КАФЕДРА

КУРСОВОЙ ПРОЕКТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ

РАЗРАБОТКА ПРОГРАММЫ

«Информация о сотрудниках»

по дисциплине: ОСНОВЫ ПРОГРАММИРОВАНИЯ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

	подпись, дата	инициалы, фамилия

Санкт-Петербург 2022

Оглавление

1. Постановка задачи 3
2. Описание структур данных 3
3. Описание программы и созданных функций 4
4. Результаты тестирования программы 6
5. Заключение 12
6. Список использованной литературы 13
7. Исходный код программы на языке C++ 14

1. Постановка задачи

   Задачей курсового проекта является разработка программы для предметной области «Информация о сотрудниках» с использованием заданных структур данных, которая позволяет вводить информацию, хранить её в файле, осуществлять поиск, модификацию и удаление данных.

   Вариант 16

   Данные о человеке хранятся в структуре с именем ZNAK, содержащей следующие поля:

   - фамилия, имя;
   - знак зодиака;
   - дата рождения (массив из трёх чисел).

   Задание на поиск: найти информацию о людях, родившихся в месяц, значение которого введено с клавиатуры.

2. Описание структур данных

   Данные о человеке хранятся в массиве

```
struct ZNAK
{
        string name;        //  фамилия, имя
        string zodiak;      //  знак зодиака
        int birthday[3];    //  дата рождения
};
```

   string name - строка, состоящая только из алфавита кириллицы. Первая буква должна быть заглавная, последующие должны иметь нижний регистр, невозможно начать с букв “Ъ” и “Ь”, так же строка не должна содержать пробелов

   Допустимые названия знаков зодиака хранятся во множестве (контейнере set).

   int birthday[3] - массив на 3 элемента, содержащих целый числовой тип переменных.

Данные сохраняются в файл, имя файла вводит пользователь.

Формат хранения:

Фамилия Имя знак зодиака дата рождения

Для работы с БД в процессе выполнения программы используется линейный однонаправленный список.

3. Описание программы и созданных функций

Программа реализована на языке C++ в виде консольного приложения. В главной функции main() реализовано меню пользователя, в котором каждому действию соответствует определенная цифра. Реализованы следующие функции для работы с данными: добавление, редактирование, удаление записи, поиск людей по месяцу рождения, сохранение данных в текстовый файл, чтение данных из файла, вывод на экран.

При запуске программы на экран выводится консольное приложение с меню пользователя. При нажатии на клавиатуре на определенную цифру выполняется соответствующая функция.

Пользовательское меню:

1 - Добавить в конец

2 - Вывод списка

3 - Добавить в начало

4 - Поиск

5 - Поиск по месяцу рождения

6 - Удаление

7 - Редактировать

8 - Сохранить в файл

9 - Считать файл

0 – Выход

Выберите действие:

Добавление записи

void AddFirstElem(List **begin, ZNAK elem)

void AddElem(List **begin, List ** cur, ZNAK elem)

Функции выполняют добавление записи elem в начало и конец списка begin соответственно. При вводе данных выполняется проверка формата введенных данных. Фамилия и имя должны быть соответственно верно написаны. Строка со знаком зодиака должна содержать его корректное название, день рождения должен быть целым числом, месяц – от 1 до 12, год – от 1900 до 2022, дата рождения так же проходит проверку на количество дней в месяце, високосные года. Кроме того, все строковые значения не могут быть пустыми или содержать только пробельные символы. Если введены некорректные данные, пользователю предлагается ввести их повторно.

Удаление записи

void DelElem(List **begin, List* ptrCur)

Функция удаляет узел ptrCur из списка begin

Если при выборе данной функции, окажется, что база данных пуста, пользователю выведется на экран сообщение с ошибкой.

При выборе функции пользователю необходимо ввести все данные пользователя, которые необходимо удалить.

Редактирование

void UpdateElem(List** ptr, ZNAK r2)

Функция принимает на вход указатель на узел ptr списка, значение которого должно быть обновлено, а также запись r2, содержащую поля с новыми значениями для этого узла. Эти значения вводятся с клавиатуры.

Если при выборе данной функции, окажется, что база данных пуста, пользователю выведется на экран сообщение с ошибкой.

При выборе функции пользователю необходимо ввести все данные пользователя, которые необходимо редактировать, а затем ввести их новые значения.

Вывод на экран, файл

PrintList(List *begin, ostream &os)

Функция выводит в поток os содержимое списка begin. Для вывода на экран используется стандартный поток вывода cout. Если при выборе данной функции, окажется, что список пуст, пользователю выведется на экран сообщение с ошибкой.

Пример выполнения:

Иванов Сергей весы 11.10.1985
Абрамян Михаил водолей 15.02.1962
Асанова Диана скорпион 02.11.1976

Поиск

FindElem(List *begin, string sur)

FindElemMes(List* begin, ZNAK elem)

Функция выполняет поиск узла списка begin, для которого подстрока строки name, содержащей фамилию и имя, эквивалентна sur. Если такой узел не будет найден, программа выдаст соответствующее сообщение.

При поиске пользователю необходимо ввести все данные пользователя, которого необходимо найти. При поиске по месяцу рождения необходимо ввести месяц – при выводе будут все пользователи с данным месяцем.

4. Результаты тестирования программы

После запуска программы на экране появится окно консольного приложения, в котором отображаются пункты меню (рисунок 1).

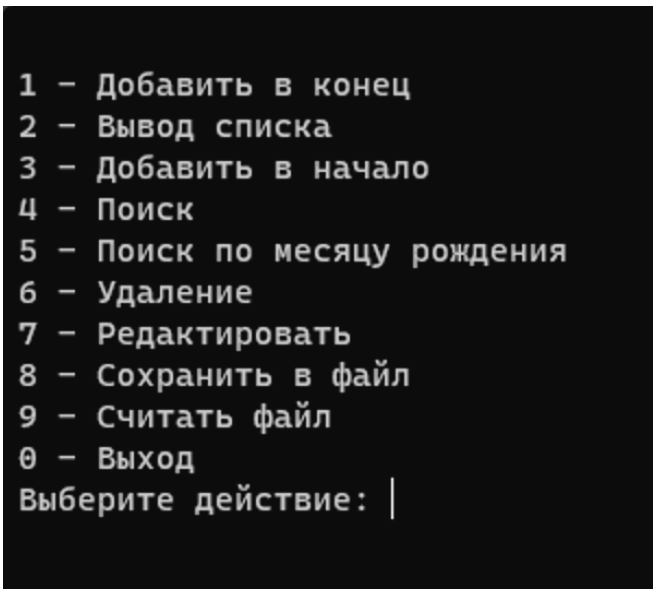

Рисунок 1 – Вид окна программы при запуске

При попытке вывести содержимое списка программа сообщит об ошибке, поскольку список пуст, после чего предложит выбрать команду меню еще раз (рисунок 2).

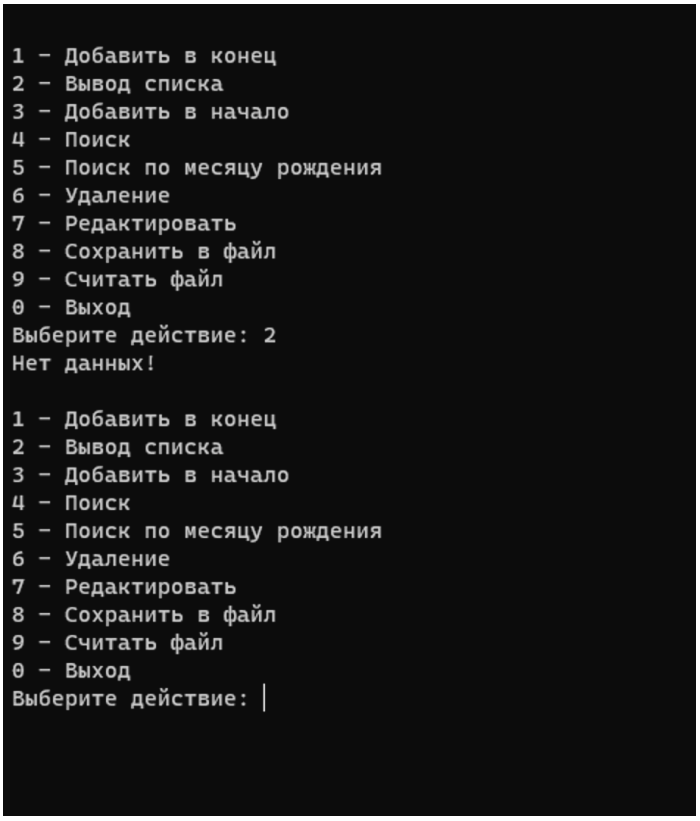

Рисунок 2 – Вид окна программы при попытке печати пустого списка

На рисунке 3 продемонстрирован вид окна программы в процессе добавления записи в конец списка. Как видно из рисунка, программа обрабатывает любые некорректные значения, предлагая пользователю выполнить повторный ввод.

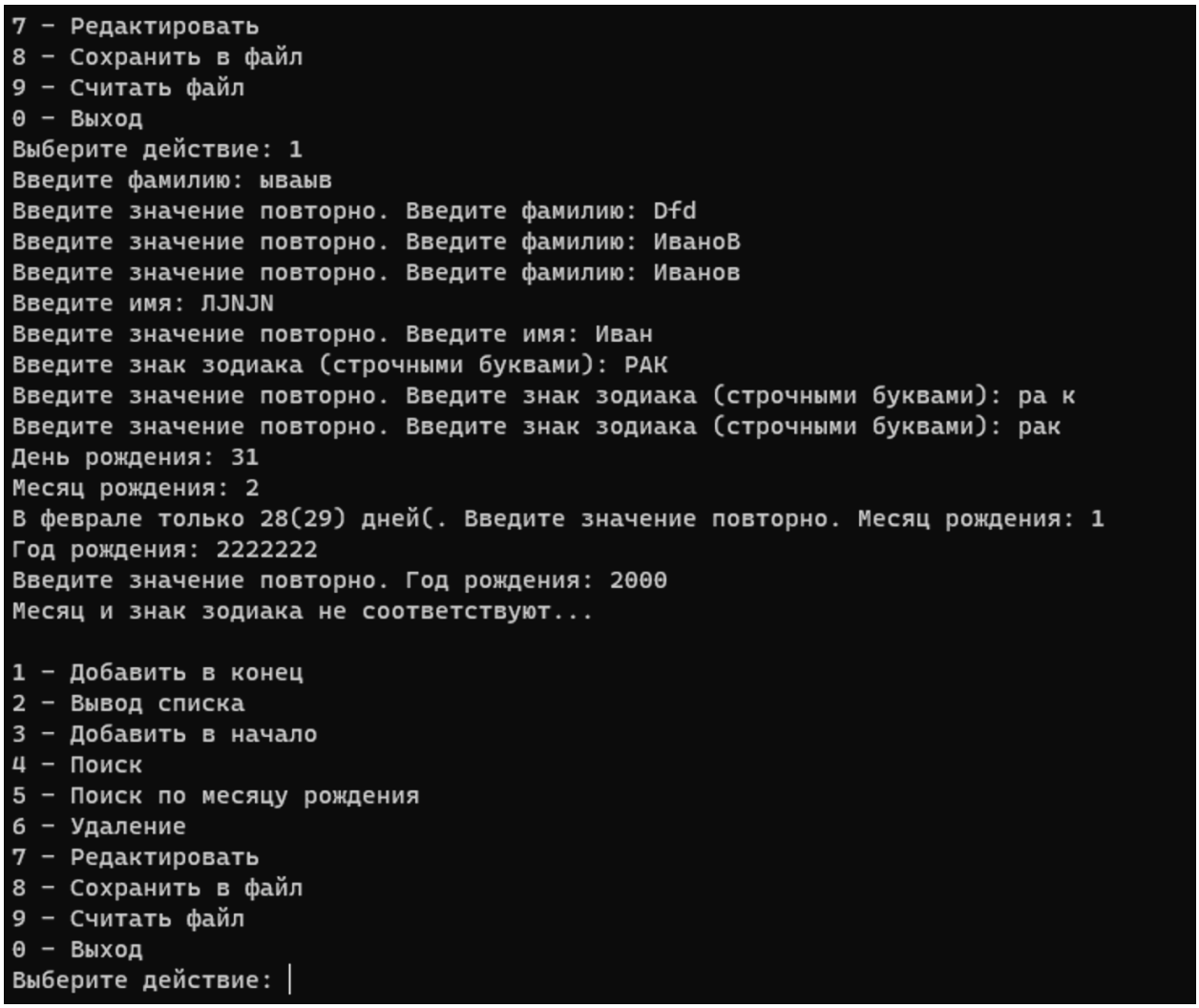

Рисунок 3 – Вид окна программы после добавления элемента в список

На рисунке 4 продемонстрирован вид окна программы после ввода нескольких записей и печати списка перед удалением одной из записей (на экране три записи) и после него (две записи на экране).

```
Выберите действие: 2
 Иванов Иван                          рак 31.1.2000
 Иванов Сергей                        весы 11.10.1985
 Асанова Диана                        скорпион 2.11.1976

1 - Добавить в конец
2 - Вывод списка
3 - Добавить в начало
4 - Поиск
5 - Поиск по месяцу рождения
6 - Удаление
7 - Редактировать
8 - Сохранить в файл
9 - Считать файл
0 - Выход
Выберите действие: 6
Введите фамилию: Иванов
Введите имя: Иван
Введите знак зодиака (строчными буквами): рак
День рождения: 31
Месяц рождения: 1
Год рождения: 2000
Месяц и знак зодиака не соответствуют...
Запись удалена!

1 - Добавить в конец
2 - Вывод списка
3 - Добавить в начало
4 - Поиск
5 - Поиск по месяцу рождения
6 - Удаление
7 - Редактировать
8 - Сохранить в файл
9 - Считать файл
0 - Выход
Выберите действие: 2
 Иванов Сергей                        весы 11.10.1985
 Асанова Диана                        скорпион 2.11.1976

1 - Добавить в конец
2 - Вывод списка
```

Рисунок 4 – Вид окна программы после удаления элемента из списка и печати списка

На рисунке 5 продемонстрирован вид окна программы после добавления записей в начало и в конец и поиска по месяцу рождения

```
7 - Редактировать
8 - Сохранить в файл
9 - Считать файл
0 - Выход
Выберите действие: 2
 Куданова Дарья                  козерог 25.12.2002
 Кириченко Дмитрий               телец 2.5.1989
 Иванов Сергей                   весы 11.10.1985
 Асанова Диана                   скорпион 2.11.1976
 Иванов Пётр                     стрелец 5.12.1989
 Калинин Сергей                  весы 15.10.1974

1 - Добавить в конец
2 - Вывод списка
3 - Добавить в начало
4 - Поиск
5 - Поиск по месяцу рождения
6 - Удаление
7 - Редактировать
8 - Сохранить в файл
9 - Считать файл
0 - Выход
Выберите действие: 5
Месяц рождения: 5
 Кириченко Дмитрий               телец 2.5.1989

1 - Добавить в конец
2 - Вывод списка
3 - Добавить в начало
4 - Поиск
5 - Поиск по месяцу рождения
6 - Удаление
7 - Редактировать
8 - Сохранить в файл
9 - Считать файл
0 - Выход
Выберите действие: 5
Месяц рождения: 10
 Иванов Сергей                   весы 11.10.1985
 Калинин Сергей                  весы 15.10.1974
```

Рисунок 5 – Вид окна программы после добавления в начало и конец списка и поиска по месяцу рождения

На рисунке 6 продемонстрирован вид окна программы в процессе редактирования последней добавленной записи и после вывода отредактированного списка на экран. Как следует из рисунка, запись «Калинин Сергей весы 15.10.1974» была заменена на «Петров Иннокентий весы 8.10.1999».

```
6 - Удаление
7 - Редактировать
8 - Сохранить в файл
9 - Считать файл
0 - Выход
Выберите действие: 7
Введите фамилию: Калинин
Введите имя: Сергей
Введите знак зодиака (строчными буквами): весы
День рождения: 15
Месяц рождения: 10
Год рождения: 1974
Введите фамилию: Петров
Введите имя: Иннокентий
Введите знак зодиака (строчными буквами): весы
День рождения: 8
Месяц рождения: 10
Год рождения: 1999
Запись отредактирована!

1 - Добавить в конец
2 - Вывод списка
3 - Добавить в начало
4 - Поиск
5 - Поиск по месяцу рождения
6 - Удаление
7 - Редактировать
8 - Сохранить в файл
9 - Считать файл
0 - Выход
Выберите действие: 2
 Куданова Дарья                козерог 25.12.2002
 Кириченко Дмитрий             телец 2.5.1989
 Иванов Сергей                 весы 11.10.1985
 Асанова Диана                 скорпион 2.11.1976
 Иванов Пётр                   стрелец 5.12.1989
 Петров Иннокентий             весы 8.10.1999

1 - Добавить в конец
2 - Вывод списка
3 - Добавить в начало
```

Рисунок 6 – Вид окна программы после редактирования последней добавленной записи

На рисунке 7 приведен вид окна программы и содержимого файла Список.txt после ввода команды сохранения списка в этот файл и вывода информации из него.

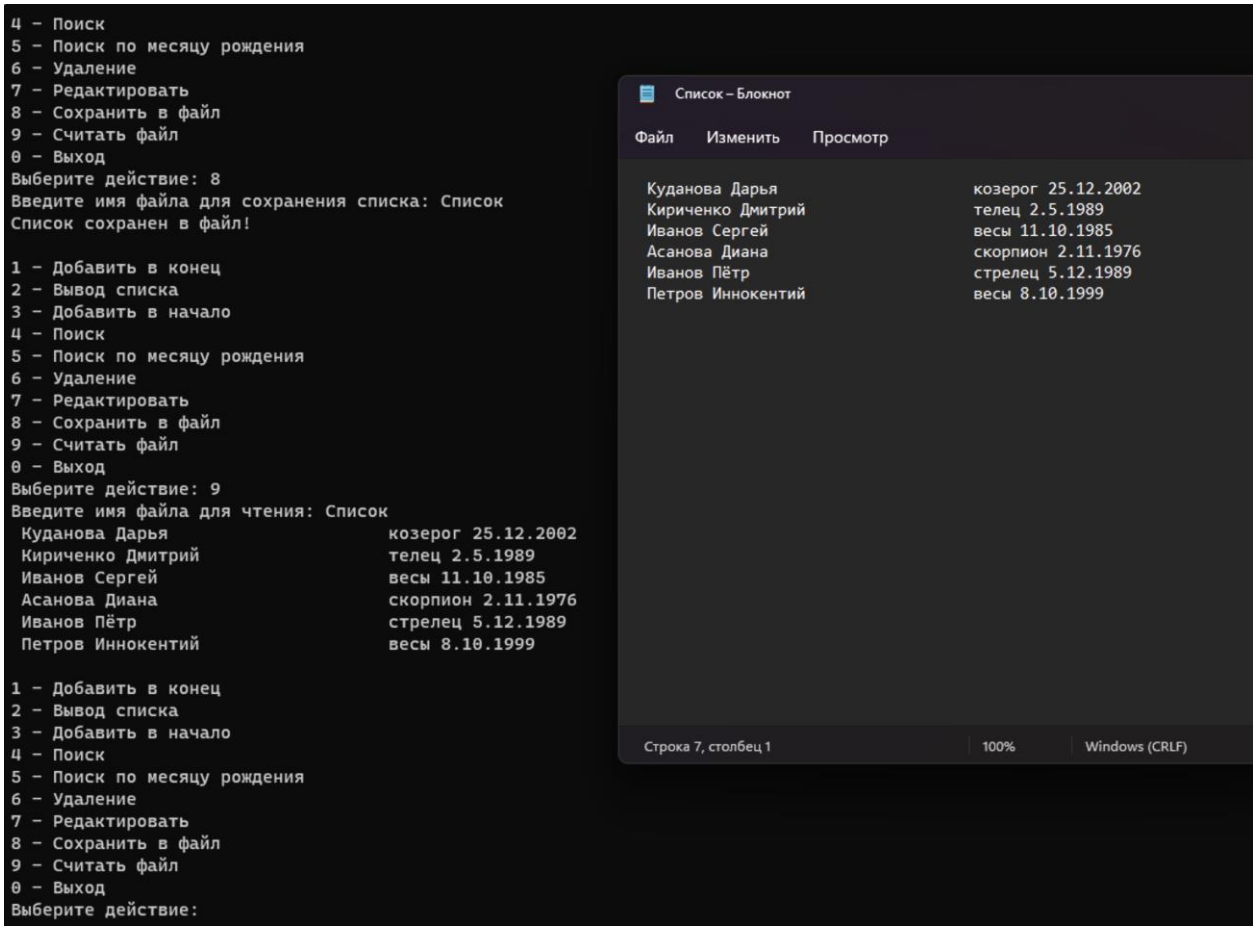

Рисунок 7 – Вид окна программы и содержимого выходного файла после сохранения списка

5. Заключение

В рамках курсовой работы была разработана программа на языке программирования C++ для предметной области «Информация о сотрудниках», выполняющая все указанные в задании функции: программа позволяет вводить информацию, хранить её в файле, осуществлять поиск, модификацию и удаление данных. Программа имеет понятный консольный

интерфейс, выводит все необходимые пояснения и подсказки, является удобной для использования.

6. Список использованной литературы

1. *Ключарев А.А., Матьяш В.А., Щекин С.В.* Структуры и алгоритмы обработки данных: Учебное пособие / СПбГУАП. СПб., 2004.

2. *Колдаев В.Д.* Основы алгоритмизации и программирования: Учебное пособие / Колдаев В.Д; Под ред. проф.Л.Г. Гагариной - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2016. - 416 с. http://znanium.com/catalog.php?bookinfo=537513

3. *Павловская Т. А.* C/C++. Программирование на языке высокого уровня: учебник. СПб. : ПИТЕР, 2007. - 461 с

4. *Кузин А. В. Чумакова Е. В.* Программирование на языке Си. М.: Форум, НИЦ ИНФРА-М, 2015. - 144 с. http://znanium.com/catalog.php?bookinfo=505194

5. Страуструп, Б. Язык программирования C++ [Текст] = The C++ Programming Language : специальное издание / Б. Страуструп ; пер.: С. Анисимов, М. Кононов ; ред.: Ф. Андреев, А. Ушаков. - [Б. м.] : Бином-Пресс, 2008. - 1098 с.

6. Кнут, Д. Искусство программирования [Текст] = The art of computer programming : [в 3 т.]. Т. 1. Основные алгоритмы / Д. Кнут ; ред. Ю. В. Козаченко. - 3-е изд. - М. : Вильямс, 2014. - 720 с.

7. Демидович, Е.М. Основы алгоритмизации и программирования. Язык СИ [Текст] : учебное пособие / Е. М. Демидович. - 2-е изд., испр. и доп. - СПб. : БХВ - Петербург, 2008. - 440 с.

8. Вирт, Н Алгоритмы и структуры данных. Новая версия для Оберона + CD [Текст] / Н. Вирт ; пер. Д. Б. Подшивалов. - 2-е изд., испр. - М. : ДМК Пресс, 2012. - 272 с.

7. Исходный код программы на языке C++

```
#define _CRT_SECURE_NO_WARNINGS

//Работа с линейным однонаправленным списком
#define _CRTDBG_MAP_ALLOC
#include <stdlib.h>
#include <crtdbg.h>

#ifdef _DEBUG
#ifndef DBG_NEW
#define DBG_NEW new ( _NORMAL_BLOCK , __FILE__ , __LINE__ )
#define newDBG_NEW
#endif
#endif

#include <locale.h>
#include <iostream>
#include <string>
#include <set>
#include <iomanip>
#include <sstream>
#include <fstream>
#include <windows.h>
#include <ctype.h>
#include <algorithm>
#include <cctype>

#include <cstdlib>
```

```
#include <fcntl.h>
#include <io.h>

using namespace std;



struct ZNAK
{
	string name; // фамилия, имя
	string zodiak; // знак зодиака
	int birthday[3]; // дата рождения
};

struct List
{
	ZNAK data; // информационное поле (данные) узла списка
	List* next; // указатель на следующий узел списка
};

// множество, содержащее названия знаков зодиака

set<string> zodiaks = { "овен", "телец", "близнецы", "рак", "лев", "дева", "весы","скорпион",
"стрелец", "козерог", "водолей", "рыбы" };

// вспомог. функция для проверки строки на пустоту
bool isEmpty(const string& str) { return str.find_first_not_of(' ') == str.npos || str.empty(); }

// вспомог. функция для вывода одной записи в поток os
void OutLine(ZNAK d, ostream& os)
{
	os << " " << left << setw(30) << d.name << setw(5) << " " << d.zodiak << " "
		<< d.birthday[0] << "." << d.birthday[1] << "." << d.birthday[2] << endl;
```

```
}

//проверка на ввод числа
int read_int() {
	int n;
	while (!(cin >> n) || (cin.peek() != '\n')) {
		cin.clear();
		while (cin.get() != '\n');
		cout << "Ошибка ввода, выберите действие: ";
	}
	return n;
}

// Редактирование списка
void UpdateElem(List** ptr, ZNAK r2)
{
	List* p = *ptr;
	p->data.birthday[0] = r2.birthday[0];
	p->data.birthday[1] = r2.birthday[1];
	p->data.birthday[2] = r2.birthday[2];
	p->data.name = r2.name;
	p->data.zodiak = r2.zodiak;
}

// Добавление элемента в конец списка
void AddElem(List** begin, List** cur, ZNAK elem)
{
	List* p = new List;
	p->data = elem; //проверка, является ли список пустым
	if (*begin == NULL)
	{
		p->next = NULL;
```

```
                *begin = p;
        }
        else
        {
                p->next = (*cur)->next;// или p->next = NULL;
                (*cur)->next = p;
        }
        *cur = p;
}

// Добавление элемента в начало списка
void AddFirstElem(List** begin, ZNAK elem)
{
        List* p = new List;
        p->data = elem; //проверка, является ли список пустым
        if (*begin == NULL)
        {
                p->next = NULL;
        }
        else
        {
                p->next = *begin;
        }
        *begin = p;
}

// Вывод элементов списка на экран
void PrintList(List* begin, ostream& os)
{
        List* p = begin;
        while (p != NULL)
        {
```

```
                OutLine(p->data, os);
                p = p->next;
        }
}

// Поиск элемента в списке по имени, знаку зодиака, дате рождения
List* FindElem(List* begin, ZNAK elem)
{
        List* p = begin;
        while (p != NULL)
        {
                if ((p->data.name.compare(elem.name) == 0) && (p->data.zodiak.compare(elem.zodiak)
== 0)
                        && (p->data.birthday[0] == elem.birthday[0]) && (p->data.birthday[1] ==
elem.birthday[1])
                        && (p->data.birthday[2] == elem.birthday[2]))
                        break;
                p = p->next;
        }
        return p;
}

// Поиск элемента в списке по месяцу рождения
List* FindElemMes(List* begin, ZNAK elem)
{
        List* p = begin;
        bool f = false;
        while (p != NULL)
        {
                if (p->data.birthday[1] == elem.birthday[1])
                {
                        f = true;
                        OutLine(p->data, cout);
```

```
                }
                p = p->next;
        }
        if (!f)
                cout << "Данные не найдены" << endl;
        return p;
}

// Удаление элемента из списка
void DelElem(List** begin, List* ptrCur)
{
        List* p;
        if (ptrCur == *begin)
        {
                // удаляем первый элемент
                *begin = (*begin)->next;
        }
        else
        {
                // устанавливаем вспомогательный указатель на элемент, предшествующий
удаляемому
                p = *begin;
                while (p->next != ptrCur)
                        p = p->next; // удаление элемента
                p->next = ptrCur->next;
        }
        delete ptrCur;
}

// Очистка памяти
void Free(List** begin)
{
```

```
	if (*begin == 0)
		return;
	List* p = *begin;
	List* t;
	while (p)
	{
		t = p;
		p = p->next;
		delete t;
	}
	*begin = NULL;
}


bool FirstLit(string F) {
	if (F[0] <= -33 && F[0] >= -64 || F[0] == -88) {
		if (F[0] <= 120 && F[0] >= 65) {
			return true;
		}
		if (F[0] <= 48 && F[0] >= 57) {
			return true;
		}
		if (F[0] == 'Ъ' || F[0] == 'ъ') {
			return true;
		}
		if (F[0] == 32) {
			return true;
		}
		else {
			return false;
		}
	}
```

```
		else {
			return true;
		}
}
bool AllLit(string F) {
		for (int i = 1; F.size(); i++) {
			if (F[i] == NULL) {
				break;
			}
			if (F[i] <= -33 && F[i] >= -64) {
				return true;
			}
			if (F[i] <= 126 && F[i] >= 33) {
				return true;
			}
			if (F[i] == 'Ъ' || F[i] == 'Ь') {
				return true;
			}
			if (F[i] == -88 || F[i] == 32) {
				return true;
			}

		}
		return false;
}


// Ввод данных - фамилия, имя, знак зодиака и дата рождения
void inputData(ZNAK& r)
{
```

```
cin.ignore(cin.rdbuf()->in_avail());
string name, surname;
cout << "Введите фамилию: ";
getline(cin, surname);
int ctr = 0; // счетчик символов в строке



while (cin.fail() || isEmpty(surname) || FirstLit(surname) || AllLit(surname))
{
	cin.clear();
	cin.ignore(cin.rdbuf()->in_avail());
	cout << "Введите значение повторно. Введите фамилию: ";
	getline(cin, surname);
}


cout << "Введите имя: ";
getline(cin, name);
while (cin.fail() || isEmpty(name) || FirstLit(name) || AllLit(name))
{
	cin.clear();
	cin.ignore(cin.rdbuf()->in_avail());
	cout << "Введите значение повторно. Введите имя: ";
	getline(cin, name);
}

r.name = surname + " " + name;
r.name = string(/*RUS*/(r.name.c_str()));
```

```
	cout << "Введите знак зодиака (строчными буквами): ";
	getline(cin, r.zodiak);
	r.zodiak = string((r.zodiak.c_str()));
	while (zodiaks.find(r.zodiak) == zodiaks.end())
	{
		cin.clear();
		cin.ignore(cin.rdbuf()->in_avail());
		cout << "Введите значение повторно. Введите знак зодиака (строчными буквами): ";
		getline(cin, r.zodiak);
		r.zodiak = string((r.zodiak.c_str()));
	}


	cout << "День рождения: ";
	cin >> r.birthday[0];
	while (cin.fail() || r.birthday[0] < 1 || r.birthday[0] > 31)
	{
		cin.clear();
		cin.ignore(cin.rdbuf()->in_avail());
		cout << "Введите значение повторно. День рождения: ";
		cin >> r.birthday[0];
	}

	//общее количетсво месяцев
	cout << "Месяц рождения: ";
	cin >> r.birthday[1];
	while (cin.fail() || (r.birthday[1] < 1 || r.birthday[1] > 12) ||
		((r.birthday[1] == 4 || r.birthday[1] == 6 || r.birthday[1] == 9 || r.birthday[1] == 10) &&
r.birthday[0] > 30) ||
		(r.birthday[1] == 2 && r.birthday[0] > 29))
	{
		if (r.birthday[1] < 1 || r.birthday[1] > 12)
```

```
		{
			cin.clear();
			cin.ignore(cin.rdbuf()->in_avail());
			cout << "Введите значение повторно. Месяц рождения: ";
			cin >> r.birthday[1];
		}
		if ((r.birthday[1] == 4 || r.birthday[1] == 6 || r.birthday[1] == 9 || r.birthday[1] == 10)
&& r.birthday[0] > 30)
		{
			cin.clear();
			cin.ignore(cin.rdbuf()->in_avail());
			cout << "В этом месяце нет 31 дня. Введите значение повторно. Месяц
рождения: ";
			cin >> r.birthday[1];
		}
		if (r.birthday[1] == 2 && r.birthday[0] > 29)
		{
			cin.clear();
			cin.ignore(cin.rdbuf()->in_avail());
			cout << "В феврале только 28(29) дней(. Введите значение повторно. Месяц
рождения: ";
			cin >> r.birthday[1];
		}
	}


	cout << "Год рождения: ";
	cin >> r.birthday[2];
	while (cin.fail() || (r.birthday[2] < 1900 || r.birthday[2] > 2022) ||
		(r.birthday[2] % 4 != 0 && r.birthday[1] == 2 && r.birthday[0] == 29))
	{
		if (r.birthday[2] < 1900 || r.birthday[2] > 2022)
		{
```

```
                cin.clear();
                cin.ignore(cin.rdbuf()->in_avail());
                cout << "Введите значение повторно. Год рождения: ";
                cin >> r.birthday[2];
            }
            if (r.birthday[2] % 4 != 0 && r.birthday[1] == 2 && r.birthday[0] == 29)
            {
                cin.clear();
                cin.ignore(cin.rdbuf()->in_avail());
                cout << "Введён не високосный год, Вы указали дату 29.02, следовательно год должен быть високосным." << endl;
                cout << "Введите значение повторно. Год рождения: ";
                cin >> r.birthday[2];
            }
        }
        //соответсвие знаку зодиака
        if ((r.birthday[1] == 3 && r.birthday[0] >= 21 && r.zodiak != "овен" || r.birthday[1] == 4 && r.birthday[0] <= 19 && r.zodiak != "овен") ||
            (r.birthday[1] == 4 && r.birthday[0] >= 20 && r.zodiak != "телец" || r.birthday[1] == 5 && r.birthday[0] <= 20 && r.zodiak != "телец") ||
            (r.birthday[1] == 5 && r.birthday[0] >= 21 && r.zodiak != "близнецы" || r.birthday[1] == 6 && r.birthday[0] <= 21 && r.zodiak != "близнецы") ||
            (r.birthday[1] == 6 && r.birthday[0] >= 22 && r.zodiak != "рак" || r.birthday[1] == 7 && r.birthday[0] <= 22 && r.zodiak != "рак") ||
            (r.birthday[1] == 7 && r.birthday[0] >= 23 && r.zodiak != "лев" || r.birthday[1] == 8 && r.birthday[0] <= 22 && r.zodiak != "лев") ||
            (r.birthday[1] == 8 && r.birthday[0] >= 23 && r.zodiak != "дева" || r.birthday[1] == 9 && r.birthday[0] <= 22 && r.zodiak != "дева") ||
            (r.birthday[1] == 9 && r.birthday[0] >= 23 && r.zodiak != "весы" || r.birthday[1] == 10 && r.birthday[0] <= 23 && r.zodiak != "весы") ||
            (r.birthday[1] == 10 && r.birthday[0] >= 24 && r.zodiak != "скорпион" || r.birthday[1] == 11 && r.birthday[0] <= 22 && r.zodiak != "скорпион") ||
            (r.birthday[1] == 11 && r.birthday[0] >= 23 && r.zodiak != "стрелец" || r.birthday[1] == 12 && r.birthday[0] <= 21 && r.zodiak != "стрелец") ||
            (r.birthday[1] == 12 && r.birthday[0] >= 22 && r.zodiak != "козерог" || r.birthday[1] == 1 && r.birthday[0] <= 20 && r.zodiak != "козерог") ||
```

```
		(r.birthday[1] == 1 && r.birthday[0] >= 21 && r.zodiak != "водолей" || r.birthday[1] == 2
&& r.birthday[0] <= 18 && r.zodiak != "водолей") ||
		(r.birthday[1] == 2 && r.birthday[0] >= 19 && r.zodiak != "рыбы" || r.birthday[1] == 3
&& r.birthday[0] <= 20 && r.zodiak != "рыбы"))

		cout << "Месяц и знак зодиака не соответствуют...\n";
}

// Ввод данных - месяц
void inputDataMes(ZNAK& r)
{
	//общее количетсво месяцев
	cout << "Месяц рождения: ";
	cin >> r.birthday[1];
	while (cin.fail() || r.birthday[1] < 1 || r.birthday[1] > 12)
	{
		cin.clear();
		cin.ignore(cin.rdbuf()->in_avail());
		cout << "Введите значение повторно. Месяц рождения: ";
		cin >> r.birthday[1];
	}
}


int main(int argc, char** argv)
{

	(LC_CTYPE, "Russian");
	SetConsoleCP(1251);
	SetConsoleOutputCP(1251);

	List* head = NULL;
	List* cur = NULL;
```

```
int n = -1;
ZNAK r;



// Меню пользователя
while (n != 0)
{
	cout << endl << "1 - Добавить в конец" << endl <<
		"2 - Вывод списка" << endl <<
		"3 - Добавить в начало" << endl <<
		"4 - Поиск" << endl <<
		"5 - Поиск по месяцу рождения" << endl <<
		"6 - Удаление" << endl <<
		"7 - Редактировать" << endl <<
		"8 - Сохранить в файл" << endl <<
		"9 - Считать файл" << endl <<
		"0 - Выход\nВыберите действие: ";



	cin.clear();
	cin.ignore(cin.rdbuf()->in_avail());
	cin >> n;
	while (cin.fail() || (n < 0) || (n > 9))
	{
		if ((n < 0) || (n > 9))
		{
			cin.clear();
			cin.sync();
			cout << "Ошибка ввода, выберите действие: ";
			cin >> n;
		}
```

```
        n = read_int();
}


switch (n)
{
case 1:
{
        inputData(r);
        AddElem(&head, &cur, r);
        break;
}


case 2:
{
        if (head)
                PrintList(head, cout);
        else
                cout << "Нет данных!" << endl;
        break;
}


case 3:
{
        inputData(r);
        AddFirstElem(&head, r);
        break;
}
```

```
case 4:
{
	if (!head)
	{
		cout << "Нет данных!" << endl;
		break;
	}
	List* ptr;
	inputData(r);
	ptr = FindElem(head, r);
	if (ptr == NULL)
		cout << "Запись не найдена!" << endl;
	else OutLine(ptr->data, cout);
	break;
}

case 5:
{
	if (!head)
	{
		cout << "Нет данных!" << endl;
		break;
	}
	List* ptr;
	inputDataMes(r);
	ptr = FindElemMes(head, r);
	break;
}

case 6:
```

```
{
	if (!head)
	{
		cout << "Нет данных!" << endl;
		break;
	}
	List* ptr;
	inputData(r);
	ptr = FindElem(head, r);
	if (ptr == NULL)
		cout << "Запись не найдена!" << endl;
	else
	{
		DelElem(&head, ptr);
		cout << "Запись удалена!" << endl;
	};
	break;
}



case 7:
{
	if (!head)
	{
		cout << "Нет данных!" << endl;
		break;
	}
	List* ptr;
	inputData(r);
	ptr = FindElem(head, r);
	if (ptr == NULL)
		cout << "Запись не найдена!" << endl;
```

```
                else
                {
                        ZNAK r2;
                        inputData(r2);
                        UpdateElem(&ptr, r2);
                        cout << "Запись отредактирована!" << endl;
                };
                break;
        }


        case 8:
        {
                string fname;
                cin.clear();
                cin.ignore(cin.rdbuf()->in_avail());
                cout << "Введите имя файла для сохранения списка: ";
                getline(cin, fname);
                ofstream file;
                file.open(fname);
                while (!file.is_open())
                {
                        cin.clear();
                        cin.ignore(cin.rdbuf()->in_avail());
                        cout << "Невозможно открыть файл для сохранения списка! Введите
имя файла для сохранения списка: ";
                        getline(cin, fname);
                }
                if (head)
                {
                        PrintList(head, file);
                        cout << "Список сохранен в файл!" << endl;
```

```
		}
		else
			file << "Нет данных!" << endl;
		file.close();
		break;
	}


	case 9:
	{
		string fname;
		cin.clear();
		cin.ignore(cin.rdbuf()->in_avail());
		cout << "Введите имя файла для чтения: ";
		getline(cin, fname);

		ifstream file;
		file.open(fname);
		while (!file.is_open())
		{
			cin.clear();
			cin.ignore(cin.rdbuf()->in_avail());
			cout << "Невозможно открыть файл! Введите имя файла: ";
			getline(cin, fname);
			file.open(fname);
		}
		if (file.is_open())
		{
			struct stat t_stat;
			stat(fname.c_str(), &t_stat);
			struct tm* timeinfo = localtime(&t_stat.st_ctime); // or gmtime()
depending on what you want
```

```
                                char mod[100];
                                ctime_s(mod, 100, &t_stat.st_mtime);
                                string create = asctime(timeinfo);
                                string edit = mod;
                                if (create != edit)
                                {
                                        cout << "Файл был изменён после создания." << "\nДата
создания: " << create << "Дата изменения: " << edit << endl;
                                }

                                while (getline(file, fname))
                                {
                                        cout << fname << endl;
                                }
                        }
                        file.close();
                        break;
                }
                }
        }


        Free(&head);
        _CrtSetReportMode(_CRT_WARN, _CRTDBG_MODE_FILE);
        _CrtSetReportFile(_CRT_WARN, _CRTDBG_FILE_STDOUT);
        _CrtSetReportMode(_CRT_ERROR, _CRTDBG_MODE_FILE);
        _CrtSetReportFile(_CRT_ERROR, _CRTDBG_FILE_STDOUT);
        _CrtSetReportMode(_CRT_ASSERT, _CRTDBG_MODE_FILE);
        _CrtSetReportFile(_CRT_ASSERT, _CRTDBG_FILE_STDOUT);
        system("pause");
        return 0;
}
```