Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №11**

Дисциплина: основы алгоритмизации и программирования

Тема: “ Динамические структуры данных.

Однонаправленные списки”

Вариант 19

Выполнил:

Студент группы РИС-20-2Б

Пономарёв Артём Викторович

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

Пермь, 2021

**Цель задачи**

Получить практические навыки работы с однонаправленными списками.

**Постановка задачи**

Тип информационного поля char\*. Добавить в список элементы с номерами 1, 3, 5 и т. д.

2

**Анализ задачи**

**1.** Для решения задачи необходимо:

**1.1.** Организовать структуру, поля которой будут использоваться для ввода данных типа char, для хранения адреса следующего элемента списка.

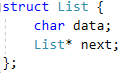
**1.2.** Организовать функцию make() для формирования списка.

**1.3.** Организовать функцию make2() для добавления нового элемента по нечётному индексу.

**1.4.** Организовать функцию GetList() для вывода списка.

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

**2.1.** Структура List для организации однонаправленного списка.



**2.2.** Для функции make() использовать в качестве параметра int size - количество элементов в списке.

List\* make(int size){

List\* first, \* p;

...

p = new List;

...

}

**2.3.** Для функции make2() использовать в качестве параметра указатель на первый элемент в списке, переменная x типа char, переменная i типа int, указатель p типа List.

List\* make2(List\* first){

char x;

int I;

List\* p = first;

}

3

**2.4.** Для функции GetList использовать указатель на первый элемент в списке в качестве параметра, указатель p типа List.

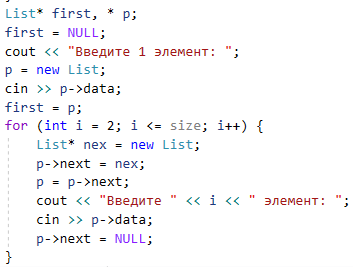
void GetList(List\* first){

List\* p = first;

}

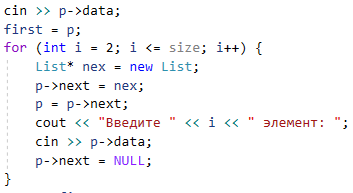
**3.** Для решения задачи данные были представлены в следующем виде:

**3.1.** Данные вводятся через консоль в виде однонаправленного списка через структуру List с полем типа char и указателем на первый элемент списка.



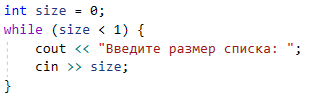
**4.** Для операций ввода и вывода использовались следующие операторы и функции:

**4.1.** Ввод данных на консоль реализован функцией cin для первого элемента списка и в цикле for (также через cin) для следующих.



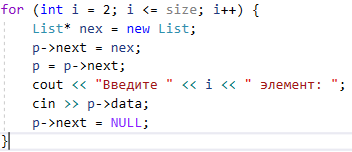
**4.2.** В функции main() вводится количество элементов в списке в переменной size через цикл while, чтобы пользователь не ввёл отрицательное число.

4

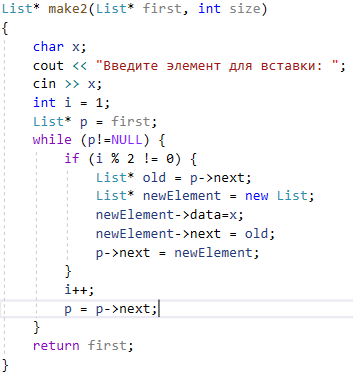


**5.** Поставленные задачи будут решены следующими действиями:

**5.1.** В функции make() осуществляется ввод всех элементов в списке. Первый вводится вне цикла for и используется в качестве указателя на список. В цикле for заполняются элементы от 2 до size: резервируется память в переменной nex, в которую мы вводим следующий элемент после первого.

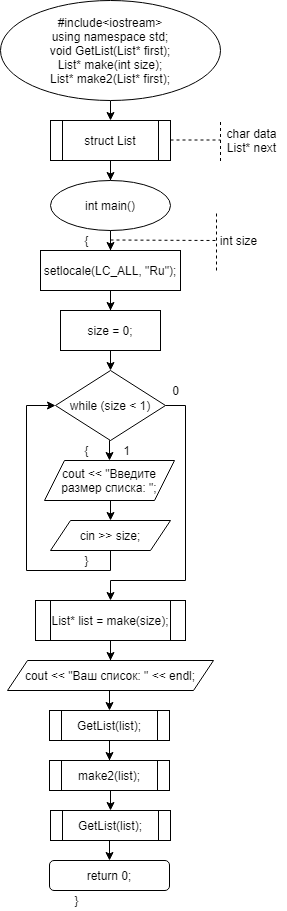


**5.2.** В функции make2() в переменную x ввести элемент, который необходимо вставить по нечётным индексам списка. Организуется цикл while, в котором в ветвлении для нечётных индексов произвести следующие действия: в указателе old сохранить следующий элемент после первого, произвести резервирование памяти для переменной newElement для записи в неё x через метод .data структуры List. Через метод next для newElement присвоить указатель на второй элемент списка (old), чтобы сделать его (второй элемент) третьим. Через метод next для указателя p присвоить newElement.

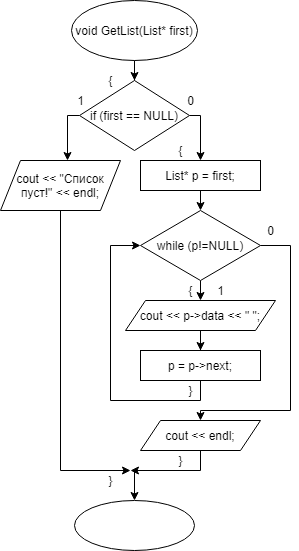


5

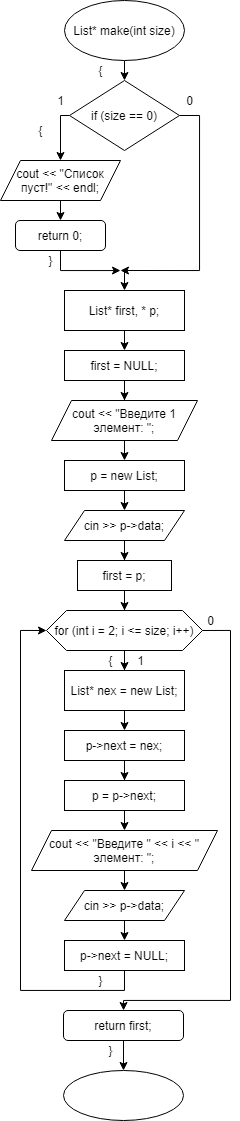
**Блок-схема**



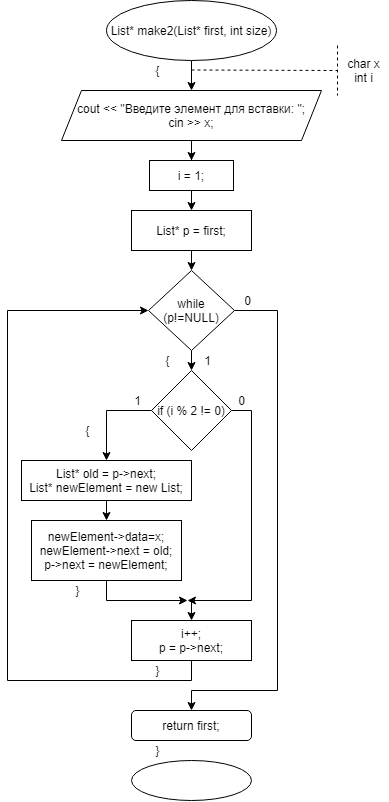
6



7



8



9

**Код**

#include<iostream>

using namespace std;

struct List {

char data;

List\* next;

};

void GetList(List\* first);

List\* make(int size);

List\* make2(List\* first);

int main()

{

setlocale(LC\_ALL, "Ru");

int size = 0;

while (size < 1) {

cout << "Введите размер списка: ";

cin >> size;

}

List\* list = make(size);

cout << "Ваш список: " << endl;

GetList(list);

make2(list);

GetList(list);

}

List\* make(int size)

{

if (size == 0) {

cout << "Список пуст!" << endl;

return 0;

}

List\* first, \* p;

first = NULL;

cout << "Введите 1 элемент: ";

p = new List;

cin >> p->data;

first = p;

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

List\* nex = new List;

p->next = nex;

p = p->next;

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

cin >> p->data;

p->next = NULL;

}

return first;

}

void GetList(List\* first)

{

if (first == NULL)

cout << "Список пуст!" << endl;

else {

List\* p = first;

while (p!=NULL) {

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

p = p->next;

}

cout << endl;

}

}

10

List\* make2(List\* first)

{

char x;

cout << "Введите элемент для вставки: ";

cin >> x;

int i = 1;

List\* p = first;

while (p!=NULL) {

if (i % 2 != 0) {

List\* old = p->next;

List\* newElement = new List;

newElement->data=x;

newElement->next = old;

p->next = newElement;

}

i++;

p = p->next;

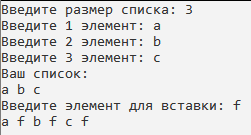
}

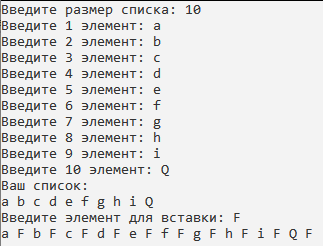
return first;

}

11

**Скриншоты**





12