Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №11.1**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Информационные и динамические структуры. Одно- и Двунаправленные списки Вариант 17

Выполнил:

Студент группы Рис-20-1б

Томилов Владислав Алексеевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

1) Получить практические навыки работы с однонаправленными списками;

2) Получить практические навыки работы с двунаправленными списками;

**Постановка задачи**

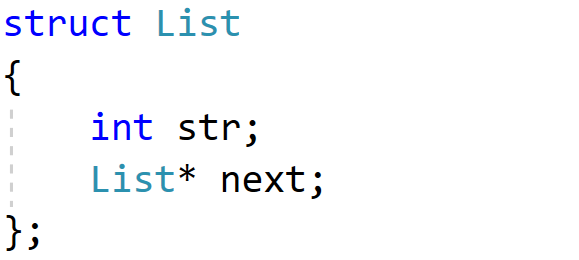
(Вариант 17)

1. Тип информационного поля int. Удалить из списка все элементы с четными информационными полями.
2. Тип информационного поля char\*. Добавить в список элемент с заданным номером

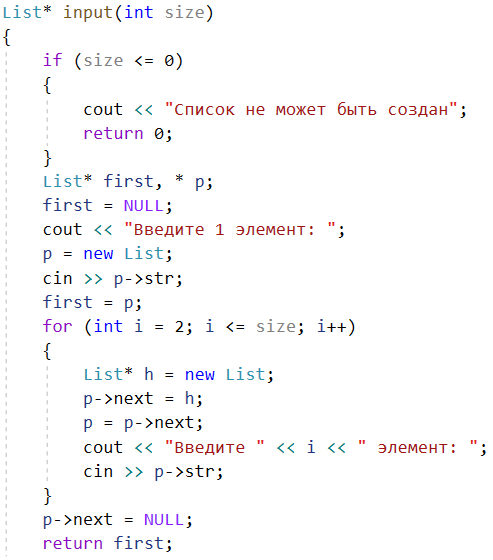
**11.1 Анализ задачи**

**1.** Для решения задачи необходимо:

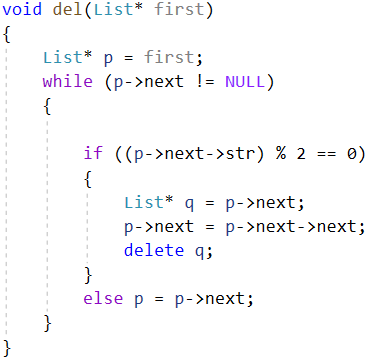
**1.1. (11.1)** Создать структуру List, в которой будет информационное поле и указатель на следующий элемент в списке



**1.2.** Разработать функцию с вводом элементов списка input. Определить можно ли создать список с помощью условия (size <= 0). Вводится первый элемент списка - у него указатель NULL, затем последовательно вводятся оставшиеся элементы списка, начиная со второго элемента.



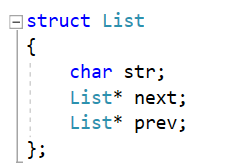
**1.3.**Разработать функцию del, которая будет проверять на четность элементы списка и удалять четные из них.



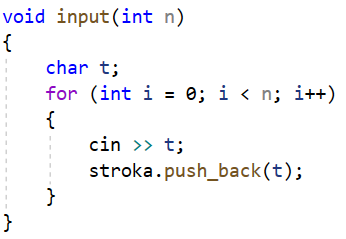
**1.4.** Разработать функцию output, которая будет выводить элементы изменённого списка в консоль.

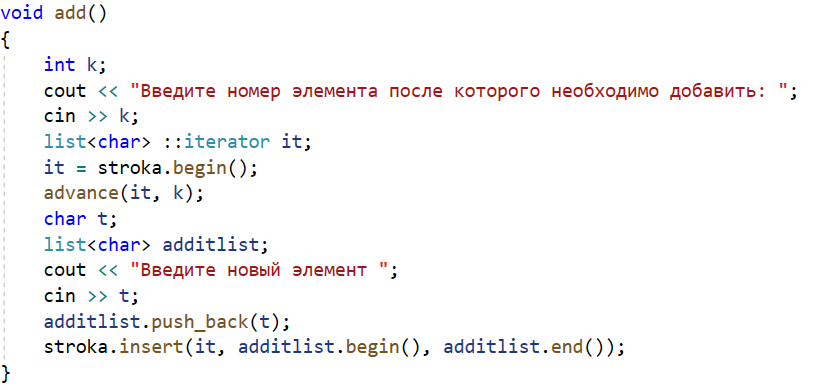


**1.5. (11.2)** Создать структуру List, в которой будет информационное поле и указатель на следующий и предыдущий элемент в списке.



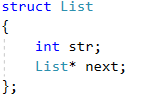
**1.6.** Разработать функцию с вводом элементов списка void input(int n). N – размер списка. Элементы по очереди вводятся пользователем в консоль и записываются в конец списка с помощью операции stroka.push\_back(t).



**1.7.** Разработать функцию с добавлением элементов списка void add(). Сначала запрашивается номер элемента, после которого необходимо добавить элемент списка. С помощью итератора переходим к введенному номеру элемента и после него добавляем с помощью операции additlist.push\_back (t).

**2.**В ходе работы были использованы следующие типы данных:

**2.1. (11.1)** Тип информационного поля элементов в списке int.



**2.2.** Для функции input используются следующие аргументы:

1. Тип int: Размер списка, который пользователь будет вводить в консоль.



**2.4.** Для функции del пользуются следующие аргументы:

1. Указатель на первый элемент списка



Сама функция имеет тип void, так как в ходе ее работы нет необходимости возвращать значение для этой функции.

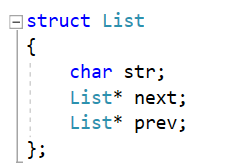
**2.5.** Для функции output используются следующие аргументы:

1. Указатель на первый элемент списка

Сама функция имеет тип void, так как в ходе ее работы нет необходимости возвращать значение для этой функции.



**2.6. (11.2)** Тип информационного поля элементов в списке int.



**2.7.** Для функции input используются следующие аргументы:

1. Тип int: Размер списка, который пользователь будет вводить в консоль.



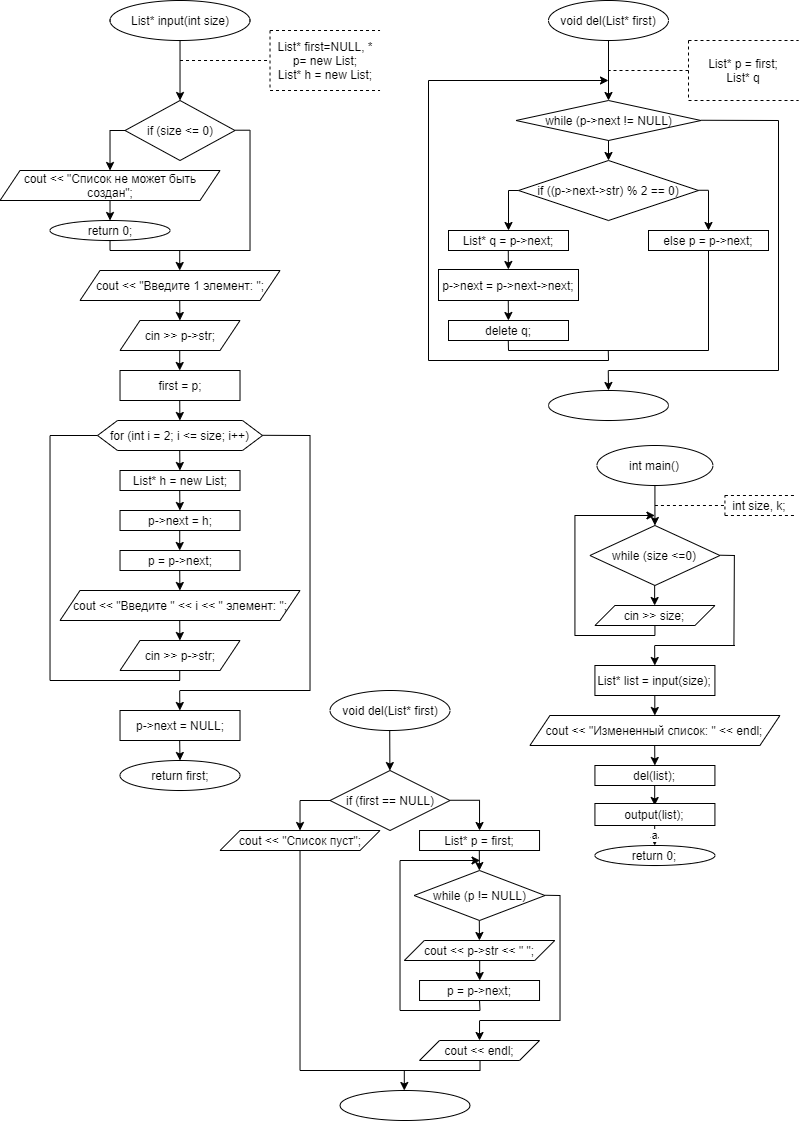
Сама функция имеет тип void, так как в ходе ее работы нет необходимости возвращать значение для этой функции.

**3.**Для решения задачи данные были представлены в следующем виде:

**3.1.** Ввод данных производится пользователем через консоль.

**Блок-схема программы**

**(11.1)**



Код программы

(11.1)

#include <iostream>

using namespace std;

struct List

{

int str;

List\* next;

};

List\* input(int size)

{

if (size <= 0)

{

cout << "Список не может быть создан";

return 0;

}

List\* first, \* p;

first = NULL;

cout << "Введите 1 элемент: ";

p = new List;

cin >> p->str;

first = p;

for (int i = 2; i <= size; i++)

{

List\* h = new List;

p->next = h;

p = p->next;

cout << "Введите " << i << " элемент: ";

cin >> p->str;

}

p->next = NULL;

return first;

}

void del(List\* first)

{

List\* p = first;

while (p->next != NULL)

{

if ((p->next->str) % 2 == 0)

{

List\* q = p->next;

p->next = p->next->next;

delete q;

}

else p = p->next;

}

}

void output(List\* first)

{

if (first == NULL)

{

cout << "Список пуст";

}

else

{

List\* p = first;

while (p != NULL)

{

cout << p->str << " ";

p = p->next;

}

cout << endl;

}

}

int main()

{

system("chcp 1251 > nul");

int size, k;

cout << "Введите размер списка: ";

while (!(cin >> size) || (cin.peek() != '\n') || size <= 0)

{

cin.clear();

system("cls");

while (cin.get() != '\n');

{

cout << "Введите размер списка: ";

}

}

List\* list = input(size);

cout << "Измененный список: " << endl;

del(list);

output(list);

}

Код программы

(11.2)

#include <iostream>

#include <list>

#include <iterator>

using namespace std;

list<char> stroka;

struct List

{

char str;

List\* next;

List\* prev;

};

void input(int n)

{

char t;

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

{

cin >> t;

stroka.push\_back(t);

}

}

void add()

{

int k;

cout << "Введите номер элемента после которого необходимо добавить: ";

cin >> k;

list<char> ::iterator it;

it = stroka.begin();

advance(it, k);

char t;

list<char> additlist;

cout << "Введите новый элемент ";

cin >> t;

additlist.push\_back(t);

stroka.insert(it, additlist.begin(), additlist.end());

}

int main()

{

system("chcp 1251 > nul");

int size;

cout << "Введите размер списка: ";

while (!(cin >> size) || (cin.peek() != '\n') || size <= 0)

{

cin.clear();

system("cls");

while (cin.get() != '\n');

{

cout << "Введите размер списка: ";

}

}

input(size);

add();

cout << "Список: " << endl;

copy(stroka.begin(), stroka.end(), ostream\_iterator<char>(cout, " "));

return 0;

}