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

Электротехнический факультет  
Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т  
по лабораторной работе**

Дисциплина: «Основы алгоритмизации и программирования»

Выполнил  
студент группы КЗИ-21-1б  
Полковников Т.Ю.

Проверил  
доцент кафедры ИТАС  
ст. преп. Яруллин Д.В.

Пермь, 2022

## Постановка задачи:

1. Сформировать однонаправленный и двунаправленный списки или стек и очередь. 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.

## Анализ решения задачи:

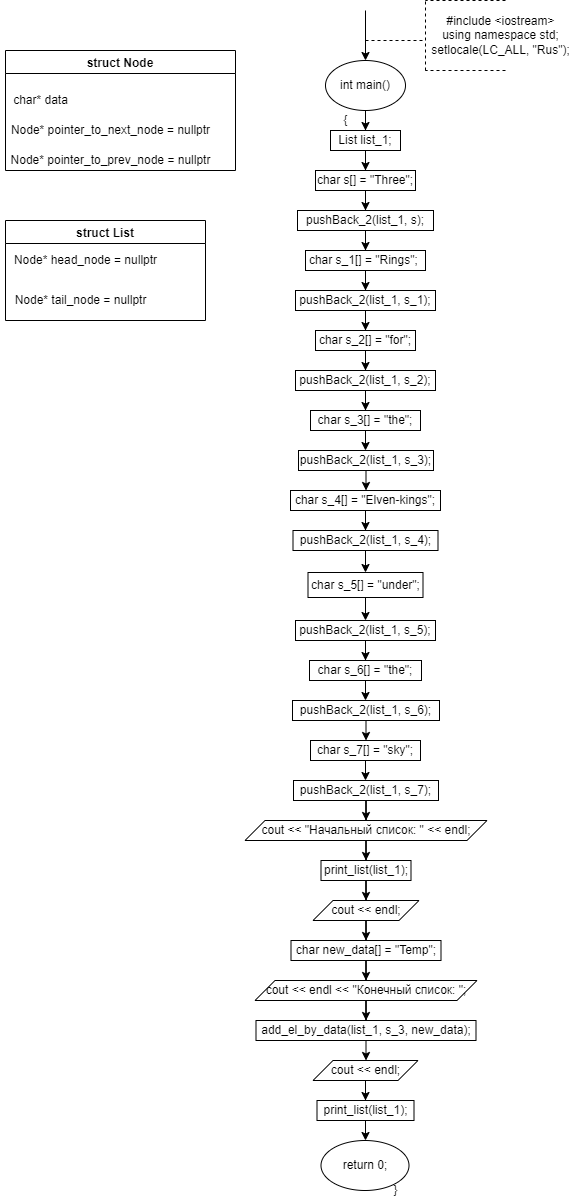
Тип информационного поля char\*. Добавить в список элемент после элемента с заданным информационным полем.

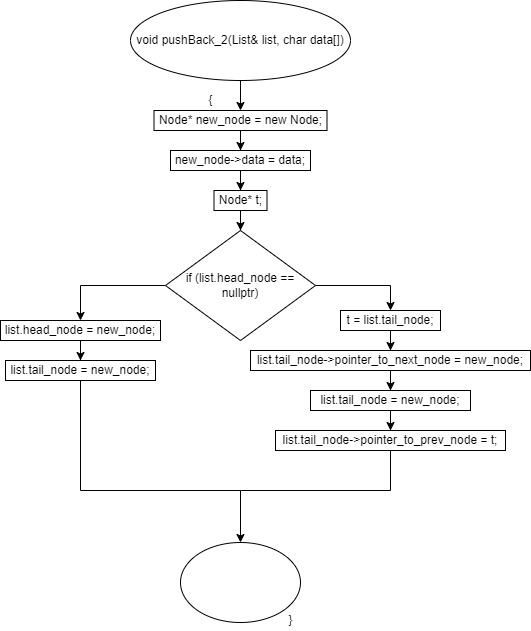
## Описание переменных:

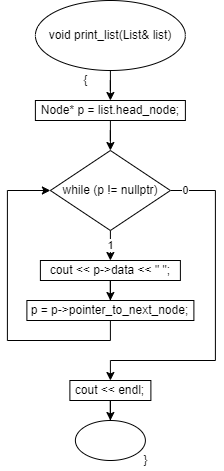
list\_1 – двусвязный список

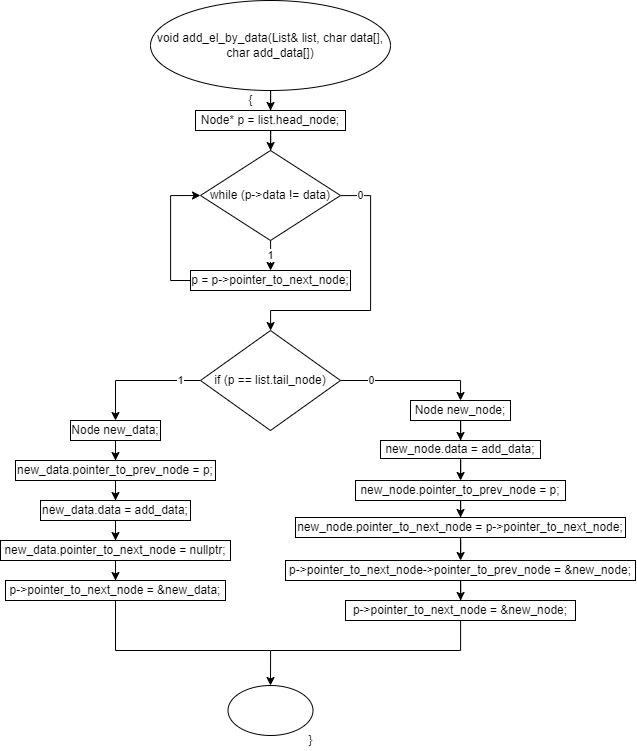
s, s\_1, s\_1,… s\_7 – строки для заполнения списка

**Блок схема**









## Код программы

#include <iostream>

using namespace std;

// узел

struct Node {

char\* data;

Node\* pointer\_to\_next\_node = nullptr;

Node\* pointer\_to\_prev\_node = nullptr;

};

// список

struct List {

Node\* head\_node = nullptr;

Node\* tail\_node = nullptr;

};

// добавление элемента в конец списка

void pushBack\_2(List& list, char data[]) {

Node\* new\_node = new Node;

new\_node->data = data;

Node\* t;

if (list.head\_node == nullptr) {

list.head\_node = new\_node;

list.tail\_node = new\_node;

}

else {

t = list.tail\_node;

list.tail\_node->pointer\_to\_next\_node = new\_node;

list.tail\_node = new\_node;

list.tail\_node->pointer\_to\_prev\_node = t;

}

}

// вывод списка

void print\_list(List& list) {

Node\* p = list.head\_node;

while (p != nullptr) {

cout << p->data << " ";

p = p->pointer\_to\_next\_node;

}

cout << endl;

}

// добавление элемента после элемента с заданным информационным полем

void add\_el\_by\_data(List& list, char data[], char add\_data[]) {

Node\* p = list.head\_node;

while (p->data != data) {

p = p->pointer\_to\_next\_node;

}

// p = элементу с заданным полем

if (p == list.tail\_node) {

Node new\_data;

new\_data.pointer\_to\_prev\_node = p;

new\_data.data = add\_data;

new\_data.pointer\_to\_next\_node = nullptr;

p->pointer\_to\_next\_node = &new\_data;

}

else {

Node new\_node;

new\_node.data = add\_data;

new\_node.pointer\_to\_prev\_node = p;

new\_node.pointer\_to\_next\_node = p->pointer\_to\_next\_node;

p->pointer\_to\_next\_node->pointer\_to\_prev\_node = &new\_node;

p->pointer\_to\_next\_node = &new\_node;

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

// создаём список

List list\_1;

// заполняем поле data узлов списка

char s[] = "Three";

pushBack\_2(list\_1, s);

char s\_1[] = "Rings";

pushBack\_2(list\_1, s\_1);

char s\_2[] = "for";

pushBack\_2(list\_1, s\_2);

char s\_3[] = "the";

pushBack\_2(list\_1, s\_3);

char s\_4[] = "Elven-kings";

pushBack\_2(list\_1, s\_4);

char s\_5[] = "under";

pushBack\_2(list\_1, s\_5);

char s\_6[] = "the";

pushBack\_2(list\_1, s\_6);

char s\_7[] = "sky";

pushBack\_2(list\_1, s\_7);

cout << "Начальный список: " << endl;

print\_list(list\_1);

cout << endl;

char new\_data[] = "Temp";

cout << endl << "Конечный список: ";

add\_el\_by\_data(list\_1, s\_4, new\_data);

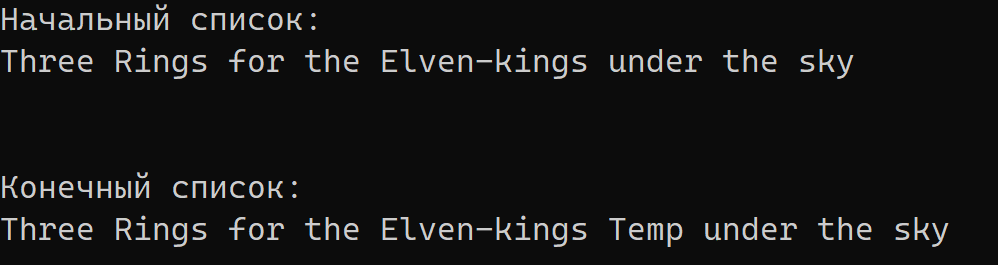
cout << endl;

print\_list(list\_1);

return 0;

}

Скриншоты результатов:



Анализ результатов:

Программа работает правильно. Выводится: начальный двусвязный список и конечный двусвязный список.