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 №11.1**

Дисциплина: основы алгоритмизации и программирования

Тема: динамические структуры данных. Одно— и двунаправленные списки

Вариант 5

Выполнил работу

студент группы РИС-20-1б

Зверев А.Д.

Проверила

Доцент кафедры ИТАС

Полякова О.А.

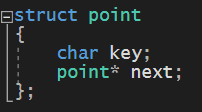
Пермь, 2021

**Постановка задачи**

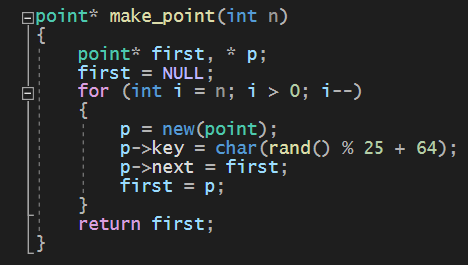
1. Сформировать однонаправленный список типа char\*.
2. Распечатать полученную структуру.
3. Добавить в список элемент с заданным информационным полем.
4. Распечатать полученный результат.

**Анализ задачи**

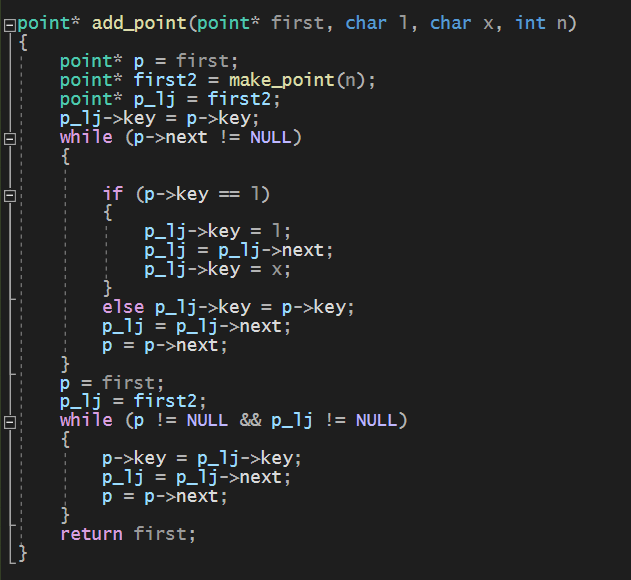
1. Для решения задачи необходимо…
   1. Создать структуру point, которая содержит в себе данные типа char и указатель на следующий элемент одностороннего списка;



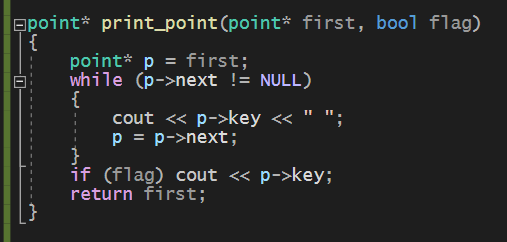
* 1. Написать функцию make\_point, которая будет давать значение элементам списка;



* 1. Написать функцию add\_point, которая будет добавлять элементы после нужного;



* 1. Написать функцию print\_point, которая выводит элементы списка в консоль;



* 1. Вызвать все вышеперечисленные функции в основном теле программы;





1. Для решения задачи были использованы данные следующих типов:
   1. Bool для логического флажка;



* 1. Char для хранения элементов однонаправленного списка;



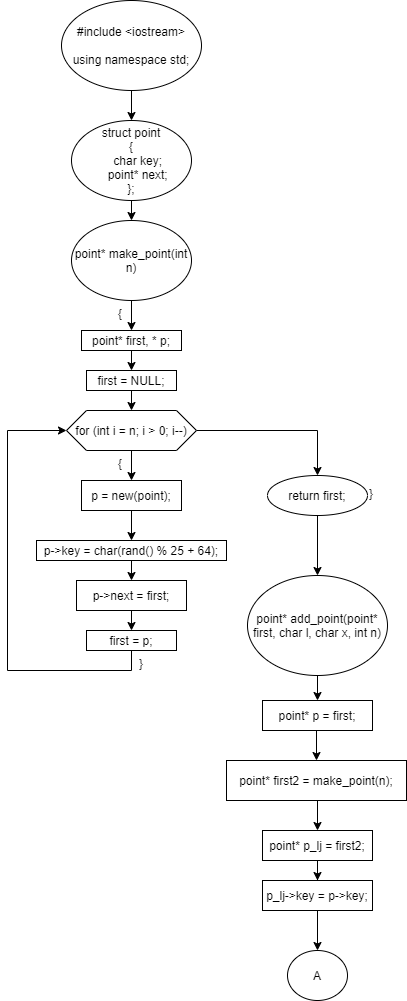
* 1. Int для локальных переменных цикла и для хранения количества элементов списка;

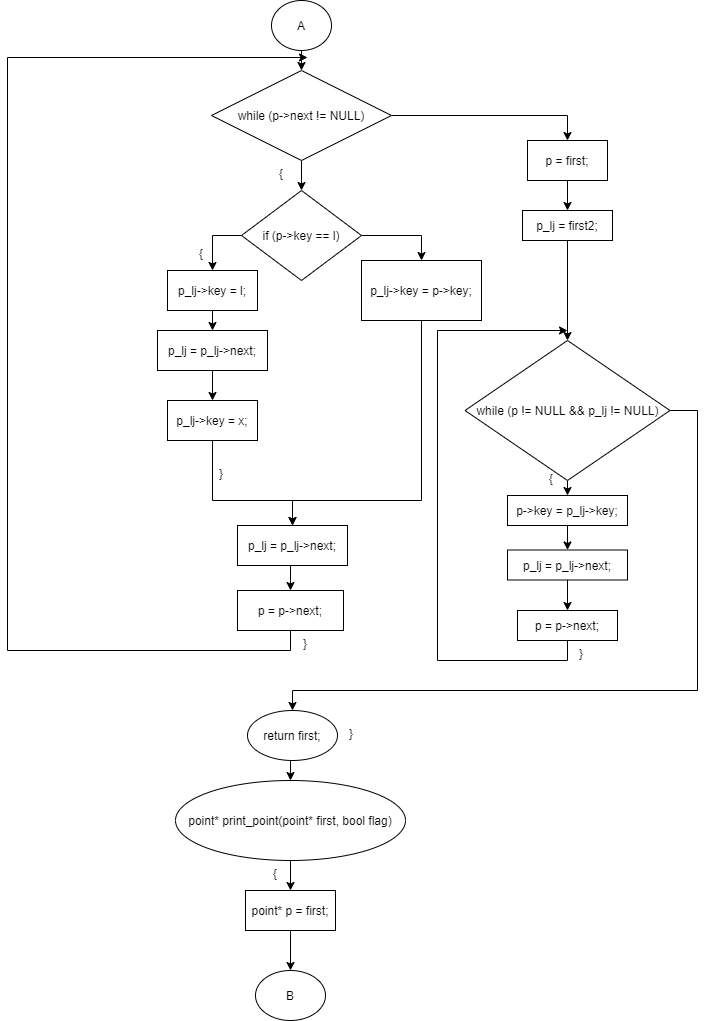


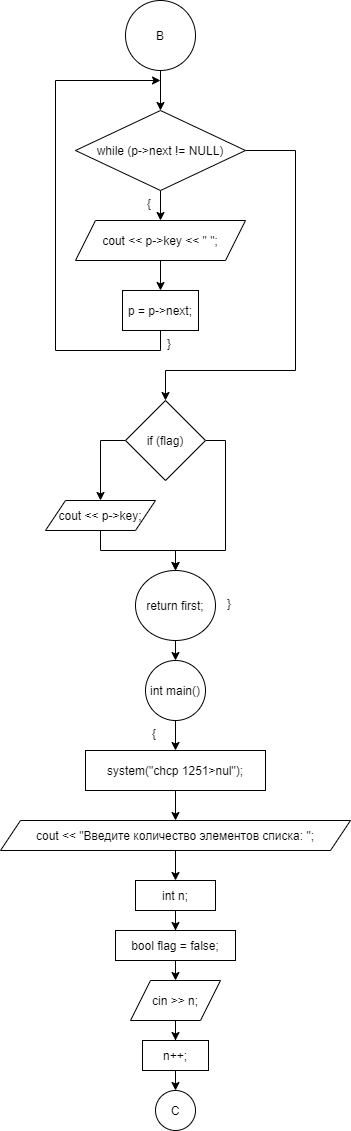
* 1. Структура для хранения однонаправленного списка;

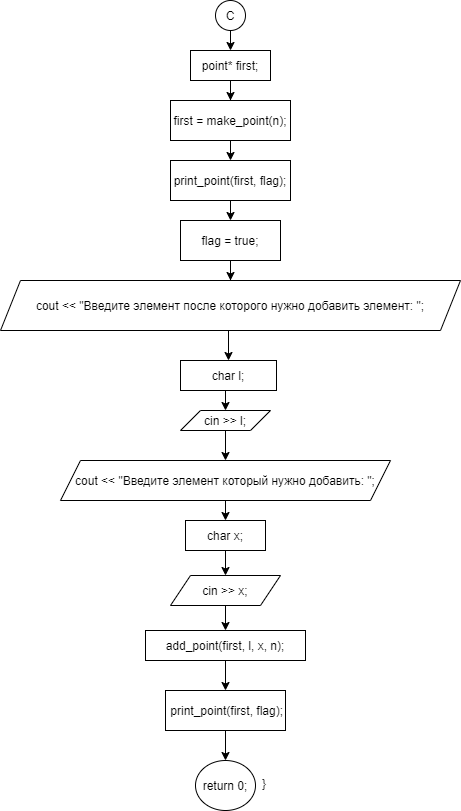


**Блок—схема программы**









**Решение**

#include <iostream>

using namespace std;

struct point

{

char key;

point\* next;

};

point\* make\_point(int n)

{

point\* first, \* p;

first = NULL;

for (int i = n; i > 0; i--)

{

p = new(point);

p->key = char(rand() % 25 + 64);

p->next = first;

first = p;

}

return first;

}

point\* add\_point(point\* first, char l, char x, int n)

{

point\* p = first;

point\* first2 = make\_point(n);

point\* p\_lj = first2;

p\_lj->key = p->key;

while (p->next != NULL)

{

if (p->key == l)

{

p\_lj->key = l;

p\_lj = p\_lj->next;

p\_lj->key = x;

}

else p\_lj->key = p->key;

p\_lj = p\_lj->next;

p = p->next;

}

p = first;

p\_lj = first2;

while (p != NULL && p\_lj != NULL)

{

p->key = p\_lj->key;

p\_lj = p\_lj->next;

p = p->next;

}

return first;

}

point\* print\_point(point\* first, bool flag)

{

point\* p = first;

while (p->next != NULL)

{

cout << p->key << " ";

p = p->next;

}

if (flag) cout << p->key;

return first;

}

int main()

{

system("chcp 1251>nul");

cout << "Введите количество элементов списка: ";

int n;

bool flag = false;

cin >> n;

n++;

point\* first;

first = make\_point(n);

print\_point(first, flag);

flag = true;

cout << "Введите элемент после которого нужно добавить элемент: ";

char l;

cin >> l;

cout << "Введите элемент который нужно добавить: ";

char x;

cin >> x;

add\_point(first, l, x, n);

print\_point(first, flag);

return 0;

}

**Скриншоты**

