

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Дальневосточный федеральный университет»**

(ДВФУ)

**ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**

**(ШКОЛА)**

**Департамент программной инженерии и искусственного интеллекта**

РУДЕНКО ДАРЬЯ СТАНИСЛАВНА

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ РАБОТЫ СО СПРАВОЧНИКОМ ПРЕДМЕТНОЙ ОБЛАСТИ «ПРЕПОДАВАТЕЛЬСКИЙ СОСТАВ»

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Фундаментальные структуры данных и алгоритмы»

по образовательной программе подготовки бакалавров по направлению

09.03.04 «Программная инженерия», профиль «Программная инженерия»

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |  |  | Студент группы Б9122-09.03.04 | | | | | | | |
|  |  |  | | | | Руденко Д.С. | | | |
|  | | | | | | |  |  | (подпись) | | | |  | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | |  |  | Руководитель | | | | | | | |
|  | | | | | | |  |  | ст. преподаватель ДПИиИИ | | | | | | | |
|  | | | | | | |  |  | ученая степень, должность | | | | |  | Крестникова О.А. | |
|  | | | | | | |  |  | (подпись) | | | | |  | (ФИО) | |
|  | | | | | | | | | | | | | | | | |
| Регистрационный № | | | | | | |  |  | Защищен с оценкой | | | | | | | |
|  | | | | | | |  |  |  | | | | | | | |
|  | | | |  |  | |  |  | « |  | » |  |  | | | 2024 г. |
| (подпись) | | | |  | (ФИО) | |  |  |  | | | | | | | |
| « |  | » |  | | | 2024 г. |  |  |  | | | | | | | |

г. Владивосток

2024

# Оглавление

[Оглавление 2](#_Toc172297159)

[Введение 3](#_Toc172297160)

[1 Анализ предметной области 4](#_Toc172297161)

[1.1 Объект предметной области 4](#_Toc172297162)

[1.2 Законы ПО 5](#_Toc172297163)

[2 Теоретическая часть 6](#_Toc172297164)

[2.1 Хеш-таблица 6](#_Toc172297165)

[2.1.1 Хеш-функция 7](#_Toc172297166)

[2.1.2 Разрешение коллизий методом цепочек 8](#_Toc172297167)

[2.1.3 Цепочка – двусвязный список 8](#_Toc172297168)

[3 Требования к информационной системе 9](#_Toc172297169)

[3.1 Требования к данным 9](#_Toc172297170)

[3.1.1 Требования к входным данным 9](#_Toc172297171)

[3.1.2 Требования к выходным данным 10](#_Toc172297172)

[3.2 Функциональные требования 10](#_Toc172297173)

[3.2.1 Общие требования 10](#_Toc172297174)

[4 Реализация 12](#_Toc172297175)

[4.1 Спецификация структур данных 12](#_Toc172297176)

[4.2 Описание среды разработки 19](#_Toc172297179)

[4.3 Руководство пользователя 19](#_Toc172297180)

[4.3.1 Работа со справочником «Преподавательский состав» 19](#_Toc172297181)

[4.4 Тестирование 22](#_Toc172297182)

[Заключение 29](#_Toc172297183)

[Список литературы 30](#_Toc172297184)

# Введение

В современном мире в любой организации необходим учёт сотрудников. Так, в учебных заведениях для эффективной работы нужно разработать информационную систему, способную хранить в себе сведения о составе преподавателей ВУЗа. Одним из ключевых шагов к достижению этой цели является создание справочника для сбора и хранения информации о преподавателях.

Целью курсового проекта является: разработка информационной системы для работы со справочником предметной области «Преподавательский состав».

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Провести анализ предметной области «Преподавательский состав».

2. Изучить теоретические основы методов построения справочников.

3. Определить требования к информационной системе.

4. Реализовать информационную систему и провести тестирование.

# 1 Анализ предметной области

Требуется разработать информационную систему для работы со справочником предметной области (ПО) «Преподавательский состав».

Система должна решать следующие задачи:

1. хранить, позволять просматривать, добавлять и удалять информацию о преподавателях: код направления, название дисциплины, ФИО, дата;
2. позволять искать информацию о преподавателях по коду направления, дате и названию дисциплины.
3. отображать внутреннее представление структур данных.

1.1 Объект предметной области

Исходя из задач, которые должна решать информационная система, выделен объект ПО, представленный в Таблице 1.

Таблица 1 – Объект ПО

|  |  |
| --- | --- |
| Тип и название объекта | Описание объекта |
| Справочник «Преподавательский состав» | Хранит информацию о каждом преподавателе |

Информация о каждом преподавателе из справочника характеризуется следующими параметрами: код направления, ФИО, название дисциплины, ФИО, дата.

**ФИО преподавателя** – аббревиатура, расшифровывается как: фамилия, имя, отчество. Фамилия, имя и отчество являются словами, состоящими из букв латинского алфавита, первая буква каждого слова – заглавная, остальные – строчные. Слова разделены одним пробелом. Отчество не пустое. В каждом слове минимум две буквы.

**Название дисциплины** – строка, состоящая из набора строчных и заглавных букв латинского алфавита, может содержать в себе как одно, так и множество слов, разделенных пробелом.

**Код направления** – три целых числа, разделенные точками. Если число однозначное, то перед ним стоит 0.

**Дата –** строка вида: DD.MM.YY, где DD – число от 1 до 31, MM – число от 1 до 12, YY – четырехзначное число от 1990 до 2024

Пример справочника «Экипаж» представлен в таблице 2.

Таблица 2 – Справочник «Преподавательский состав»

|  |  |  |  |
| --- | --- | --- | --- |
| **Код направления** | **Название дисциплины** | **ФИО преподавателя** | **Дата** |
| 09.03.04 | Matanaliz | Ivanov Ivan Ivanovich | 28.05.2024 |
| 01.03.04 | Theory veroyatnosti | Petrov Pavel Alexandrovich | 28.05.2024 |
| 09.03.04 | Fizra | Semenov Kirill Evgenevich | 05.05.2023 |
| 06.03.01 | Biology | Kozlov Dmitrii Vladimirovich | 16.05.2024 |
| 10.03.01 | IT technologies | Nikitin Alexandr Igorevich | 28.04.2024 |
| 11.03.01 | Radiotechnique | Vasileva Olga Petrovna | 25.04.2024 |
| 11.03.02 | Infocommunication technologies | Kuznetsov Sergey Victorovich | 22.04.2000 |
| 07.03.04 | Pochvovedenie | Popova Maria Alexeevna | 19.04.2003 |
| 06.03.02 | Biology | Sokolov Igor Nikolaevich | 13.05.2008 |
| 05.03.06 | Ecology | Mihailova Elena Andreevna | 10.05.2024 |

1.2 Законы ПО

Комбинация кода направления и даты уникальна. Одной дате может соответствовать единственный код направления.

# 2 Теоретическая часть

Для хранения данных справочника будут использоваться следующие структуры данных: статическая хеш-таблица с цепочкой из двусвязного списка. Дата и код направления уникальны.

2.1 Хеш-таблица

Хеш-таблица – структура данных для хранения пар ключей и их значений. Она представляет собой массив, где местоположение элемента зависит от значения самого элемента. Связь между значением элемента и его позицией в хеш-таблице задаёт хеш-функция. Важное свойство хеш-таблицы: поиск, вставка и удаление элементов из таблицы выполняются за фиксированное время О(1). [3]

Статическая реализация хеш-таблицы предполагает, что размер массива индексов в ней не изменяется, поэтому при добавлении большого количества элементов, вычислительная сложность операций может повыситься до O(n), где n – количество элементов в структуре данных. [1]

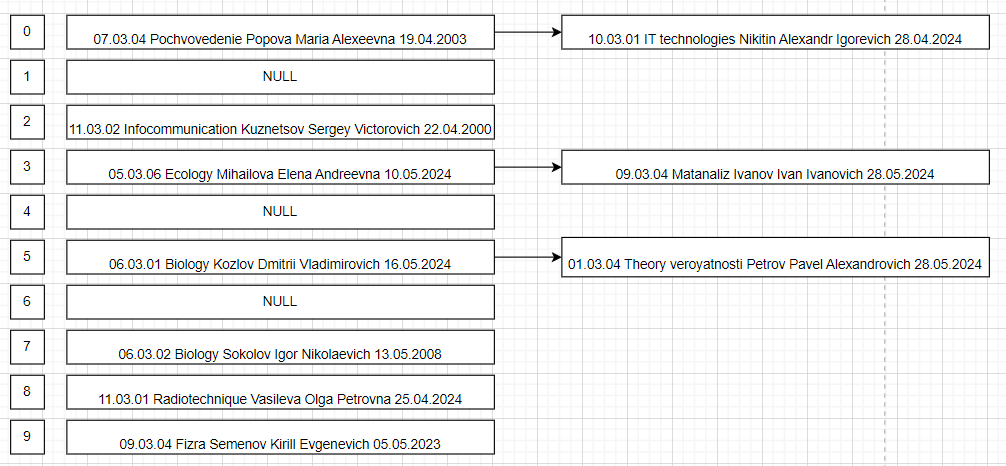


Рисунок 1 – Пример хеш-таблицы размерностью 10

2.1.1 Хеш-функция

Хеш-функция – это функция, которая принимает в качестве аргумента какой-то элемент (который нужно вставить в хеш-таблицу), а в результате выдает позицию заданного элемента в хеш-таблице. При многократном вызове хеш-функции с одинаковыми данными она всегда будет возвращать одно и то же число. [3]

В курсовом проекте будет использоваться метод модульного хеширования. Это наиболее простой и эффективный метод – один из наиболее часто используемых методов хеширования – выбор в качестве размера M таблицы простого числа и вычисление остатка от деления k на M, или h(k) = k mod M для любого целочисленного ключа k. Такая функция называется **модульной хеш-функцией**. Её очень просто вычислить (k % M в языке C++), и она эффективна для достижения равномерного распределения значений ключей между значениями, которые меньше M. [2]

Пример:

Размер хеш-таблицы = 10 = k:

1. Хеш 09.03.04 + 28.05.2024 mod k = 1
2. Хеш 01.03.04 + 28.05.2024 mod k = 0
3. Хеш 09.03.04 + 05.05.2023 mod k = 5
4. Хеш 06.03.01 + 16.05.2024 mod k = 4
5. Хеш 10.03.01 + 28.04.2024 mod k = 5
6. Хеш 11.03.01 + 25.04.2024 mod k = 3
7. Хеш 11.03.02 + 22.04.2000 mod k = 5
8. Хеш 07.03.04 + 19.04.2003 mod k = 3
9. Хеш 06.03.02 + 13.05.2008 mod k = 0
10. Хеш 05.03.06 + 10.05.2024 mod k = 2

В примере (2, 9), (3, 5, 7), (6, 8) хеши получились одинаковыми. В данном случае используется метод разрешения коллизии путем цепочек.

2.1.2 Разрешение коллизий методом цепочек

При использовании данного метода мы связываем все элементы, хешированные в одну и ту же ячейку, в цепочку-список. При возникновении коллизии элемент добавляется в связный список.

Добавляя элементы с одинаковым хешем, они связываются в одну цепочку. На рисунке 1 под индексами 0, 3, 5 за счёт коллизий образовались цепочки, содержащие по два элемента.

Удаляя элемент из цепочки, список уменьшается. Приведём пример удаления из исходной таблицы элементов под индексами 0 и 5 (рисунок 2).

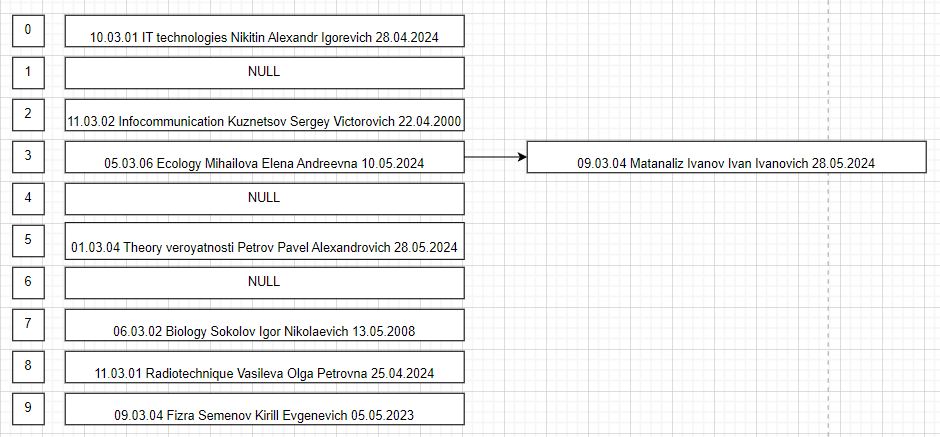


Рисунок 2 – Удаление элемента из цепочки

2.1.3 Цепочка – двусвязный список

Цепочка – двусвязный список с добавлением в начало.

Двусвязный список – это список, где каждый элемент ссылается на следующий и предыдущий. Первый и последний соответственно не имеют ссылок на предыдущий и следующий.

Добавим в таблицу два элемента с хешами 5, образуется двусвязный список - цепочка из трех элементов:



Рисунок 3 – Цепочка

# 3 Требования к информационной системе

В данной главе описываются требования к информационной системе для работы со справочником, а именно: требования к входным данным системы, требования к ее выходным данным и требования к ее функционалу.

3.1 Требования к данным

3.1.1 Требования к входным данным

Основываясь на анализе ПО, входными данными является:

* текстовый файл input.txt, каждая строка файла содержит информацию о преподавателе, а именно код направления, название дисциплины, ФИО, дата, разделенные пробелом;

Пример текстового файла:

09.03.04 Matanaliz Ivanov Ivan Ivanovich 28.05.2024

01.03.04 Theory veroyatnosti Petrov Pavel Alexandrovich 28.05.2024

09.03.04 Fizra Semenov Kirill Evgenevich 05.05.2023

06.03.01 Biology Kozlov Dmitrii Vladimirovich 16.05.2024

10.03.01 IT technologies Nikitin Alexandr Igorevich 28.04.2024

11.03.01 Radiotechnique Vasileva Olga Petrovna 25.04.2024

11.03.02 Infocommunication Kuznetsov Sergey Victorovich 22.04.2000

07.03.04 Pochvovedenie Popova Maria Alexeevna 19.04.2003

06.03.02 Biology Sokolov Igor Nikolaevich 13.05.2008

05.03.06 Ecology Mihailova Elena Andreevna 10.05.2024

* размер хеш-таблицы;
* код направления;
* название дисциплины;
* ФИО;
* дата.

3.1.2 Требования к выходным данным

Выходными данными являются:

* текстовый файл output.txt, каждая строка файла содержит информацию об объекте преподаватель, а именно код направления, название дисциплины, ФИО, дата, разделенные пробелом;

сообщения об ошибках:

* Размер хеш-таблицы не задан
* Ошибка добавления
* Ошибка удаления
* Вы уже задавали размер хеш-таблицы
* Файл не удалось записать
* Данные некорректны
* Неверные данные

3.2 Функциональные требования

3.2.1 Общие требования

Информационная система должна позволять:

1. Считывать данные для справочника из текстового файла. При считывании проверяются полнота и корректность данных;
2. Сохранять данные справочника в текстовый файл;
3. Просматривать данные справочника;
4. Добавлять информацию в справочник по значениям: код направления, название дисциплины, ФИО, дата. Для функции добавления проверяются следующие условия: проверка правильности формата даты, проверка правильности формата ФИО, проверка на уникальность данных.
5. Удалять информацию о преподавателе из справочника по коду направления, названию дисциплины, ФИО и дате. При удалении данные проверяются на корректность, удаление происходит полном совпадении введенных данных в справочнике;
6. Находить информацию по значениям: дата, код направления, название дисциплины.
7. Использовать окно отладки, в котором происходит вывод структур данных, а именно: хеш-таблицы и двусвязного списка.
8. Сохранять имеющийся справочник в файл.

# 4 Реализация

В данной главе содержится спецификация реализованных структур данных, описание графического интерфейса, а также приведены результаты тестирования.

4.1 Спецификация структур данных

**Класс Data** - класс, описывающий информацию о студенте по конкретному предмету.

**Поля:**

* fio: объект класса Fio, содержащий фамилию, имя и отчество студента.
* code: объект класса Code, представляющий уникальный код студента.
* nameDiscipline: строка, содержащая название учебной дисциплины.
* date: объект класса Date, хранящий дату события (например, экзамена или сдачи работы).

**Методы:**

* Конструктор Data: создает объект класса ‘Data’ и инициализирует его поля значениями, переданными в качестве параметров

Параметры:

* Fio(тип fio) – ФИО человека;
* Code(тип code) – код паспорта;
* Discipline (тип string) – название дисциплины;
* Date (тип date) - дата

**Класс Date** - класс, описывающий дату.

**Поля:**

* m\_year: короткое целое число, представляющее год.
* m\_month: короткое целое число, представляющее месяц.
* m\_day: короткое целое число, представляющее день.

**Методы:**

* Конструктор Date(): Инициализирует объект с нулевыми значениями для года, месяца и дня.

Входные данные: нет.

Выходные данные: нет.

* Конструктор Date(const short \_year, const short \_month, const short \_day): Инициализирует объект с заданными годом, месяцем и днем.

Входные данные: значения года, месяца и дня.

Выходные данные: нет.

* Конструктор Date(const string &date): Инициализирует объект, парся строку в формате "dd.mm.yyyy".

Входные данные: строка даты.

Выходные данные: нет. Выводит сообщение об ошибке в случае некорректного формата даты.

* Деструктор ~Date(): Уничтожает объект Date.

Входные данные: нет.

Выходные данные: нет.

* Метод toQString() const: Преобразует дату в строку формата "dd.mm.yyyy".

Входные данные: нет.

Выходные данные: строка даты в формате "dd.mm.yyyy".

* Метод correctDate() const: Проверяет корректность даты.

Входные данные: нет.

Выходные данные: логическое значение, показывающее корректность даты.

* short getYear() const: Возвращает год.
* short getMonth() const: Возвращает месяц.
* short getDay() const: Возвращает день.

**Класс Fio** - класс, описывающий фамилию, имя и отчество.

**Поля:**

* lastName: строка, представляющая фамилию.
* firstName: строка, представляющая имя.
* patronymic: строка, представляющая отчество.

**Методы:**

* Конструктор Fio(QString \_lastName, QString \_firstName, QString \_patronymic): Инициализирует объект с заданными фамилией, именем и отчеством.

Входные данные: строковые значения фамилии, имени и отчества.

Выходные данные: нет.

* Метод getFIO() const: Возвращает полное ФИО в формате "Фамилия Имя Отчество".

Входные данные: нет.

Выходные данные: строка, содержащая полное ФИО.

* Метод setFIO(QString \_lastName, QString \_firstName, QString \_patronymic): Устанавливает значения фамилии, имени и отчества.

Входные данные: строковые значения фамилии, имени и отчества.

Выходные данные: нет.

* Метод getLastname() const: Возвращает фамилию.

Входные данные: нет.

Выходные данные: строка, содержащая фамилию.

* Метод getFirstname() const: Возвращает имя.

Входные данные: нет.

Выходные данные: строка, содержащая имя.

* Метод getPatronymic() const: Возвращает отчество.

Входные данные: нет.

Выходные данные: строка, содержащая отчество.

**Структура HashTable** описывает хеш-таблицу для хранения записей типа Data.

### **Поля:**

* vector<LinkedList\*> table – вектор указателей на объекты LinkedList, используемый как хеш-таблица.

### **Методы:**

#### HashTable(int size) - Конструктор, инициализирующий хеш-таблицу заданного размера.

**Входные данные**: размер таблицы**Формальные параметры**: size – размер хеш-таблицы

**Выходные данные**: -

* bool insert(Data record) - функция для вставки записи в хеш-таблицу.

**Входные данные**: структура HashTable

**Формальные параметры**: record – запись типа Data, которую необходимо вставить

**Выходные данные**: логическое значение, указывающее успешность операции

#### bool remove(Data record) - функция для удаления записи из хеш-таблицы.

**Входные данные**: структура HashTable

**Формальные параметры**: record – запись типа Data, которую необходимо удалить

**Выходные данные**: логическое значение, указывающее успешность операции

* int find(Data record) - функция для поиска записи в хеш-таблице.

**Входные данные**: структура HashTable

**Формальные параметры**: record – запись типа Data, которую необходимо найти

**Выходные данные**: индекс записи в хеш-таблице или -1, если запись не найдена

* unsigned hash\_function(Data record) - функция для вычисления хеш-значения записи.

#### **Входные данные**: структура HashTable

#### **Формальные параметры**: record – запись типа Data, для которой вычисляется хеш-значение

#### **Выходные данные**: хеш-значение записи

#### vector<Data> print() const - функция для получения всех записей из хеш-таблицы в виде вектора.

**Входные данные**: структура HashTable

**Формальные параметры**: -

**Выходные данные**: вектор записей типа Data

* void print(QPlainTextEdit \*text\_edit) const - функция для вывода содержимого хеш-таблицы в виджет QPlainTextEdit.

**Входные данные**: структура HashTable

**Формальные параметры**: text\_edit – указатель на объект QPlainTextEdit, в который выводится содержимое хеш-таблицы

**Выходные данные**: -

* void write\_file(ofstream &file) const - функция для записи содержимого хеш-таблицы в файл.

**Входные данные**: структура HashTable

**Формальные параметры**: file – выходной файловый поток

**Выходные данные**:

**Структура Node** описывает узел, который содержит запись и ссылки на предыдущий и следующий узлы.

#### Поля:

* Node\* prev – указатель на предыдущий узел
* Node\* next – указатель на следующий узел
* Data record – данные, хранимые в узле

Методы:

##### Node() - конструктор по умолчанию.

##### **Входные данные**: -

##### **Формальные параметры**: -

##### **Выходные данные**: -

##### Node(Data record) - конструктор, инициализирующий узел данными.

##### **Входные данные**: структура Node

##### **Формальные параметры**: record – данные типа Data

##### **Выходные данные**: -

**Структура LinkedList** описывает связанный список, состоящий из узлов типа Node.

**Поля**: Node\* head – указатель на головной узел списка

**Методы:**

LinkedList() - конструктор по умолчанию.

**Входные данные**: -

**Формальные параметры**: -

**Выходные данные**: -

* LinkedList(Node\* node) - конструктор, инициализирующий список с заданным головным узлом.

##### **Входные данные**: структура LinkedList

##### **Формальные параметры:** node – указатель на головной узел типа Node

##### **Выходные данные**: -

##### bool add(Data record) - функция для добавления записи в список.

##### **Входные данные**: структура LinkedList

##### **Формальные параметры:** record – данные типа Data, которые необходимо добавить

##### **Выходные данные**: логическое значение, указывающее успешность операции

##### bool remove(Data record) - функция для удаления записи из списка.

##### **Входные данные**: структура LinkedList

##### **Формальные параметры**: record – данные типа Data, которые необходимо удалить

##### **Выходные данные**: логическое значение, указывающее успешность операции

##### int find(Data record) - функция для поиска записи в списке.

##### **Входные данные**: структура LinkedList

**Формальные параметры:** record – данные типа Data, которые необходимо найти

**Выходные данные**: индекс записи в списке или -1, если запись не найдена

##### void print(vector<Data>& records) - функция для добавления всех записей из списка в вектор.

**Входные данные**: структура LinkedList

**Формальные параметры**: records – вектор записей типа Data, в который добавляются все записи из списка

**Выходные данные:** -

##### void print(QPlainTextEdit\* text\_edit) - функция для вывода содержимого списка в виджет QPlainTextEdit.

##### **Входные данные**: структура LinkedList

##### **Формальные параметры**: text\_edit – указатель на объект QPlainTextEdit, в который выводится содержимое списка

##### **Выходные данные**: -

4.2 Описание среды разработки

Для реализации информационной системы была выбрана среда разработки Qt Creator. Язык реализации – C++.

4.3 Руководство пользователя

При запуске приложения открывается окно с полями для записи размера хеш-таблицы, кода направления, названия дисциплины, ФИО, даты. Изображено на рисунке 1.

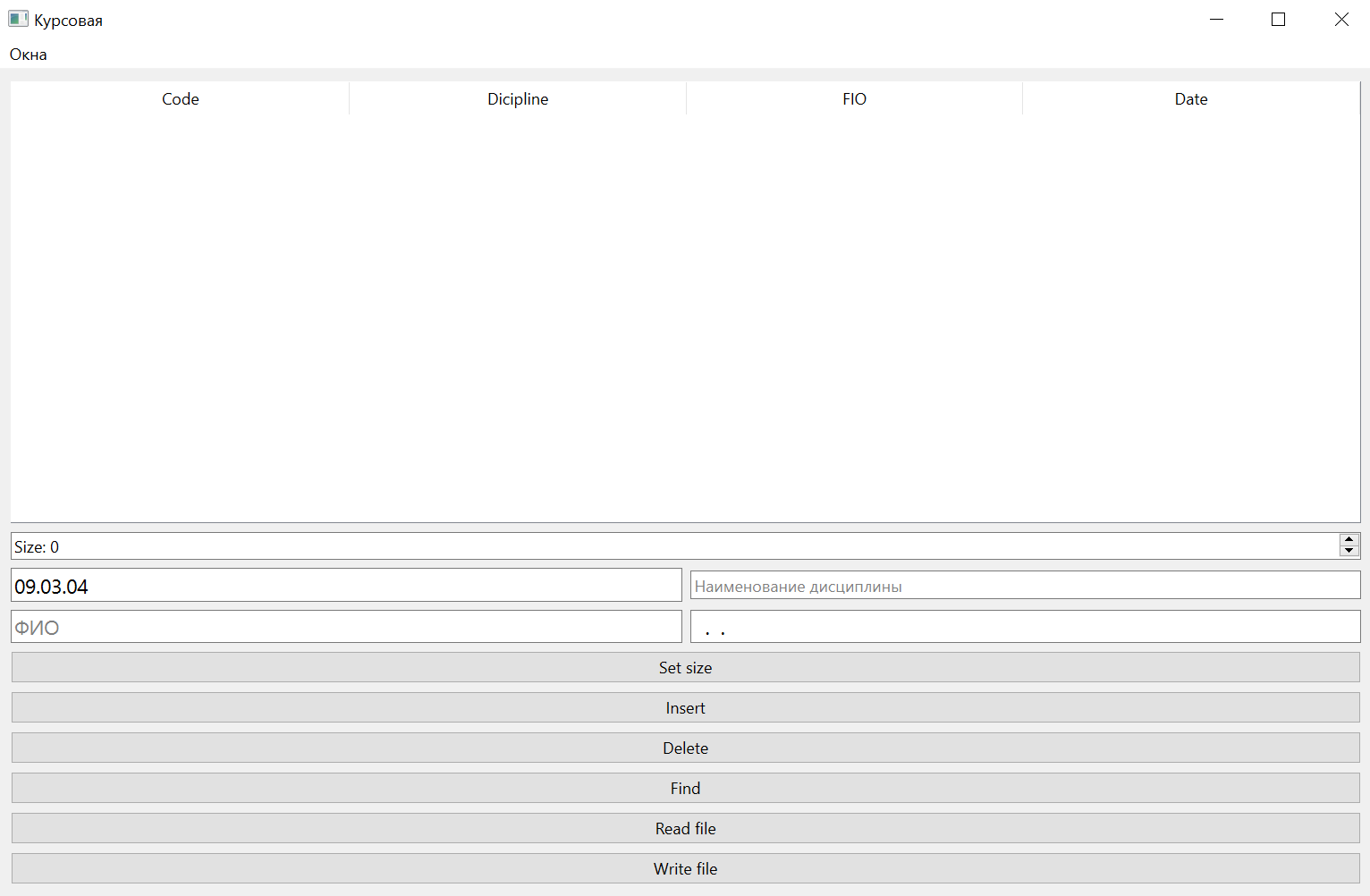


Рисунок 4 – Начальное окно

4.3.1 Работа со справочником «Преподавательский состав»

В главном меню, изображенном на рисунке 4, можно определить размер хеш-таблицы, нажав на кнопку «Set size». Кнопка «Insert» позволяет вставить в хеш-таблицу данные, введенные в пустые поля (рисунок 5).

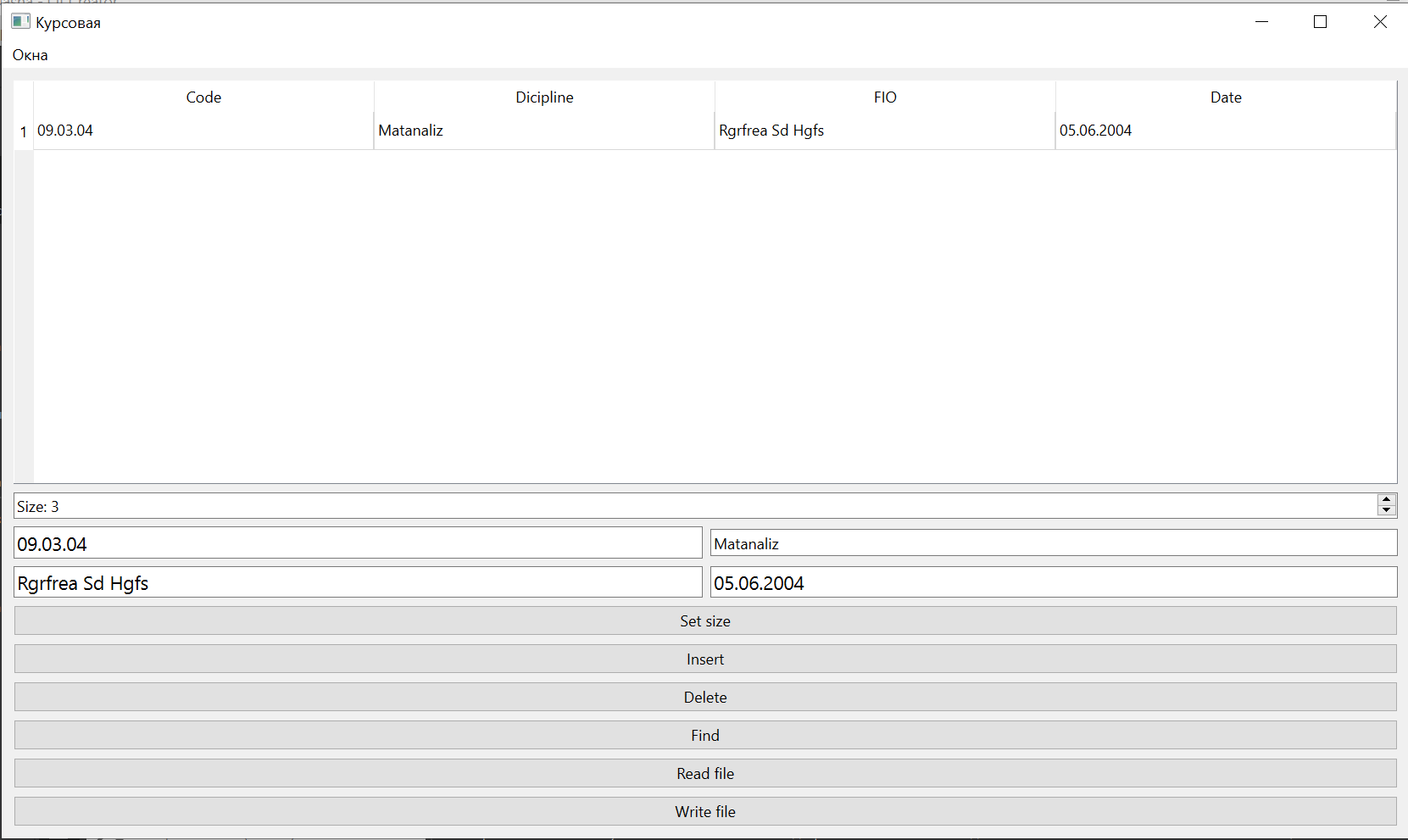


Рисунок 5 – Вставка

«Delete» позволяет удалить информацию по данным, введенным в поля. «Find» выполняет поиск по коду направления, названию дисциплины и дате.

«Read File» позволяет считать входные данные из файла. Нажав на данную кнопку, необходимо выбрать и открыть текстовый файл, после чего в справочник добавятся данные (рисунок 6)

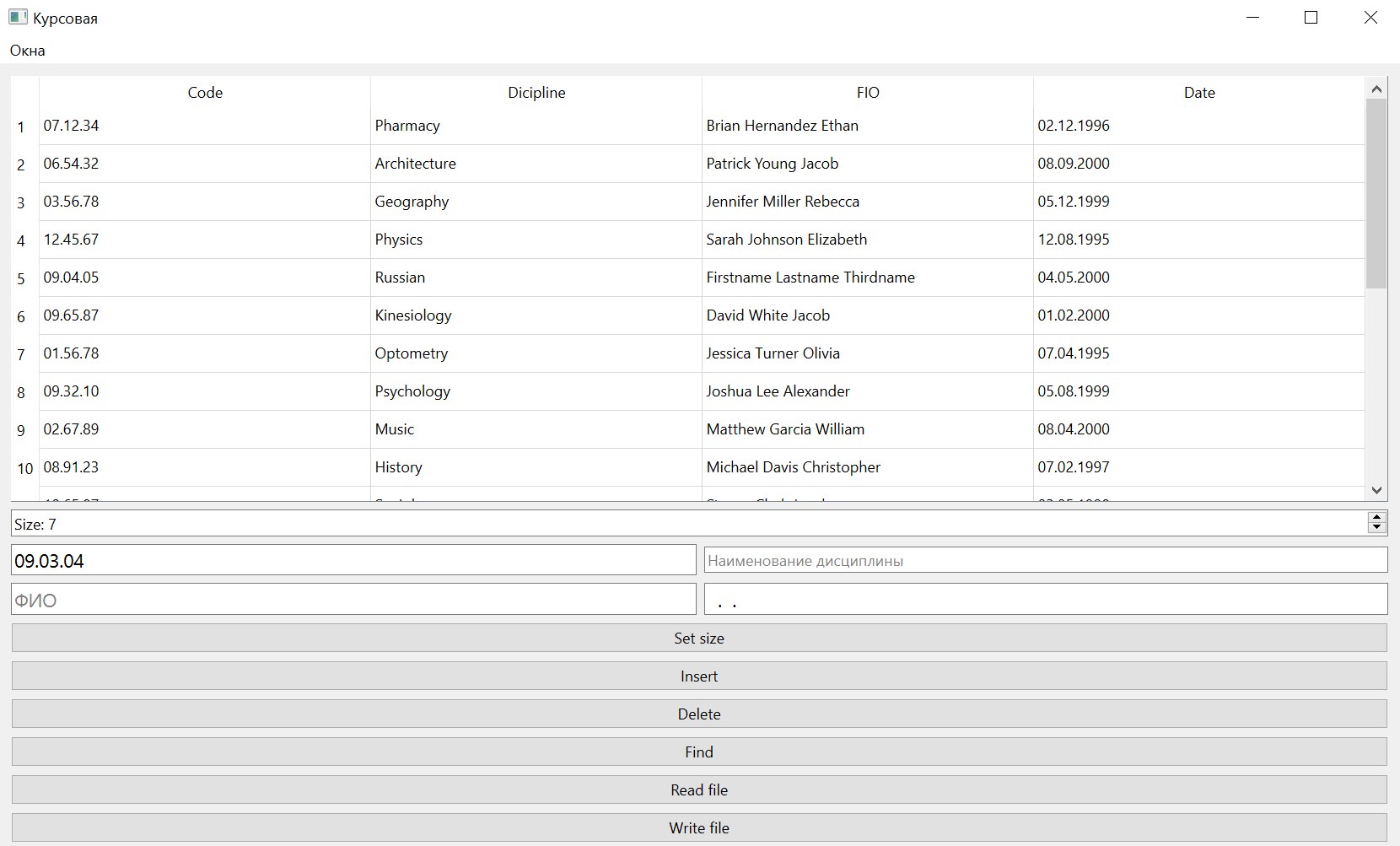


Рисунок 6 – Считывание данных из файла

На рисунке 7 в левом верхнем углу видна кнопка «окна», позволяющая переключаться между ними

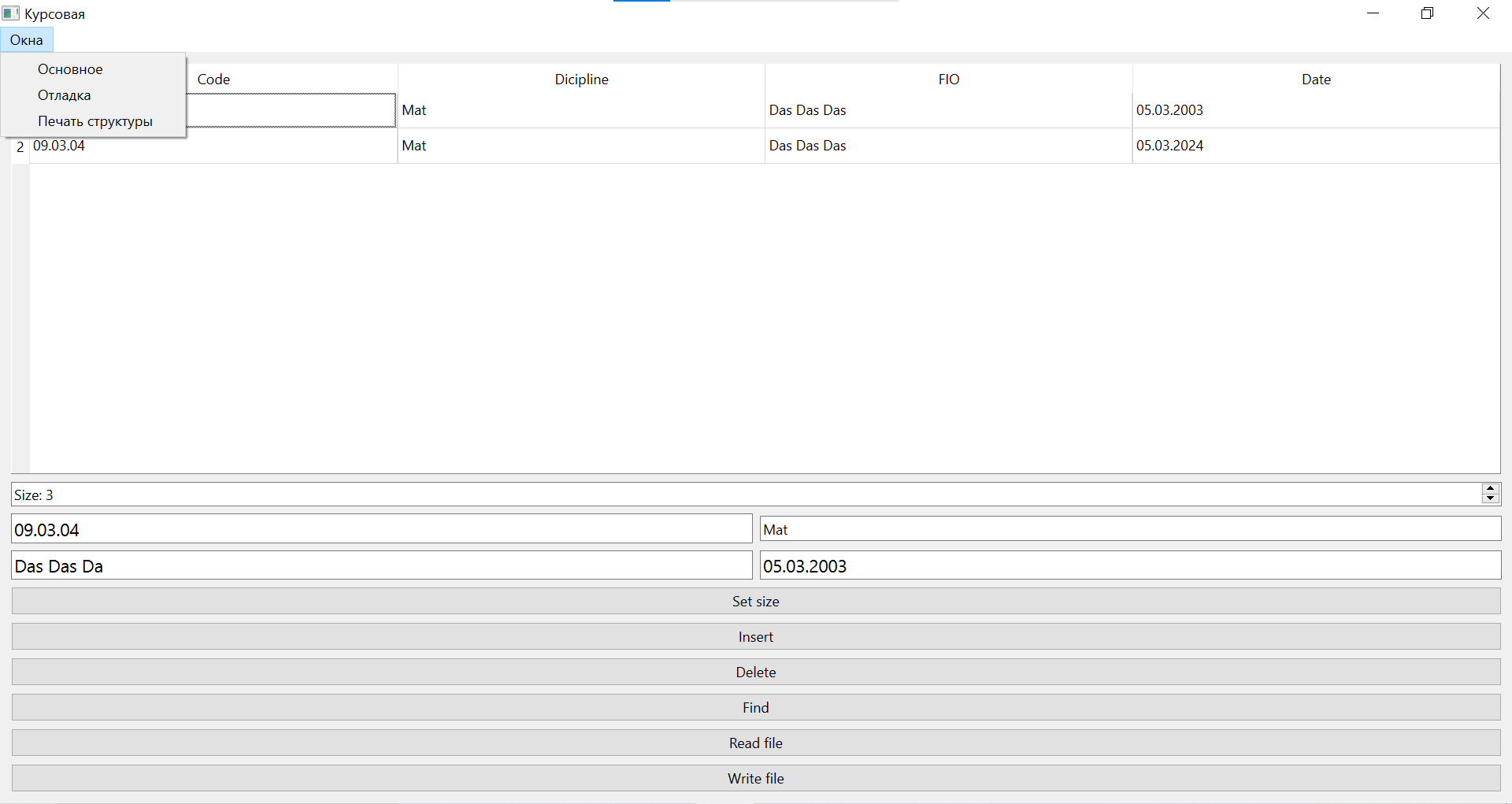


Рисунок 7 – Окна

Перейдя в окно «Отладка» можем увидеть результат выполнения той или иной операции или сообщения об ошибках (рисунок 8)

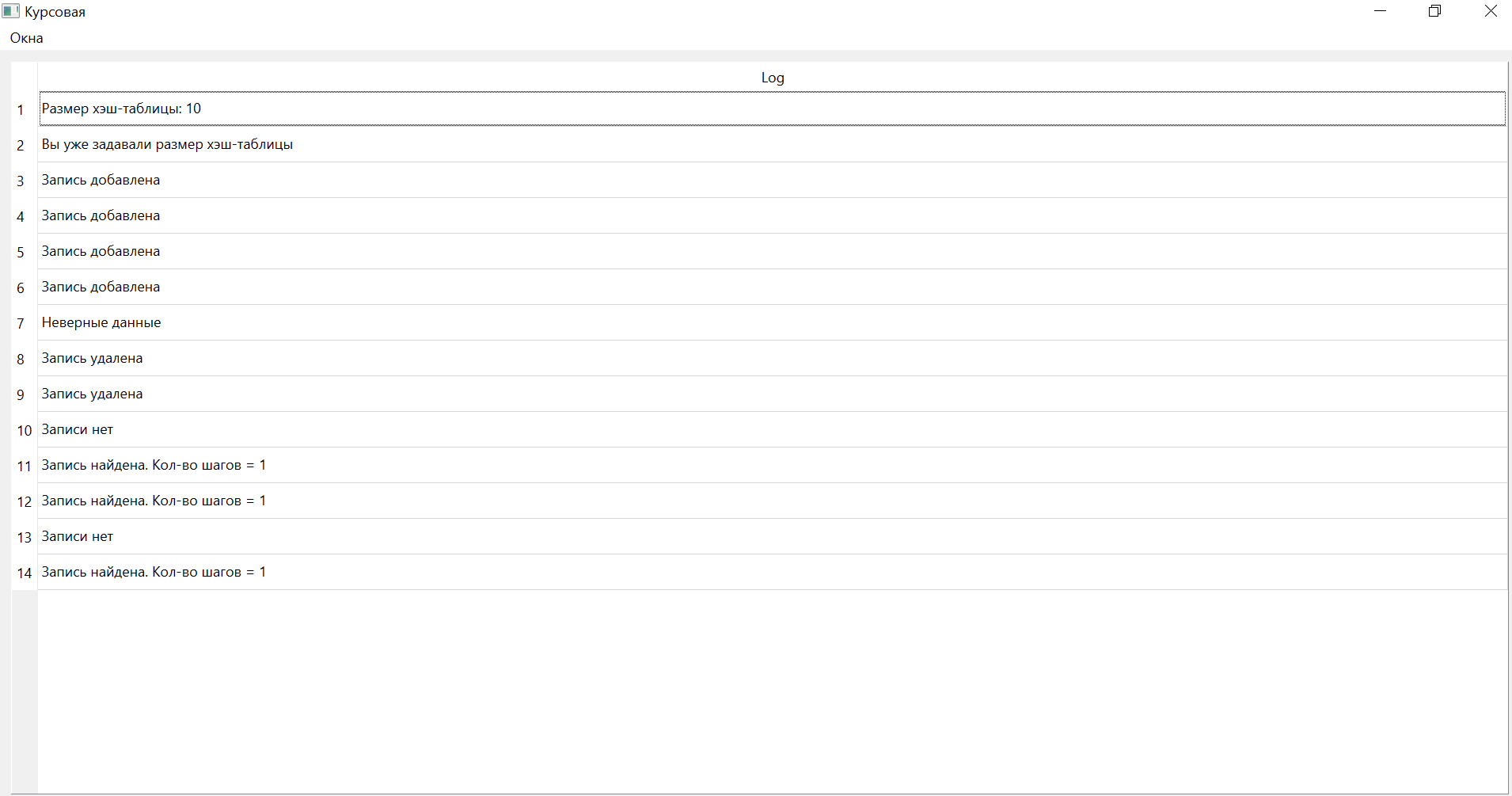


Рисунок 8 – Отладка

В окне «Печать структуры» выводится результат хеш-таблицы с двусвязными списками (рисунок 9).

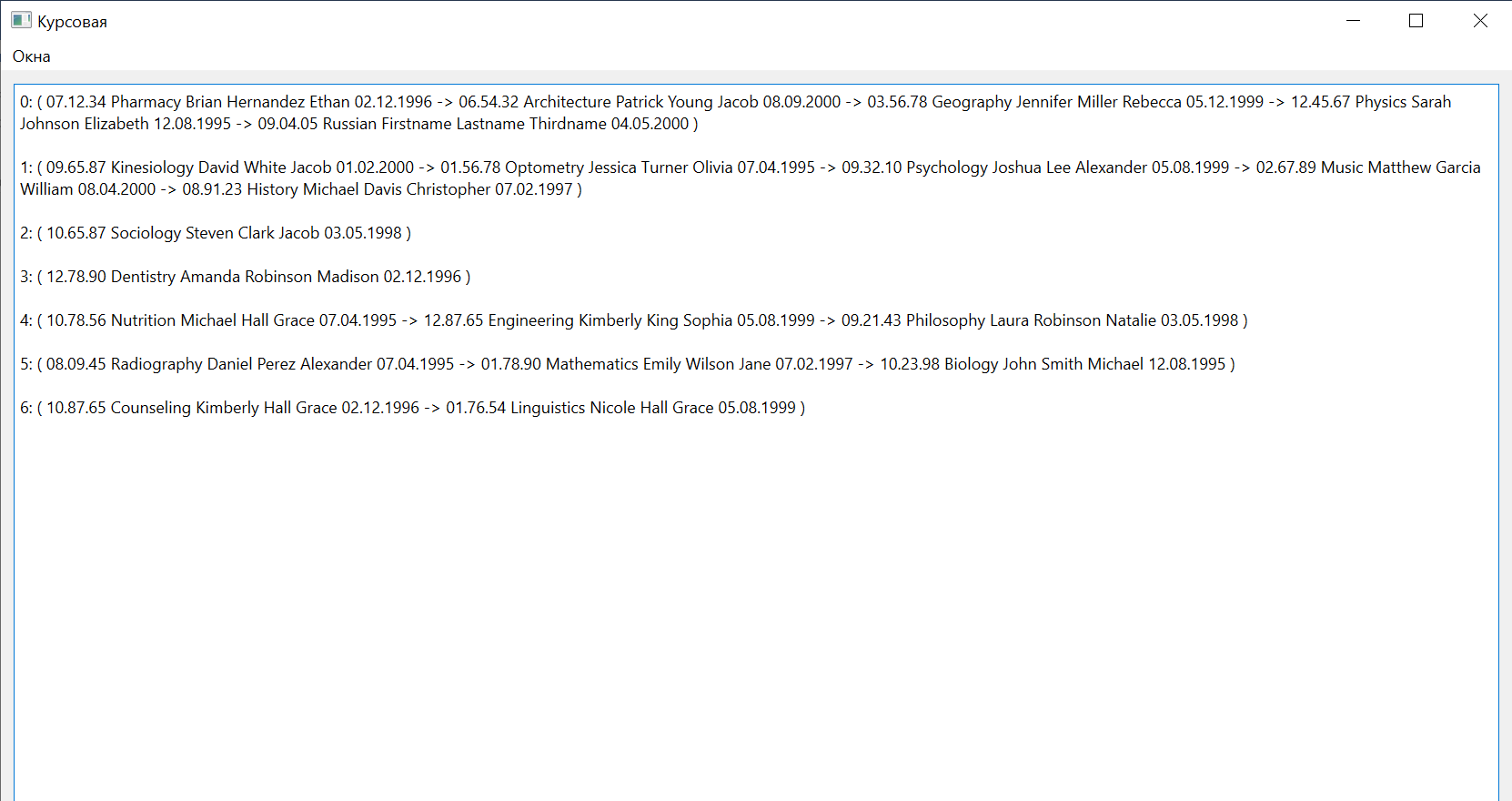


Рисунок 9 – Печать структуры

«Write File» записывает получившиеся данные в выбранный текстовый файл (рисунок 10)



Рисунок 10 – Write File

4.4 Тестирование

Тестирование проводилось методом белого ящика, результаты которого представлены в Таблицах 3, 4, 5.

Таблица 3 – Тестирование работы со справочником «Преподавательский состав»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание тестовой ситуации | | Входные данные | | Выходные данные | |
| Ваш справочник | Ваши поля | Ваш справочник | Сообщение |
| Добавление | | | | | |
| 1 | Добавление некорректных данных | - | 10.05.03 Biology Kuznetsova Elena Viktorovna 19.95.9698 | - | Неверные данные |
| 2 | Добавление пустых данных | - | - | - | Неверные данные |
| 3 | Добавление корректных данных | - | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Запись добавлена |
| 4 | Проверка уникальности | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Ошибка добавления |
| Удаление | | | | | |
| 5 | Удаление несуществующего ключа | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.02.64 Sports Dimov Dima Dimovich 03.02.2005 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Ошибка удаления |
| 6 | Удаление при неполном совпадении | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.05 Algorithm Klimanov Klim Klimovic 03.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Ошибка удаления |
| 7 | Удаление ключа | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | - | Запись удалена |
| Поиск | | | | | |
| 8 | Запись не существует | - | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | - | Записи нет |
| 9 | Запись существует | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Запись найдена |
| 10 | Поиск при несовпадении ФИО | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Dimov Dima Dimov 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Запись найдена |
| 11 | Поиск при несовпадении даты | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2005 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Записи нет |
| 12 | Поиск при несовпадении кода направления | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.04.08 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Записи нет |
| 13 | Поиск при несовпадении названия дисциплины | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Sport Klimanov Klim Klimovich 05.05.2004 | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | Записи нет |

Таблица 4 – Тестирование структуры данных хеш-таблица с методом разрешения коллизии цепочками

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание тестовой ситуации | | Входные данные | | Выходные данные | |
| Хеш-таблица | Ваши поля | Хеш-таблица | Ваши поля |
| Добавление | | | | | |
| 1 | Добавление пустых данных | - | - | - | Ничего не добавится |
| 2 | Добавление корректных данных | - | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 |  |  |
| 3 | Добавление корректных данных в свободную ячейку |  | 09.03.05 Algorithms Klimanov Klim Klimov 05.05.2000 |  |  |
| 4 | Добавление дубликата |  | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 |  | Такой ключ уже есть |
| Удаление | | | | | |
| 3 | Удаление несуществующего элемента |  | 09.03.04 Sport Dima Dimov Dimovich 05.05.2000 |  | Ошибка удаления |
| 4 | Удаление при неполном совпадении |  | 09.03.04 Algorithm Klimanov Klim Klimov 05.05.2005 |  | Ошибка удаления |
|  | Удаление элемента |  | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | - | Запись удалена |
| Поиск | | | | | |
| 5 | Поиск существующего элемента |  | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 |  | Запись найдена |
| 6 | Поиск несуществующей записи |  | 01.01.04 English Tim Timur Timurovich 09.09.2001 |  | - |
|  | Поиск при коллизии |  | 09.03.01 Sport Timur Tim Timurovich 05.07.2008 |  | Запись найдена |

Таблица 5 – Тестирование структуры данных двусвязный список

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание тестовой ситуации | | Входные данные | | Выходные данные | |
| Двусвязный список | Ваши поля | Двусвязный список | Ваши поля |
| Добавление | | | | | |
| 1 | Добавление пустых данных | - | - | - | Ничего не добавится |
| 2 | Добавление корректных данных в пустой список | - | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 |  |  |
| 3 | Добавление корректных данных в непустой список |  | 09.03.04 Sport Dima Dimov Dimovich 05.05.2000 |  |  |
| 4 | Добавление дубликата |  | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 |  | Такой ключ уже есть |
| Удаление | | | | | |
| 3 | Удаление несуществующего элемента |  | 09.03.39 English Artem Ivanovich Artemov 05.05.2020 |  | Ошибка удаления |
| 4 | Удаление при неполном совпадении |  | 09.03.05 Algorithm Klimanov Klim Klimovic 05.05.2005 |  | Ошибка удаления |
|  | Удаление элемента |  | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 | - | Запись удалена |
|  | Удаление последнего элемента |  | 02.07.09 Sports Artemiev Artem Artemov 19.01.2020 |  |  |
|  | Удаление первого элемента |  | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 |  |  |
| Поиск | | | | | |
| 5 | Поиск существующего элемента |  | 09.03.04 Algorithms Klimanov Klim Klimovich 05.05.2004 |  | Запись найдена |
| 6 | Поиск несуществующей записи |  | 01.01.04 English Tim Timur Timurovich 09.09.2001 |  |  |

# Заключение

Целью курсового проекта было: разработка информационной системы для работы со справочником предметной области «Преподавательский состав».

Цель достигнута. Для достижения поставленной цели были выполнены следующие задачи:

1. Проведён анализ предметной области «Преподавательский состав» и построена её модель;
2. Изучены теоретические основы методов построения справочников;
3. Определены требования к информационным системам;
4. Информационная система было реализована и спроектирована;
5. Была изучена среда разработки Qt Creator;
6. Повысился навык написания алгоритмов на языке C++

# Список литературы

1. <https://tfcp.ru/ru/Раздел№3-1/хеш-таблица>
2. Седжвик Р. Фундаментальные алгоритмы на C++. Анализ/Структуры данных/Сортировка/Поиск. / Роберт Седжвик. – К.: Издательство «ДиаСофт», 2001. - 688 с.
3. <https://javarush.com/quests/lectures/questharvardcs50.level05.lecture06>
4. Лекции ФСДиА

