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 ЛАБОРАТОРНОЙ РАБОТЕ**

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

**Выполнил:** студент группы 381706-2

Мышкин Андрей Александрович

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Руководитель:**

ассистент кафедры МОСТ ИИТММ,

Лебедев Илья Геннадьевич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

Нижний Новгород

2019

**Содержание**

1. Введение………………………………………………………………………...3

2. Постановка задачи……………………………………………………...………4

3. Руководство пользователя…………………………………………...………...5

4. Руководство программиста…………………………………………...………..7

4.1. Описание структур данных…..……………………………………….7

4.2. Описание алгоритмов………………………………………………….8

4.3. Описание структуры программы….……………………………..…...8

6. Заключение…………………………………………………………………….10

7. Литература…………………………………………………………………….11

**Введение**

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.

Основной операцией при работе с таблицами является операция доступа к записи по ключу. Она реализуется процедурой поиска. Поскольку поиск может быть значительно более эффективным в таблицах, упорядоченных по значениям ключей, довольно часто над таблицами необходимо выполнять операции сортировки. Эти операции рассматриваются в следующих разделах.

Объективным критерием, позволяющим оценить эффективность того или иного алгоритма, является, так называемый, порядок алгоритма. Порядком алгоритма называется функция O(N), позволяющая оценить зависимость времени выполнения алгоритма от объема перерабатываемых данных (N - количество элементов в массиве или таблице). Эффективность алгоритма тем выше, чем меньше время его выполнения зависит от объема данных.

**Постановка задачи**

Основная задача – это реализовать структуру хранения данных такой, как класс просмотровая таблица. Необходимым условием реализации класса будет написание основных операций, проводимых над таблицей. В них входят:

* Инициализация таблицы
* Включение указанного элемента совместно с его ключом
* Удаление указанного элемента из таблицы по его ключу
* Поиск элемента через линейный или бинарный поиск

**Руководство пользователя**

При запуске тестирование структуры данных типа просмотровая таблица пользователю будет дана возможность заполнять таблицу самому.

Для облегчения понимания работы с данной структурой все действия по ходу работы будут поддерживаться небольшими комментариями.

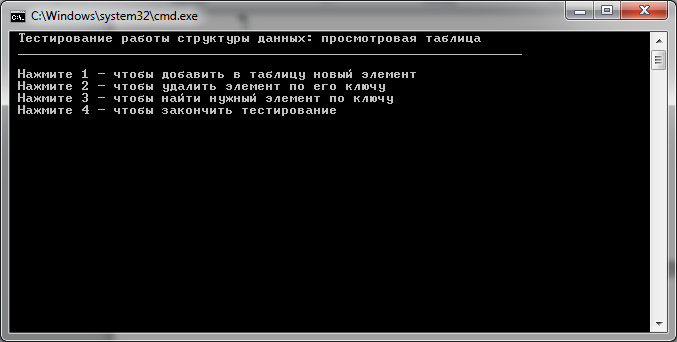


Рисунок 1. Начало тестирования. Сразу после запуска

На экране будут представлены возможные действия, проводимые с таблицей. Так как изначально таблица пуста, пользователь первым делом должен положить в неё какие-то значения, указав при этом к каждому свой ключ.

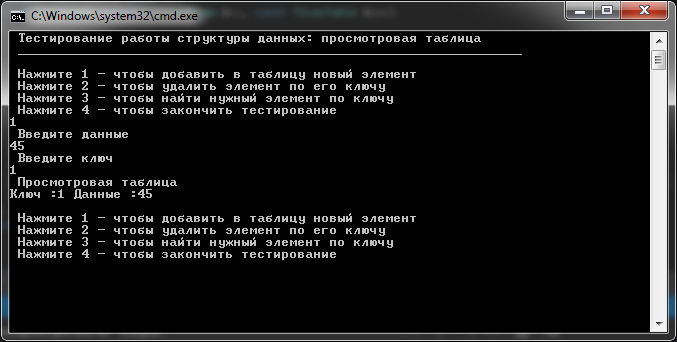


Рисунок 2. Начальное заполнение таблицы

Введем ещё одну переменную, и затем попробуем её удалить.

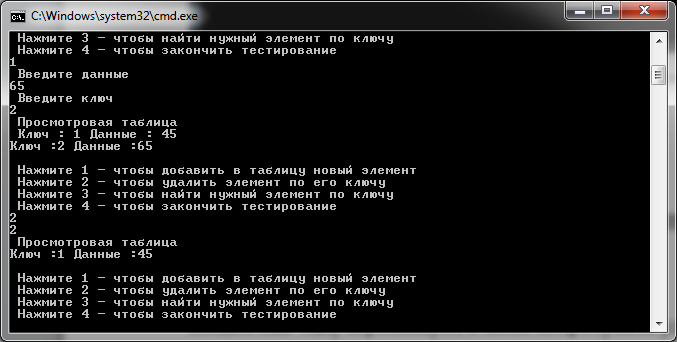


Рисунок 3. Удаление элемента по ключу

Все прошло хорошо. Для завершения тестирования нажимаем цифру «4».

**Руководство программиста**

**Описание структур данных.**

**Класс TKey**

Поля:

char\* key – значение самого ключа

int length – длина у ключа

Методы:

TKey(char\* kkey = "") – конструктор класса

TKey(const TKey &k) – конструктор копирования

void SetKey(char\* kkey) – метод класса, задающий значение ключа

int GetLength() – метод класса, возвращающий длину ключа

char\* GetKey() – метод класса, возвращающий сам ключ

**Класс TTabRecord**

Поля:

TKey key – значение ключа

ValType\* value – данные, на которые указывает ключ

bool needToDel – булевая переменная, определяющая необходимость удаления \*value после удаления записи

Методы:

TTabRecord(TKey k = "", ValType\* val = NULL, bool del = true) – конструктор класса

TTabRecord(TTabRecord<ValType> &tko) – конструктор копирования

void SetKey(TKey k) – метод класса, вкладывающий значение ключа

void SetValue(ValType\* val) – метод класса, вкладывающий данные

TKey GetKey() – метод класса, возвращающий ключ

ValType\* GetVal() – метод класса, возвращающий данные

**Класс TScanTable**

Поля:

int dataC – количество существующих записей в таблице

int tableSize – максимально возможное число записей в таблице

TTabRecord<ValType>\*\* pRecord – массив записей под таблицу

Методы:

int Add(TTabRecord<ValType> \*tko) – метод класса, добавляющий запись в таблицу

void Delete(TKey k) – метод класса, удаляющий запись из таблицы по ключу

int GetTableSize() – метод класса, возвращающий размер таблицы

int GetDataC() – метод класса, возвращающий число записей в данный момент

TTabRecord<ValType>\* Search(TKey k) – метод класса, осуществляющий поиск записи по ключу

ValType\* operator[] (TKey k) – метод класса, осуществяющий поиск по ключу и возвращает данные, а в случае если не находит такой ключ, создает пустую запись с этим ключом

**Описание алгоритмов.**

Добавление новой записи.

Если максимальный размер таблицы превышен, выходим с исключением. Иначе добавляем указанную запись в конец таблицы и возвращаем новое значение количества записей в таблице.

Поиск записи в таблице.

В цикле от 0 до количества записей в таблице пытаемся найти совпадения с указанным ключом и ключами в таблице. Если находим такой ключ, то возвращаем эту запись, иначе возвращаем NULL.

**Описание структуры программы.**

table – модуль с тестированием данной структуры хранения данных. Включает в себя main.cpp

tablelib – модуль с реализацией всех необходимых классов, такие как: TKey, TTabRecord, TScanTable

tabletest – модуль, содержащий тесты данного класса для прохождения их с помощью использования Google C++ Testing Framework

**Заключение**

В ходе проведение данной лабораторной работы была создана и протестирована такая структура хранения данных как просмотровая таблица. В классе были реализованы все необходимые операции, такие как: инициализация таблицы, включение указанного элемента совместно с его ключом, удаление указанного элемента из таблицы по его ключу, поиск элемента через линейный или бинарный поиск. Были усвоены все тонкости, с которыми пришлось столкнуться при выполнении данной лабораторной работы.

Также были освоены инструменты разработки программного обеспечения, как система контроля версий Git и фрэймворк для разработки автоматических тестов Google Test.

**Литература**

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015
2. Таблицы в программировании - <https://life-prog.ru/view_struktura_dannih.php?id=8>