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:

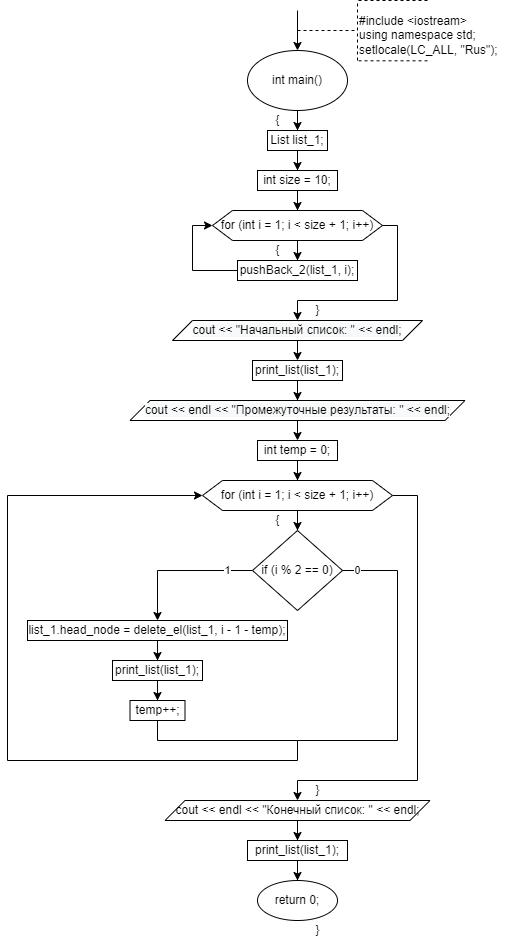
Тип информационного поля double. Удалить из списка элементы с четными номерами (2, 4, 6 и т.д.).

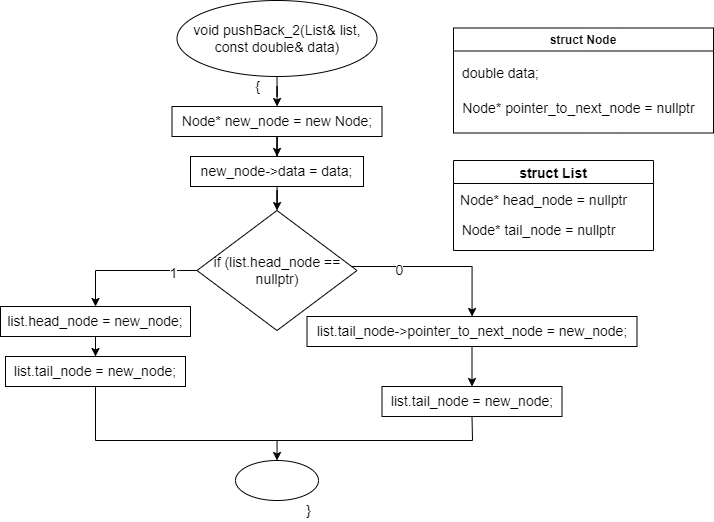
## Описание переменных:

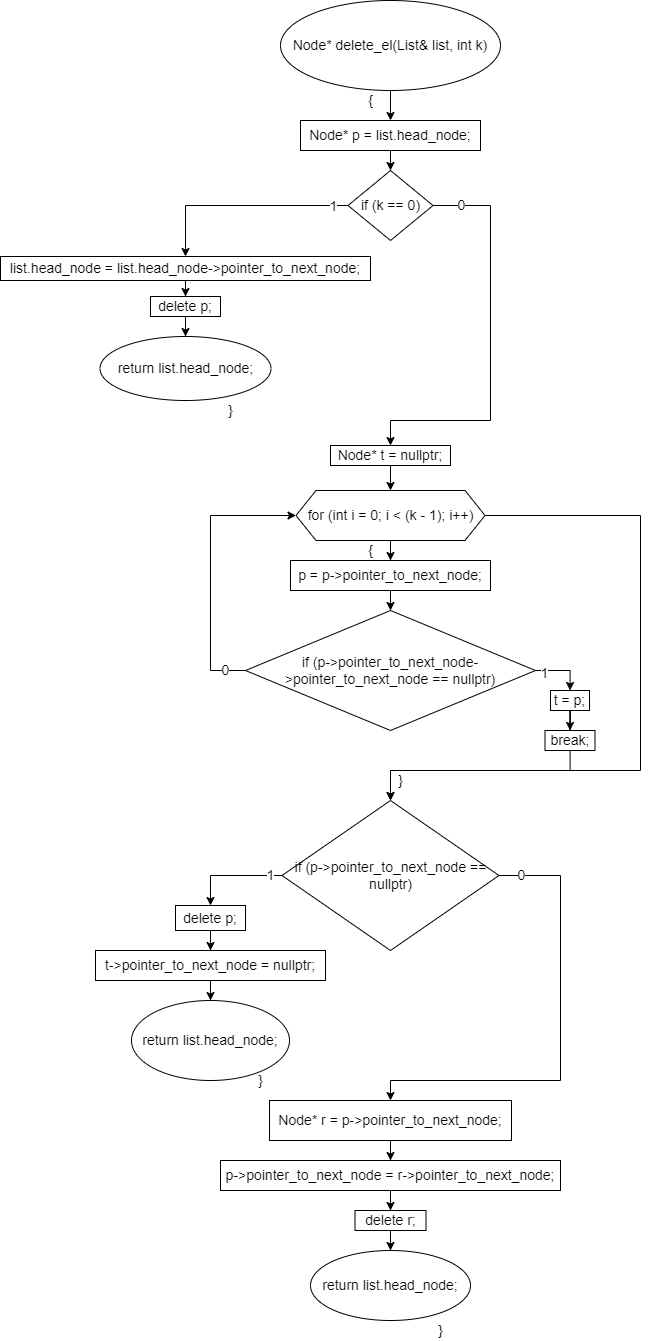
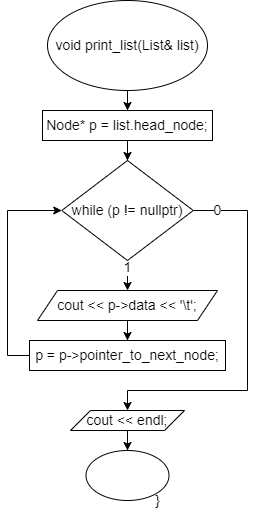
list\_1 – односвязный список

size – длинна односвязного списка

**Блок схема**

****





## Код программы

#include <iostream>

using namespace std;

// узел

struct Node {

double data;

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

};

// список

struct List {

Node\* head\_node = nullptr;

Node\* tail\_node = nullptr;

};

// добавление элемента в конец списка

void pushBack\_2(List& list, const double& data) { // добавление узла в конец списка 2

Node\* new\_node = new Node;

new\_node->data = data;

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

list.head\_node = new\_node;

list.tail\_node = new\_node;

}

else {

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

list.tail\_node = new\_node;

}

}

// вывод списка

void print\_list(List& list) {

Node\* p = list.head\_node;

while (p != nullptr) {

cout << p->data << '\t';

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

}

cout << endl;

}

// удаление элемента по номеру

Node\* delete\_el(List& list, int k) {

Node\* p = list.head\_node;

if (k == 0) {

list.head\_node = list.head\_node->pointer\_to\_next\_node;

delete p;

return list.head\_node;

}

Node\* t = nullptr;

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

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

if (p->pointer\_to\_next\_node->pointer\_to\_next\_node == nullptr) {

t = p;

break;

}

}

if (p->pointer\_to\_next\_node == nullptr) {

delete p;

t->pointer\_to\_next\_node = nullptr;

return list.head\_node;

}

Node\* r = p->pointer\_to\_next\_node;

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

delete r;

return list.head\_node;

}

int main()

{

setlocale(LC\_ALL, "Rus");

// создаём список

List list\_1;

int size = 10;

// заполняем поле data узлов списка числами от 1 до size

for (int i = 1; i < size + 1; i++) {

pushBack\_2(list\_1, i);

}

cout << "Начальный список: " << endl;

print\_list(list\_1);

// удаление элементов с чётными номерами

cout << endl << "Промежуточные результаты: " << endl;

int temp = 0;

for (int i = 1; i < size + 1; i++) {

if (i % 2 == 0) {

list\_1.head\_node = delete\_el(list\_1, i - 1 - temp);

print\_list(list\_1);

temp++;

}

}

// выводим конечный список

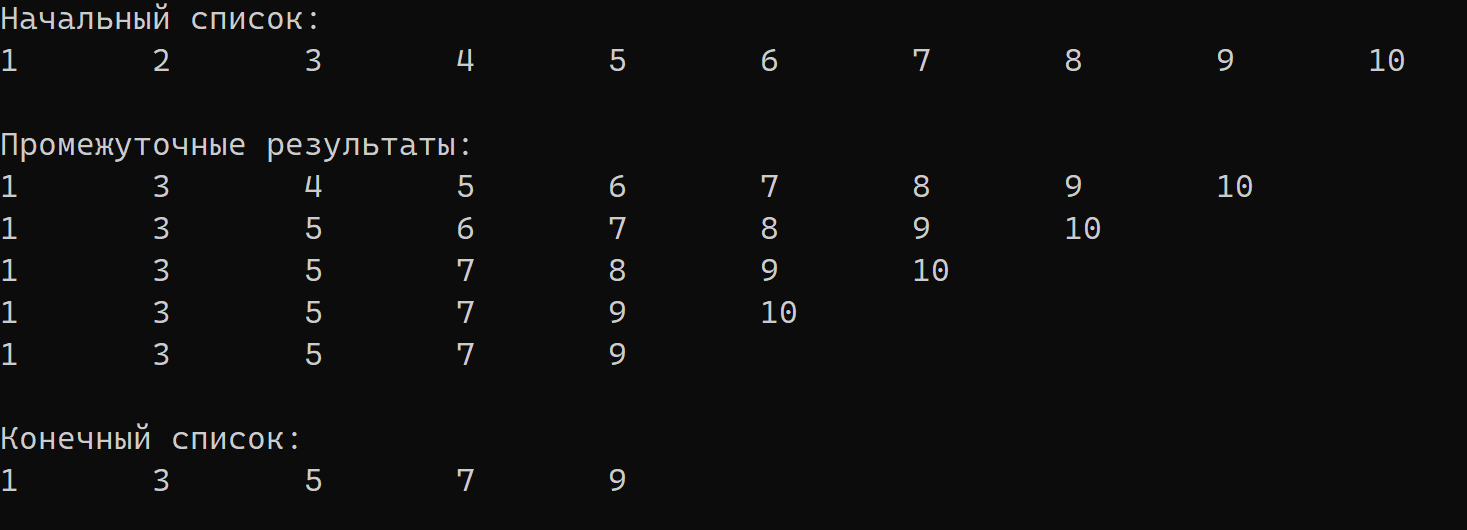
cout << endl << "Конечный список: " << endl;

print\_list(list\_1);

return 0;

}

Скриншоты результатов:



Анализ результатов:

Программа работает правильно. Выводится: начальный список, промежуточный односвязный список, конечный односвязный список.