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 №14**

Дисциплина: основы алгоритмизации и программирования

Тема: «Поиск данных в динамических структурах» в C++

Выполнил работу

студент группы РИС-20-2б

Пономарев Егор Витальевич

Проверила

доцент кафедры ИТАС

Полякова Ольга Андреевна

Пермь, 2021

**Цель работы**

**Постановка задачи**

1. Создать динамический массив записей (в соответствии с вариантом), содержащий не менее 100 элементов.
2. Выполнить поиск элемента в массиве по ключу в соответствии с вариантом. Для поиска использовать метод Бойера-Мура и Кнута-Морриса-Пратта.

Вариант 17.

Данные: ФИО, дата рождения, номер телефона

Ключ: ФИО

**Анализ задачи**

I. Необходимые действия:

1. Создать динамический массив структур с полями для ФИО, даты рождения и номера телефона;
2. Предоставить пользователю выбор между методами поиска, которые будут использоваться, где 1 – Бойра- Мура, 2 – Кнута Морриса Пратта;
3. При выборе пользователем цифры 1 (Бойра-Мура), ввести ключ, по которому будет осуществляться поиск;
4. Вызвать функцию, осуществляющую поиск метода Бойра-Мура, в которой изначально присваиваем к элементам шаблона значение определенной ячейки, чтобы впоследствии длина сдвига шаблона была равна значению ячейки. Затем мы сравниваем элементы, при несовпадении происходит сдвиг шаблона, а при совпадении всех элементов происходит окончание поиска совпадающих элементов;
5. При выборе пользователем цифры 2 (поиск Кнута Морриса Пратта), ввести ключ, по которому будет осуществляться поиск;
6. Вызвать функцию поиска методом Кнута Морриса Пратта, в которой происходит создание массива для сдвига подстроки вдоль строки и на основании этого массива ищется вхождение подстроки в строку;
7. Вывести сообщение, о том, где находится элемент, который ищут, если же элемент был не найден, требуется также сообщить об этом.

II. Типы переменных:

1. int;
2. long long;
3. string;
4. struct;.

III. Представление данных:

1. int: переменная для ввода номера выбора метода поиска (1-линейный, 2-интерполяционный, 3-прямой);

int num;

cout << endl;

cout << "Выберите, с помощью какого метода будет осуществлять поиск: " << endl << endl;

cout << "1. Метод поиска Бойера-Мура" << endl;

cout << "2. Метод поиска Кнута-Морриса-Пратта" << endl;

cout << "Ваш выбор: "; cin >> num;

1. long long: переменная для хранения данных номеров телефона(диапазон от -9 223 372 036 854 775 808 до + 9 223 372 036 854 775 807)

long long number

1. string: переменная для хранения ключей, полей структуры: ФИО и дата рождения;

string key;

string fio;

string data;

1. struct: представление данных о каком-либо человеке (ФИО, дата рождения, номер телефона);

struct List

{

string fio;

string data;

long long number;

};

IV. Поля структуры:

1. fio – хранение фамилии, имени и отчества человека;
2. data – хранение даты рождения;
3. number – хранение номера телефона.

V. Операторы ввода/вывода:

1. cout – вывод данных на консоль;
2. cout << "Введите ФИО, которое Вы ищете: ";

cout << line;

cout << "Выберите, с помощью какого метода будет осуществлять поиск: " << endl << endl;

cout << "1. Метод поиска Бойера-Мура" << endl;

cout << "2. Метод поиска Кнута-Морриса-Пратта" << endl;

cout << "Ваш выбор: ";

1. cin – ввод числовой информации;

cin >> n;

cin >> num

1. getline – ввод строковой информации, для которого требуется вывод всего, что вводится и ввод строк из входного файла;

getline(cin, key);

cout << "Полное имя: ";

getline(cin, users[i].fio);

cout << "Дата рождения: ";

getline(cin, users[i].data);

VI. Используемые операторные структуры:

1. Цикл с постусловием;

while (index\_line < lenline)

{

if (line[index\_line] == key[index\_key])

{

index\_line++;

index\_key++;

if (index\_key == lenkey)

{

string tmp;

tmp = tmp + line[index\_line];

int i = 1;

while (index\_line < lenline && (line[index\_line + 1] != ' ')) {

tmp = tmp + line[index\_line + i];

index\_line++;

}

cout << "Пользователь найден. Он находится под номером: " << stoi(tmp);

}

}

if (line[index\_line] != key[index\_key])

{

if (index\_key == 0)

{

index\_line++;

}

else

{

index\_key = array[index\_key - 1];

}

}

}

while (p < str)

{

if (key[substr - 1] != line[p])

{

p += b[int((unsigned char)line[p])];

}

else

{

for (i = substr - 1; i >= 0; i--)

{

if (key[i] != line[p - substr + i + 1])

{

p += b[int((unsigned char)line[p - substr + i + 1])];

break;

}

else if (i == 0)

{

string tmp;

tmp = tmp + line[p + 1];

int i = 2;

while (p + i < str && line[p + i] != ' ') {

tmp = tmp + line[p + i];

i++;

}

return stoi(tmp);

}

}

}

1. Цикл с счетчиком for;

for (int v = 0; v < lenkey; v++)

{

if (key[i] == key[j])

{

array[i] = j + 1; i++; j++;

}

else if (j == 0)

{

array[i] = 0;

i++;

}

else

{

j = array[j - 1];

}

}

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

{

cout << " " << array[i];

}

for (int i = 1; i <= n; i++)

{

line = line + users[i - 1].fio + to\_string(i) + " ";

}

for (int v = 0; v < lenkey; v++)

{

if (key[i] == key[j])

{

array[i] = j + 1; i++; j++;

}

else if (j == 0)

{

array[i] = 0;

i++;

}

else

{

j = array[j - 1];

}

}

1. Оператор ветвления is-else;

if (key[i] == key[j])

{

array[i] = j + 1; i++; j++;

}

else if (j == 0)

{

array[i] = 0;

i++;

}

else

{

j = array[j - 1];

}

if (line[index\_line] == key[index\_key])

{

index\_line++;

index\_key++;

if (index\_key == lenkey)

{

string tmp;

tmp = tmp + line[index\_line];

int i = 1;

while (index\_line < lenline && (line[index\_line + 1] != ' ')) {

tmp = tmp + line[index\_line + i];

index\_line++;

}

cout << "Пользователь найден. Он находится под номером: " << stoi(tmp);

}

}

if (line[index\_line] != key[index\_key])

{

if (index\_key == 0)

{

index\_line++;

}

else

{

index\_key = array[index\_key - 1];

}

}

1. Оператор switch-case;

switch (num)

{

case 1:

{

int element = BM\_search(users, key, line);

if (element != -1)

cout << "Номер элемента с введенным ФИО : " << element << endl;

else

cout << "Элемент с введенным ФИО не найден" << endl;

break;

}

case 2:

{

KMP\_search(users, key, line);

break;

}

default: cout << "Выбор некорректный"; break;

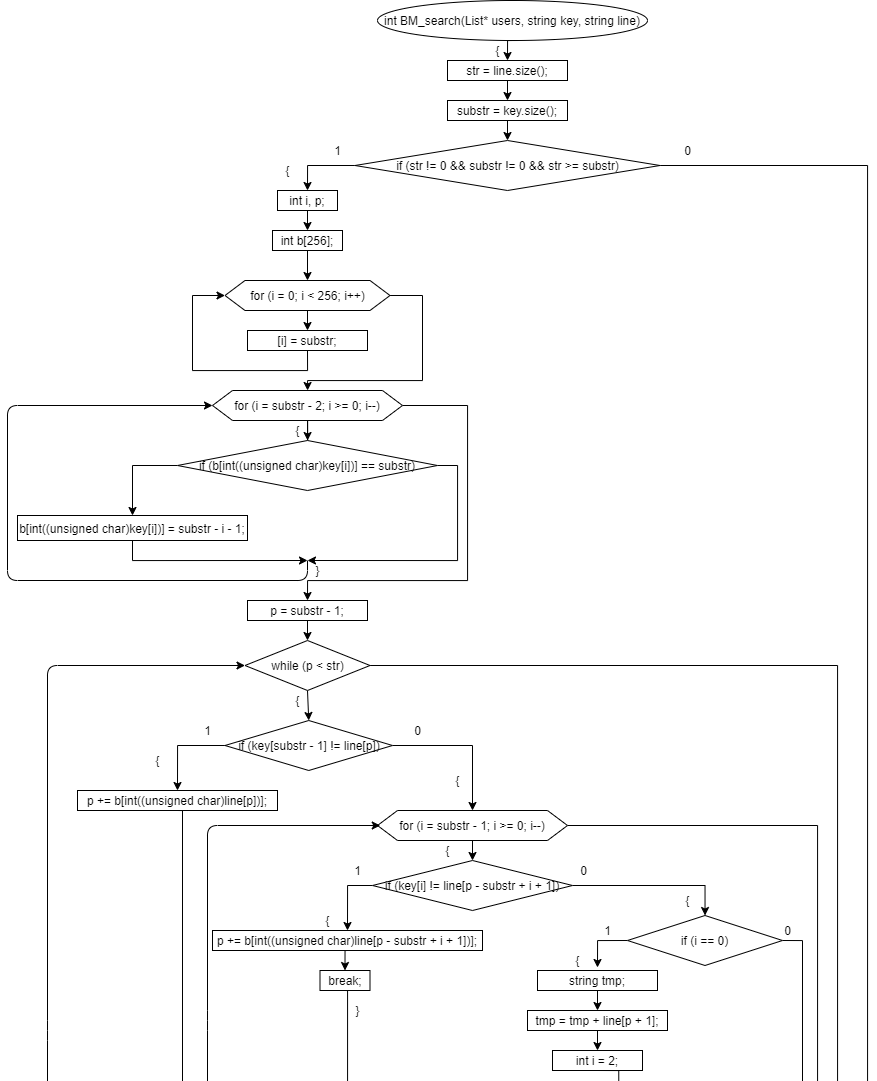
}

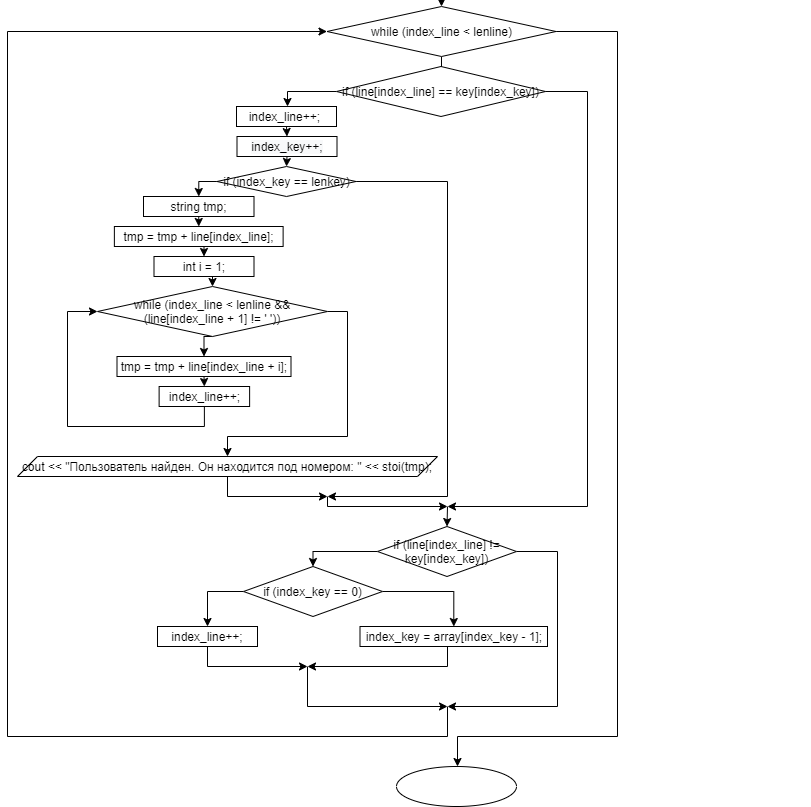
1. Оператор возврата return;

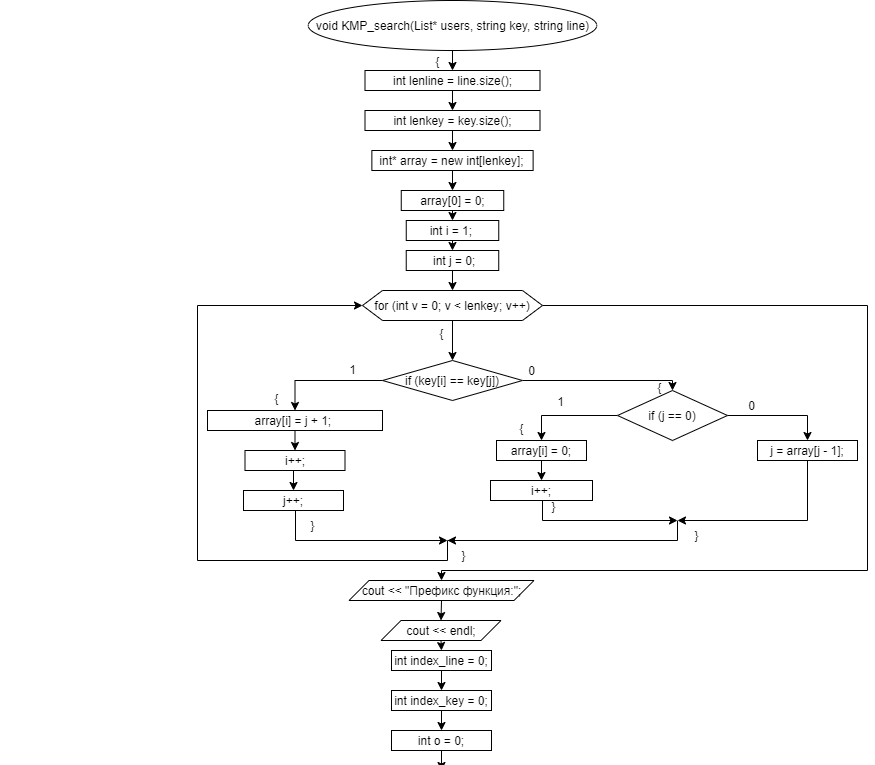
return stoi(tmp);

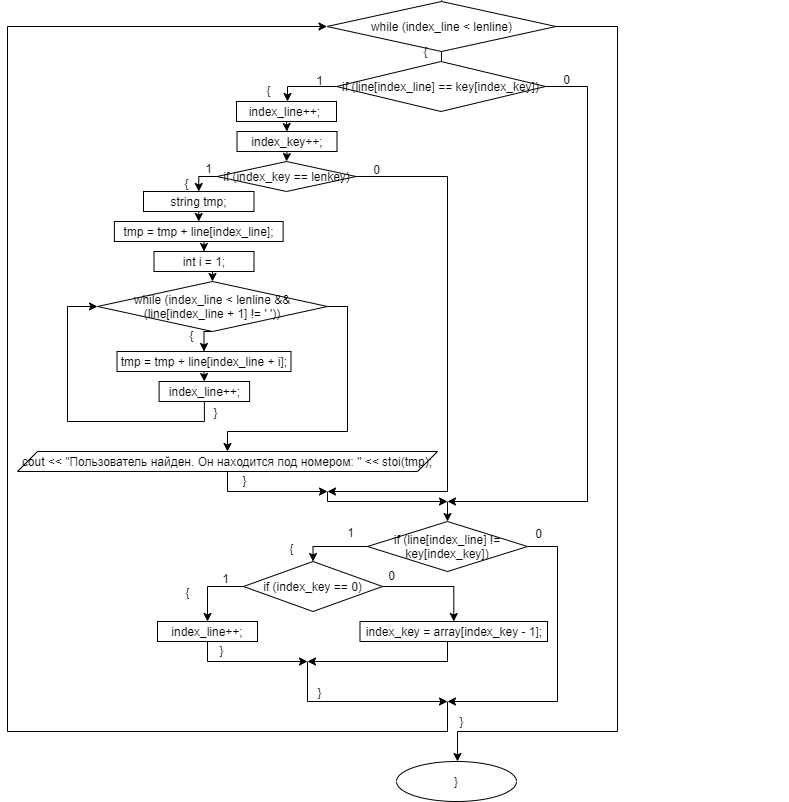
return -1;

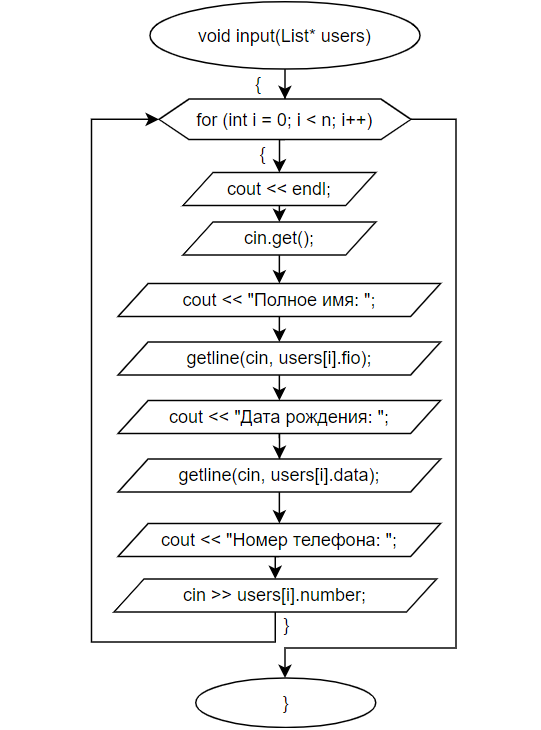
**Блок-схемы**

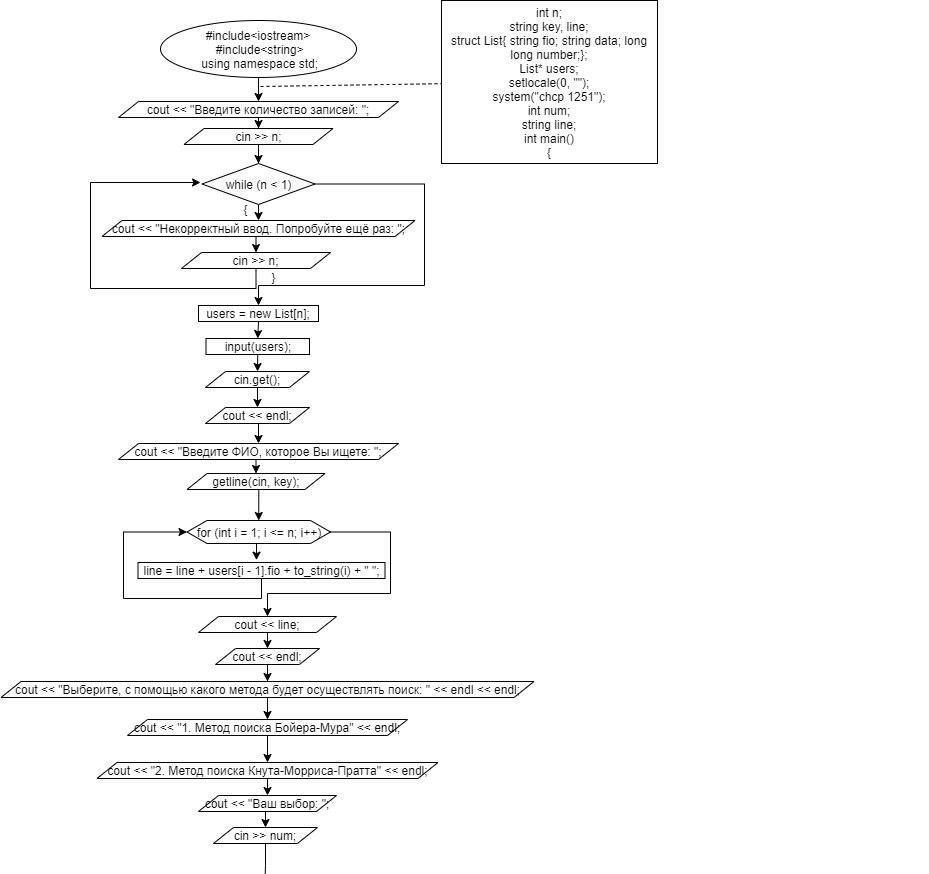


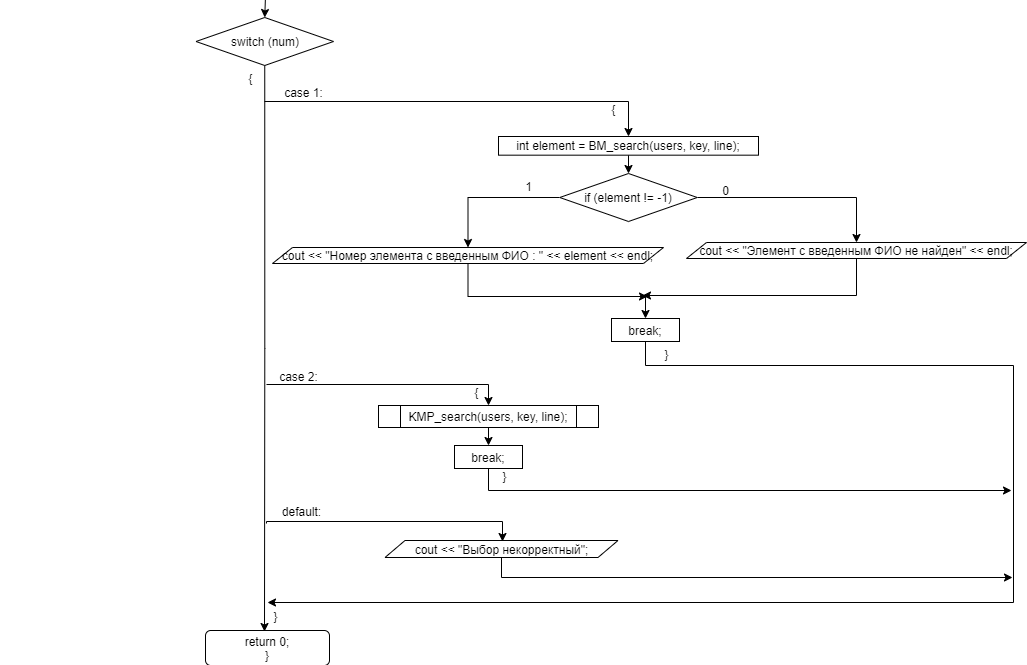












**Результаты работы программы**

