\_next = newnode;

}

int main()

{

setlocale(LC\_ALL, "RUS");

List list;

int n;

int index = 3;

char tmp;

cout << "Введите размер списка: ";

cin >> n;

cout << "Введите элементы двусвязного списка: ";

for (int i = 1; i <= n; i++)

{

cin >> tmp;

push(list, tmp);

}

cout << "Изначальный список" << endl;

printList(list);

cout << endl;

char sl;

cout << "Введите элемент после которого мы должны добавить новый: ";

cin >> sl;

char newsl;

cout << "Введите добавляемый элемент: ";

cin >> newsl;

add(list, sl, newsl);

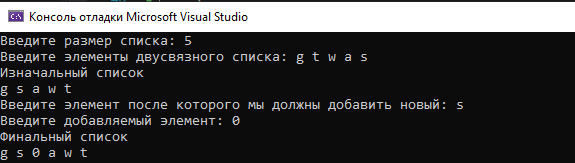
cout << "Финальный список" << endl;

printList(list);

return 0;

}

**Результат**



**Блок-схема**