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**

**«Просмотровая таблица»**

**Выполнил:** студент группы 381706-1

Корнев Никита Алексеевич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

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

Нижний Новгород

2018.

Содержание

[1. Введение 3](#_Toc533799739)

[2. Постановка задачи 4](#_Toc533799740)

[3. Руководство пользователя 5](#_Toc533799741)

[4. Руководство программиста 6](#_Toc533799742)

[4.1. Описание структуры программы 6](#_Toc533799743)

[4.2. Описание структур данных 7](#_Toc533799744)

[4.3 Описание алгоритмов 8](#_Toc533799745)

[5. Заключение 9](#_Toc533799746)

[6. Литература 10](#_Toc533799747)

# Введение

Структура данных (англ. data structure) — программная единица, позволяющая хранить и обрабатывать множество однотипных и/или логически связанных данных в вычислительной технике. Для добавления, поиска, изменения и удаления данных структура данных предоставляет некоторый набор функций, составляющих её интерфейс. Одной из часто используемых структур является просмотровая таблица.

С физической точки зрения таблица представляет собой вектор, элементами которого являются записи. Характерной логической особенностью таблиц, которая определяет их отдельное рассмотрение, является то, что доступ к элементам таблицы производится не по номеру (индексу), а по ключу - по значению одного из свойств объекта, описываемого элементом таблицы. Ключ - это свойство, идентифицирующее данную запись во множестве однотипных записей. Как правило, к ключу предъявляется требование уникальности в данной таблице. Ключ может включаться в состав записи и быть одним из ее полей, но может и не включаться в запись, а вычисляться по положению записи. Таблица может иметь один или несколько ключей.

В данной лабораторной работе нам предстоит написать собственную реализацию данной структуры.

# Постановка задачи

В данной лаборатороной работе нам предстоит решить следующие задачи:

1. Реализовать класс TTable.
2. Реализовать класс TArrayTable.
3. Реализовать класса элемента таблицы TRecord.
4. Реализовать класса таблицы TScanTable.
5. Реализовать тесты для Google Test.

# Руководство пользователя

Чтобы использовать данный класс в своем проекте, необходимо подключить библиотеку «TScanTable.h» и «TRecord.h». Библиотеки позволяют:

* Создавать объекты типа *элемент таблицы*:

TRecord<TValue> e1; **конструктор по умолчанию**

TRecord<TValue> e2("A", 1); **конструктор с параметром**

TRecord<TValue> e3(e2); **конструктор копирования**

* Создавать объекты типа *таблица*:

TScanTable<TValue> t1; **конструктор по умолчанию**

TScanTable<TValue>(100) t2; **конструктор c параметром**

* Устанавливать и получать ключ и данные элементов таблицы:

e1.SetKey(“B”);

e1.SetData(“2”);

e2.GetKey();

e2.GetData();

* Брать элементы из начала и конца *списка*:

A = li1.GetStart();

B = li1.GetEnd();

* Использовать некоторые операторы:

e2 = e1;

e1 == e3;

e3 != e2;

* Помещать, удалять, искать объекты типа *элемент таблицы* вобъектах типа *таблица:*

t1.Insert(e2);

t1.Find(1);

t1.Delete(1);

# Руководство программиста

## Описание структуры программы

Проект состоит из следующих модулей:

1. TScanTable

Модуль cодержит файл код «main.cpp», в котором продемонстрирован пример использования структуры.

1. TScanTableLib

Модуль содержит файл заголовок «TScanTable.h», описывающий структуру «просмотровая таблица» а также «TRecord.h», описывающий элемент таблицы.

1. TScanTableTest

Модуль содержит файлы кода «test\_main.cpp», «test\_TScanTable.cpp». В последнем реализованы тесты для проверки корректности работы методов данного класса.

## Описание структур данных

#### Структура TRecord

Поля со спецификатором доступа «protected»:

* ***string key;* //** ключ элемента
* ***TValue data;*** // данные, хранимые в элементе

Поля со спецификатором доступа «public»:

* ***void SetKey(string \_key);*** // установить ключ элемента
* ***void SetData(TValue \_data);*** // установить данные элемента
* ***string GetKey();*** // получить ключ элемента
* ***TValue GetData();*** // получить данные элемента
* ***TRecord<TValue>(string \_key = "", TValue \_data = 0);*** // конструктор по умолчанию/c параметром
* ***TRecord(TRecord<TValue> &A);*** // конструктор копирования
* ***TRecord<TValue>& operator = (TRecord<TValue> &A);*** // оператор =
* ***bool operator == (TRecord<TValue> &A);*** // оператор ==
* ***bool operator != (TRecord<TValue> &A);*** // оператор !=

#### Структура Ttable

Поля со спецификатором доступа «protected»:

* ***int DataCount;*** // количество записей в таблице
* ***int eff;*** // показатель эффективности выполнения операций

Поля со спецификатором доступа «public»:

* ***TTable();*** // конструктор
* ***~ TTable();*** // деструктор
* ***virtual bool IsFull() = 0;*** // проверка на полноту
* ***virtual bool Find(string key) = 0;*** // поиск
* ***virtual bool Insert(TRecord<TValue> record) = 0;*** // поместить в таблицу
* ***virtual void Delete(string key) = 0;*** // удалить
* ***virtual void GoNext() = 0;*** // перейти к следующему эелменту
* ***virtual bool IsEnd() = 0;*** // проверка на завершенность
* ***virtual void Reset() = 0;*** // устанвоить на первый элемент
* ***virtual TRecord<TValue> GetCurrent() = 0;*** // получить текущий
* ***virtual void SetCurrentValue(TValue value) = 0;*** // установить текущий

#### Структура TArrayTable

Поля со спецификатором доступа «protected»:

* ***TRecord<TValue> \*arr;*** // массив для записи
* ***int size;*** // размер
* ***int currNum;*** // номер текущего

Поля со спецификатором доступа «public»:

* ***TArrayTable(int \_size = 10);*** // конструктор
* ***~TArrayTable();*** // деструктор
* ***TArrayTable(const TRecord<TValue>& recArr);*** // конструктор копирования
* ***TRecord<TValue>\*& operator=(const TRecord<TValue>& recArr);*** // оператор =
* ***TRecord<TValue> GetCurrent();*** // получить текущий
* ***void Reset();*** // установить начало
* ***void GoNext();*** // перейти к следующему
* ***bool IsEnd();*** // проверка на последний
* ***bool IsFull();*** // проверка на полноту
* ***bool IsEmpty();*** // проверка на пустоту
* ***void SetCurrentValue(TValue val);*** // установить значение текущего
* ***int GetCurrNum();*** // получить номер текущего

#### Структура TscanTable

Поля со спецификатором доступа «public»:

* ***TScanTable(int \_size = 10);*** // конструтор по-умолчанию/с параметром
* ***bool Find(string k);*** // поиск
* ***bool Insert(TRecord<TValue> tr);*** // поместить элемент в таблицу
* ***void Delete(string k);*** // удалить

## Описание алгоритмов

1. bool Find(string k)
2. {
3. for (int i = 0; i < TTable<TValue>::DataCount; i++) // проходим по элементам таблицы
4. {
5. TTable<TValue>::eff++; // увеличиваем показатель эффективности
6. if (TArrayTable<TValue>::arr[i].GetKey() == k) // если ключ текущего элемента равен искомому
7. {
8. TArrayTable<TValue>::currNum = i; // присваиваем текущему номер найденного элемента
9. return true; // нашли элемент, возвращаем true
10. }
11. }
12. TArrayTable<TValue>::currNum = TTable<TValue>::DataCount; // прошли по всем элементам, не нашли искомый
13. return false; // возвращаем false
14. }
15. bool Insert(TRecord<TValue> tr)
16. {
17. if (TArrayTable<TValue>::IsFull()) // если таблица заполнена, бросаем исключение
18. throw "TableIsFull";
19. if (!Find(tr.GetKey())) // если этого элемента нет в таблице, помещаем
20. {
21. TArrayTable<TValue>::arr[TArrayTable<TValue>::currNum] = tr; // приравниваем текущий
22. TTable<TValue>::DataCount++; // увеличиваем число элементов в таблице
23. return true; // всё ок, возвращаем true
24. }
25. return false; // в противном случае, возвращаем false
26. }
27. void Delete(string k)
28. {
29. if (TArrayTable<TValue>::IsEmpty()) // таблица пуста, бросаем исключение
30. throw "TableIsEmpty";
31. if (Find(k)) // ищем элемент
32. {
33. TArrayTable<TValue>::arr[TArrayTable<TValue>::currNum] = TArrayTable<TValue>::arr[TTable<TValue>::DataCount - 1]; // сдвигаем элементы после него на единицу назад
34. TTable<TValue>::DataCount--; // уменьшаем число элементов
35. }

42 }

# Заключение

В данной лабораторной работе мне удалось:

* Реализовать класс для элементов прсомотровой таблицы
* Реализовать класс для просмотровой таблицы
* Продемонстрировать пример использования данного класса
* Написать тесты на основе Google Tests для проверки корректной работы данного класса

# Литература

Интернет-источники:

1. <https://ru.wikipedia.org/wiki/Структура_данных>
2. <https://habr.com/ru/post/232009/>

Книги:

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.
2. A.O. Грудзинский. Методы программирования, Издательство Нижегородского госуниверситета, 2006.