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**

**«Реализация таблиц»**

**Выполнил:** студент группы 381706-1

Остапович Д.Е.

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

аспирант каф. МОСТ ИИТММ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лебедев И.Г.

Нижний Новгород

2019.

Содержание

[1. Введение 3](#_Toc533027637)

[2. Постановка задачи 4](#_Toc533027638)

[3. Руководство пользователя 5](#_Toc533027639)

[4. Руководство программиста 6](#_Toc533027640)

4.[1. Описание структуры программы 6](#_Toc533027641)

4.[2. Описание структур данных 6](#_Toc533027642)

4.[3. Описание алгоритмов 6](#_Toc533027643)

[5. Заключение 9](#_Toc533027644)

[6. Литература 10](#_Toc533027645)

# **Введение**

**Структура данных** - программная единица, которая определяет метод хранения и обработки различных логически связанных данных в вычислительной технике. Знание структур данных позволяет наиболее компактно и практично расположить данные в памяти компьютера. В данной работе мы рассмотрим такую структуру данных, как **таблица**.

**Таблица** –  это совокупность связанных данных, хранящихся в структурированном виде. Таблица использует модель вертикальных столбцов (имеющих уникальное имя) и горизонтальных строк. Ячейка — место, где строка и столбец пересекаются. Таблица содержит определенное число столбцов, но может иметь любое количество строк. Каждая строка однозначно определяется одним или несколькими уникальными значениями, которые принимают её ячейки из определенного подмножества столбцов. Подмножество столбцов, которое уникально идентифицирует строку, называется [первичным ключом](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B2%D0%B8%D1%87%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BB%D1%8E%D1%87). Программная реализация таблиц возможна на основе различных структур данных, например, с использованием статических или динамических одномерных массивов, линейных списков или деревьев.

**Целью данной лабораторной работы** является разработка структуры данных для хранения таблиц с использованием динамических массивов, а также освоение таких инструментов разработки программного обеспечения, как система контроля версий Git и фреймворк для разработки автоматических тестов Google Test.

# **Постановка задачи**

Выполнение работы предполагает решение следующих задач:

1. Реализация класса таблицы TViewTable.
2. Реализация класса элемента таблицы TElem.
3. Создание примера использования.
4. Реализация тестов на базе фреймворка Google Test.

# **Руководство пользователя**

Пример использования таблицы:

* В начале программы создается таблица с целочисленными данными. В нее заносятся три элемента с ключами D, E и N. Полученная таблица выводится на экран.
* Далее осуществляется попытка добавить еще один элемент с ключом, который уже зарезервирован в таблице, выводится сообщение об ошибке.
* Затем программа предлагает пользователю добавить собственную строку с данными в таблицу.
* Результат всех действий с таблицей выводятся на экран.
* Выполняется метод Search с ключом D.
* Затем оператор индексации по ключу В возвращает данные из соответствующей строки таблицы.

# **Руководство программиста**

## **Описание структуры программы**

**Проект “ViewTable ” состоит из следующих файлов:**

1. main.cpp (в нём находится main)

**Проект “ ViewTableLib” состоит из следующих файлов:**

1. viewtable.h (описание класса “TViewTable” реализация методов класса “ TViewTable ” и перегрузка операций )
2. viewtable.cpp
3. telem.h (описание класса “TElem” реализация методов класса “TElem” и перегрузка операций )

**Проект “ ViewTableTest” состоит из следующих файлов:**

1. ViewTableTest.cpp (реализация тестов для класса TViewTable)
2. test\_main.cpp

## **Описание структур данных**

***Класс TElem:***

*Поля:*

* string key; - размер стека;
* T data; - элемент, расположенный на вершине стека;

*Конструкторы и деструктор:*

* TElem (string \_key = “”, T \_data=0); - конструктор инициализации;
* TElem(TElem<T> &A); - конструктор копирования;

*Методы:*

* void SetKey(string \_key); - установить ключ;
* void SetData(T \_data); - установить данные;
* string GetKey(); - получить ключ;
* T GetData(); - получить данные;

*Перегруженные операторы:*

* TElem& operator = (TElem &A); - оператор присваивания;
* bool operator == (TElem &A); - оператор равно;
* bool operator != (TElem &A);- оператор неравно;

***Класс TViewTable:***

*Поля:*

* TElem<T> not\_found; - несуществующий элемент;
* TElem<T>\* mas; - массив элементов;
* int size, count; - размер количество элементов;

*Конструкторы и деструктор:*

* TViewTable (int \_size = 0); - конструктор инициализации;
* TViewTable (TViewTable& A); - конструктор копирования;

*Методы:*

* void SetSize (int \_size); - установить размер;
* void SetCount (int \_count); - установить количество заполненных элементов;
* TElem<T>\* GetMas(); - получить массив элементов;
* int GetSize(); - получить размер;
* int GetCount(); - получить количество элемегтов;
* void Put (string \_key, T \_data); - добавить строку к таблице;
* void Put (TElem<T> A); - добавить элемент к таблице;
* void Del (string \_key); - удалить элемент по ключу;
* TElem<T>& Search (string \_key); - найти элемент по ключу;
* bool IsFull (); - проверка таблицы на зполненность;
* bool IsEmpty(); - проверка таблицы на пустоту;

*Перегруженные операторы:*

* T operator[](string \_key); - оператор индексации;

## **Описание алгоритмов**

Метод **Put:**

Если таблица заполнена, бросаем исключение. Если такой ключ уже существует выводим предупреждение об этом. Иначе заполняем последнюю строчку таблицы и увеличиваем количество элементов на 1.

Метод **Del:**

Если таблица пустая, бросаем исключение. Иначе ищем значение элемента по ключу. Если такой элемент существует, то присваиваем ему значение несуществующего элемента.

Метод **Search**:

Проходимся по массиву элементов. Если такой ключ существует, то возвращаем его. Иначе возвращаем not\_found.

# **Заключение**

В ходе выполнения лабораторной работы удалось реализовать такую структуру данных, как стэк. Вместе с ней, согласно заданному интерфейсу, был реализован класс TViewTable и TElem.

В результате проделанной работы удалось

1. Реализовать класс таблицы TViewTable.

2. Реализовать класс элемента таблицы TElem.

3. Обеспечить работоспособность примера использования.

4. Реализовать тесты с использованием фреймворка Google Test.

# **Литература**

1. Лабораторный практикум. Составители: Барышева И.В., Мееров И.Б., Сысоев А.В., Шестакова Н.В. Под редакцией Гергеля В.П. Учебно-методическое пособие. – Нижний Новгород: Нижегородский госуниверситет, 2017. – 105с.

URL: <http://www.unn.ru/books/met_files/Pract_ADS.pdf>

1. Национальный открытый университет «Интуит». Курс «Основы программирования». Лекция 11: Структуры данных: общее понятие, реализация. Простейшие структуры данных: очередь, стек. Использование стека и обратная польская запись.

URL: <https://www.intuit.ru/studies/courses/2193/67/lecture/1980?page=3>