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**

**«Структура хранения данных: упорядоченные таблицы»**

**Выполнил:** студент группы 381706-1

Кольтюшкина Янина Вадимовна

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лебедев И.Г

Нижний Новгород

2019.

**Оглавление**

[1. Введение. 2](#_Toc8237860)

[2. Постановка задачи. 3](#_Toc8237861)

[3. Руководство пользователя. 4](#_Toc8237862)

[4. Руководство программиста. 5](#_Toc8237863)

[4.1. Описание структуры программы. 5](#_Toc8237864)

[4.2. Описание структур данных. 5](#_Toc8237865)

[4.3. Описание алгоритмов. 6](#_Toc8237866)

[5. Эксперименты. 7](#_Toc8237867)

[6. Заключение. 8](#_Toc8237868)

[7. Литература. 9](#_Toc8237869)

# Введение.

**Целью** данной лабораторной работы является практическая реализация структуры данных – просмотровая таблица

Ознакомимся со следующими базовыми понятиями.

**Таблица** - двухмерный объект, состоящий из строк и столбцов, который используется для хранения данных.

**Упорядоченная таблица** –этоодномерный массив элементов с заранее известным размером. Т.к. данные таблицы упорядочены, то поиск элементов, добавление элемента и удаление элемента из таблицы имеют сложность .

# Постановка задачи.

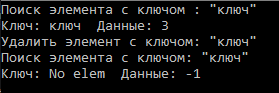
Одной из главных задач данной лабораторной работы является разработка и реализация структуры хранения «просмотровая таблица»

Программное решение будет выглядеть следующим образом:

1. Описание и реализация класса элемента таблицы **TSortElem**.
2. Описание и реализация класса просмотровой таблицы **TSortTable**.
3. Тестирование класса **TSortTable** с помощью Google Test.
4. Реализация класса **ExceptionLib** для обработки исключений, которые могут возникнуть в результате работы класса **TSortTable**.
5. Демонстрация работы класса **TSortTable**.

# Руководство пользователя.

Пример работы класса **TSortTable**:



*Рис 1. Пример работы программы*

При запуске программы создается таблица из 6 элементов. Далее она заполняется по следующему принципу: ключом элемента служит русскоязычная строка, где к каждому последующему ключу прибавляется одна буква, чтобы получить из «к» слово «ключик», данными элементов служат элементы ряда Фибоначчи. Затем производится поиск элемента с ключом «ключ» и вывод его данных на экран. После этого элемент с этим же ключом удаляется и вновь производится его поиск, результат которого означает, что элемент не найден.

# Руководство программиста.

# Описание структуры программы.

* Модуль **SortTable** – содержит пример использования класса **TSortTable**. Реализация в файле **sort\_table\_main.**.
* Модуль **SortTableLib –** содержит описание и реализацию классов **TSortTable** и **SortTableElem** в файлах T**SortTable.h** и **TSortElem.h**.
* Модуль **SortTableTest** – содержит в себе файл **test\_sort\_table.cpp**, в котором находится набор тестов, для проверки работоспособности классов **TSortElem** и **TSortTable**.
* Модуль **ExceptionLib** –содержитв себе файл **ExceptionLib.h** с реализацией класса исключений **MyException**.

# Описание структур данных.

**Класс TSortElem.**

Класс **TSortElem** является шаблонным классом и содержит два поля со спецификатором **protected**:

* **string key** – строка-ключ для доступа к элементам таблицы.
* **T data** – данные, содержащиеся в элементе таблицы.

Далее идут методы класса со спецификатором доступа **public**:

* **TSortElem(string \_key, T \_data)** – конструктор-инициализатор.
* **TSortElem(TSortElem &elem)** – конструктор копирования.
* **void SetKey(string \_key)** – метод, устанавливающий ключ.
* **void SetData(T \_data)** – метод, устанавливающий данные.
* **T GetData()** – метод, возвращающий данные.
* **T& GetDataAddress()** – метод, возвращающий адрес данных.
* **string GetKey()** – метод, возвращающий ключ.
* **TSortElem<T>& operator=(TSOrtElem &elem)** – оператор присваивания.
* **bool operator==(TElem &elem)** – оператор сравнения.
* **bool operator>(TSortElem<T> &elem)** – оператор «больше».
* **bool operator<(TSortElem<T> &elem)** – оператор «меньше».

**Класс TSortTable.**

Класс **TSortTable** является шаблонным классом и содержит четыре поля со спецификатором **protected**:

* **TSortElem<T> notFound** – специальная константа, которая означает, что элемент не найден.
* **TSortElem<T> \*mas** – массив элементов таблицы.
* **int length** – размер таблицы.
* **int count** – количество элементов в таблице.

Далее идут методы класса со спецификатором доступа **public**:

* **TSortTable(string\* \_keys, T\* \_data, int \_leng)** – конструктор-инициализатор.
* **TSortTable(int \_leng)** – конструктор по умолчанию.
* **TSortTable(TSortTable &elem)** – конструктор копирования.
* **void SetLeng(int \_leng)** – метод, устанавливающий размер таблицы.
* **void SetCount(int \_count)** – метод, устанавливающий количество элементов в таблице.
* **int GetLeng()** – метод, возвращающий размер таблицы.
* **int GetCount()** – метод, возвращающий количество элементов в таблице.
* **void Put(string \_key, T \_data)** – метод, помещающий элемент в таблицу.
* **void Del(string \_key)** – метод, удаляющий элемент из таблицы.
* **TSortElem<T>& Search(string \_key)** – метод поиска элемента в таблице.
* **T& operator[](string \_key)** – оператор доступа к элементам таблицы по ключу.

**Класс ExceptionLib.**

Класс **Exception** содержит одно поле со спецификатором доступа **private**:

* **string str** – строка, хранящая сообщение об ошибке.

Далее идут методы класса со спецификатором доступа **public**:

* **Exception(string \_str)** – конструктор инициализатор.
* **void what()** – метод вывода ошибки на экран.

# Описание алгоритмов.

**Добавление элемента в таблицу.**

Сначала проверяем таблицу на полноту. Если в ней нет свободного места, то с помощью метода **Resize(int \_size)** увеличиваем размер таблицы вдвое. Затем, с помощью бинарного поиска ищем место, куда можно поместить новый элемент так, чтобы таблица оставалась упорядоченной. После того, как мы нашли нужное место, сдвигаем элементы таблицы, стоящие после найденного места, на одно место вправо. И наконец помещаем новый элемент в найденное место.

**Удаление элемента из таблицы.**

Сначала с помощью бинарного поиска ищем нужный элемент. После того как мы нашли элемент, мы сдвигаем элементы таблицы, стоящие после найденного элемента, на одно место влево.

**Поиск элемента в таблице.**

Поиск выполняется по ключу. Совершается обход таблицы с помощью бинарного поиска. Если элемент с искомым ключом найден, то он возвращается. Если нет, то возвращается константа **notFound**.

# Эксперименты.

Эксперименты проводились на ПК с следующими параметрами:

1. Операционная система: Windows 10
2. Процессор: Intel(R) Core™ i7-4710MQ CPU @ 2.50 GHz
3. Версия Visual Studio: 2017

|  |  |  |  |
| --- | --- | --- | --- |
| Количество элементов в таблице | Время работы метода **Put(string \_key, T data)** (в млс) | Время работы метода **Del(string \_key)** (в млс) | Время работы метода **Search(string \_key)** (в млс) |
| 1000 | 0 | 0 | 0 |
| 10000 | 2 | 2 | 2 |
| 20000 | 4 | 32 | 3 |

*Таблица 1. Результаты экспериментов.*

Таким образом, можно увидеть, что сложность работы методов **Put(string \_key, T data)**, **Search(string \_key)** и **Del(string \_key)** – .

# Заключение.

В этой лабораторной работе я смогла справиться со всеми поставленными задачами: реализовала вспомогательный класс элемента таблицы **SortTableElem**, сам класс упорядоченной таблицы **TSortTable**, были реализованы тесты и обработкаисключений. В конечном итоге мне удалось реализовать структуру данных «упорядоченная таблицы».

# Литература.

1. Ссылка из Википедии про таблицы:

<https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0_(%D0%B7%D0%BD%D0%B0%D1%87%D0%B5%D0%BD%D0%B8%D1%8F)>

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», 2015.