

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ  
ФЕДЕРАЦИИ**  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
**«Нижегородский государственный университет им. Н.И. Лобачевского»  
Национальный исследовательский университет**

**Институт информационных технологий, математики и механики  
Кафедра математического обеспечения и суперкомпьютерных технологий**

**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**  
**«Структуры данных: реализация просмотрной таблицы»**

Выполнил:  
студент группы 361706-1  
Резанцев Сергей Алексеевич  
\_\_\_\_\_ Подпись

Научный руководитель:  
ассистент каф. МОСТ ИИТММ  
\_\_\_\_\_ Лебедев И. Г.

Нижний Новгород  
2019 г.

|                                 |           |
|---------------------------------|-----------|
| <b>Содержание</b>               |           |
| <b>Введение</b>                 | <b>3</b>  |
| <b>Постановка задачи</b>        | <b>4</b>  |
| <b>Руководство пользователя</b> | <b>5</b>  |
| <b>Руководство программиста</b> | <b>6</b>  |
| Описание структуры программы    | 6         |
| Описание структуры данных       | 6         |
| Описание алгоритмов             | 7         |
| <b>Эксперименты</b>             | <b>9</b>  |
| <b>Заключение</b>               | <b>10</b> |
| <b>Литература</b>               | <b>11</b> |

## 1. Введение

Представление данных во многих задачах из разных областей человеческой деятельности может быть организовано при помощи таблиц. Таблицы представляют собой последовательности строк (записей), структура строк может быть различной, но обязательным является поле, задающее имя (ключ) записи. Таблицы применяются в бухгалтерском учете (ведомости заработной платы), в торговле (прайс-листы), в образовательных учреждениях (экзаменационные ведомости) и являются одними из наиболее распространенных структур данных, используемых при создании системного и прикладного математического обеспечения. Таблицы широко применяются в трансляторах (таблицы идентификаторов) и операционных системах, могут рассматриваться как программная реализация ассоциативной памяти и т.п. Существование отношения «иметь имя» является обязательным в большинстве разрабатываемых программистами структур данных; доступ по имени в этих структурах служит для получения соответствия между адресным принципом указания элементов памяти ЭВМ и общепринятым (более удобным для человека) способом указания объектов по их именам.

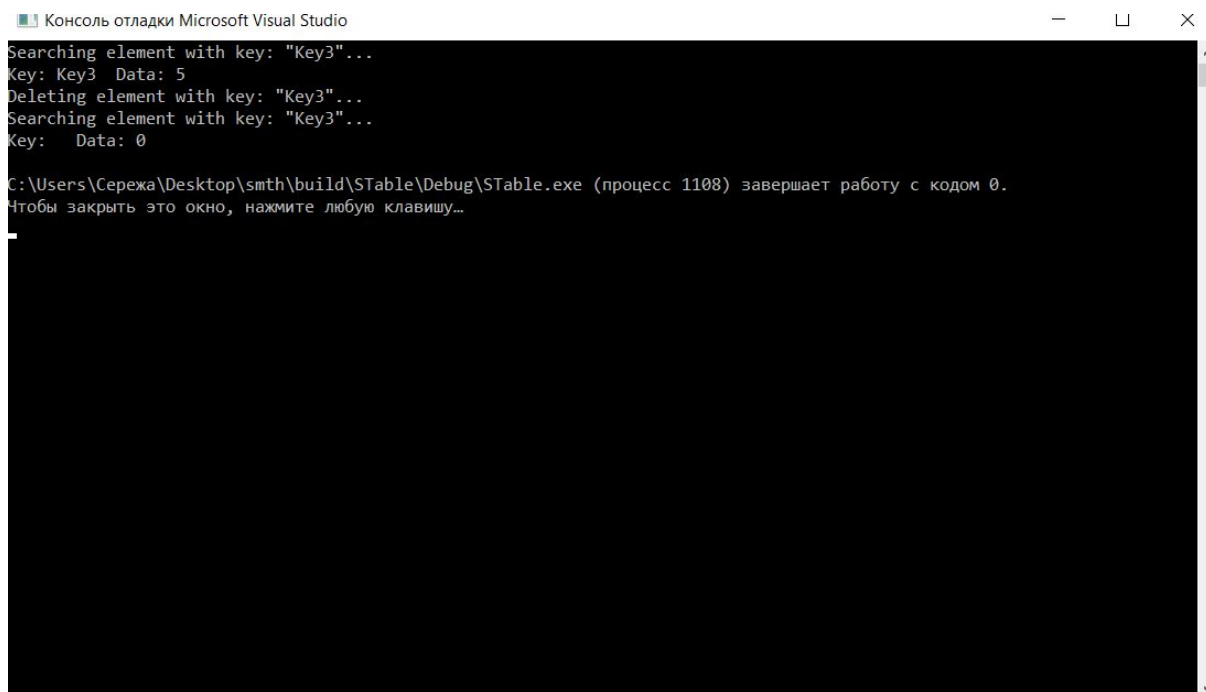
## **2. Постановка задачи**

В рамках данной лабораторной работы ставится задача создания программных средств, поддерживающих табличные динамические структуры данных (таблицы) и базовые операции над ними: поиск записи; вставка записи (без дублирования); удаление записи.

### 3. Руководство пользователя

Чтобы начать работу с программой запустите приложение STable.

На экране появится следующее:

A screenshot of the Microsoft Visual Studio debug console window. The title bar reads "Консоль отладки Microsoft Visual Studio". The console output is as follows:

```
Searching element with key: "Key3"...  
Key: Key3 Data: 5  
Deleting element with key: "Key3"...  
Searching element with key: "Key3"...  
Key: Data: 0  
  
C:\Users\Сергеа\Desktop\smith\build\STable\Debug\STable.exe (процесс 1108) завершает работу с кодом 0.  
Чтобы закрыть это окно, нажмите любую клавишу...
```

*Рисунок 1. Результат программы, выведенный на консоль.*

Затем программа завершится.

## 4. Руководство программиста

### 4.1. Описание структуры программы

Программа состоит из следующих модулей:

- TSTable.h - заголовочный файл класса «просмотровая таблица»
- TElem.h – заголовочный файл класса «элемент»
- TElem.cpp
- TSTable.cpp
- main.cpp – файл, содержащий тело программы, которая выполняет задачи поставленные ранее в пункте «Постановка задачи»

### 4.2. Описание структуры данных

Класс TElem является шаблонным.

Защищенные поля:

- string key; //ключ
- T data; //данные

Содержит публичные методы:

- TElem(); //конструктор по умолчанию
- TElem(string \_key, T \_data); //конструктор инициализатор
- TElem(TElem<T> &A); //конструктор копирования
- void SetKey(string \_key); //установить ключ
- void SetData(T \_data); //установить данные
- string GetKey(); //получить ключ
- T& GetData(); //получить данные
- TElem<T>& operator=(TElem<T> &A); //присваивание
- bool operator==(TElem<T> &A); //сравнение на равенство
- bool operator!=(TElem<T> &A); //сравнение на не равенство

Класс TSTable является шаблонным.

Защищенные поля:

- TElem<T> not\_find;
- TElem<T>\* mas; //массив записей
- int size; //размер таблицы

- `int count;` //количество занятых записей в таблице

Содержит публичные методы:

- `TSTable(int _size = 10);` //конструктор инициализатор
- `TSTable(TSTable<T> &A);` //конструктор копирования
- `void SetSize(int _size);` //установить размер таблицы
- `int GetSize();` //получить размер таблицы
- `int GetCount();` //получить количество занятых записей
- `void Put(string _key, T _data);` //вставка записи
- `void Del(string _key);` //удаление записи
- `TElem<T>& Find(string _key);` //поиск записи по ключу
- `T& operator[](string _key);` //индексация

### 4.3. Описание алгоритмов

В данном разделе не будут рассматриваться тривиальные методы.

Добавление записи в таблицу.

На вход подается переменная строкового типа, которая обозначает ключ таблицы, и сами данные.

Если вставка записи происходит в уже заполненную таблицу, бросается исключение. Иначе, добавляем запись в конец таблицы и увеличиваем значение занятых ячеек на единицу.

Поиск записи в таблице.

На вход подается переменная строкового типа-ключ, по которому мы будем искать нужную запись. В цикле, начиная с первой записи, сравниваем поочередно все ключи, до совпадения с искомым.

Если нашли совпадающие элементы, возвращаем эту запись. Иначе, возвращаем специальный элемент, который означает пустую запись.

Удаление записи из таблицы.

Если удаление записи происходит в пустой таблице, бросается исключение.

Сначала применяем метод поиска записи, которую мы хотим удалить по этому ключу. Если этот метод вернул запись, зануляем ее специальным элементом, который означает пустую запись.



## 5. Эксперименты

Параметры ПК:

- Операционная система - Windows 10
- Характеристики компьютера:
- Intel Core i5
- 8 GB DDR3 L Memory
- 128 GB SSD + 1000 GB HDD

| Кол-во элементов в таблице | Добавление элемента,с | Удаление элемента, с | Поиск элемента, с |
|----------------------------|-----------------------|----------------------|-------------------|
| 1000                       | 0                     | 0                    | 0                 |
| 100000                     | 0                     | 0,001                | 0,001             |
| 10000000                   | 0                     | 0,066                | 0,067             |

Таким образом мы видим, что сложность метода Put -  $O(1)$ , Del, Search -  $O(n)$

## **6. Заключение**

В ходе выполнения лабораторной работы была реализована динамическая структура данных - таблица и ее методы: вставка, поиск и удаление записи.

Предоставлено описание примера работы программы в разделе «Руководство пользователя».

Также разработаны и доведены до успешного выполнения тесты, проверяющие корректность методов классов TSTable, TElem.

## 7. Литература

1. Васильев А.Н. Самоучитель С++ с примерами и задачами. -СПб.: Наука и Техника, 2016. -480с.
2. Т. А. Павловская С/С++ Программирование на языке высокого уровня. - СПб.:Питер, 2011. - 461 с.
- 3.Крапенко С. Н. и др. Методы объектно-ориентированного программирования. <http://e-learning.unn.ru/course/view.php?id=251>.
- 4.Страуструп. Б. Курс «Язык программирование С++ для профессионалов» <http://www.intuit.ru/studies/courses/98/98/info>
- 5.Гергель В.П. Методические материалы по курсу “Методы программирования 2”:  
[<http://www.itmm.unn.ru/files/2018/10/Primer-1.1.-Struktury-hraneniya-mnozhestva.pdf>], 2015.
- 6.Барышева И.В., Мееров И.Б., Сысоев А.В., Шестакова Н.В. ЛАБОРАТОРНЫЙ ПРАКТИКУМ 2017. - 105 с