

Федеральное агентство по образованию Российской Федерации
Государственное образовательное учреждение
Высшего профессионального образования
Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий математики механики

ТАБЛИЦЫ НА ДЕРЕВЬЯХ

Отчет по лабораторной работе

Выполнил:

студент ИИТММ гр. 381903-3

Алилуев А.О. _____

Проверил:

ассистент каф. МОСТ, ИИТММ

Лебедев И.Г. _____

Содержание

1.Введение	3
2.Постановка целей и задач	4
3.Руководство пользователя	5
4.Руководство программиста	7
4.1.Описание структуры программы	7
4.2.Описание структур данных	8
4.3.Описание алгоритмов	10
5.Заключение	12
6.Литература	13

1. Введение

Сортировка элементов в таблицах значительно ускоряет процесс поиска по ней, но тем не алгоритмы вставки элемента и удаления его из таблицы либо становятся еще дольше, либо могут не сильно выиграть в производительности. Для решения данной проблемы был придуман метод хранения таблиц в виде отсортированного бинарного дерева, при удачном заполнении которого найти нужный элемент, вставить или удалить его, можно очень быстро. В каждая вершина такого дерева представляет собой элемент таблицы, причем элементы располагаются строго в определенном порядке: левый потомок – элемент, ключ которого лексикографически меньше ключа его предка, а правый потомок – элемент, ключ которого лексикографически больше ключа его предка. Из этого свойства, при правильном обходе дерева, можно сделать отсортированную таблицу.

2. Постановка целей и задач

Целью лабораторной работы является создание структуры хранения типа «Таблица на деревьях» и методов работы с ней, таких как:

- Добавление элементов в таблицу;
- Удаление элементов из таблицы;
- Получение доступа к элементу.

Для реализации алгоритмов будет использоваться 3 класса:

- String;
- TTreeElem;
- TTreeTable.

Классы TTreeElem и TTreeTable являются шаблонными.

Для проверки правильности работы этих классов будут написаны тесты с использованием фреймворка Google Test, а также тестовый образец программы, которая будет использует класс список.

3.Руководство пользователя

После запуска программы пользователя встречает консольное окно (рис. 1):

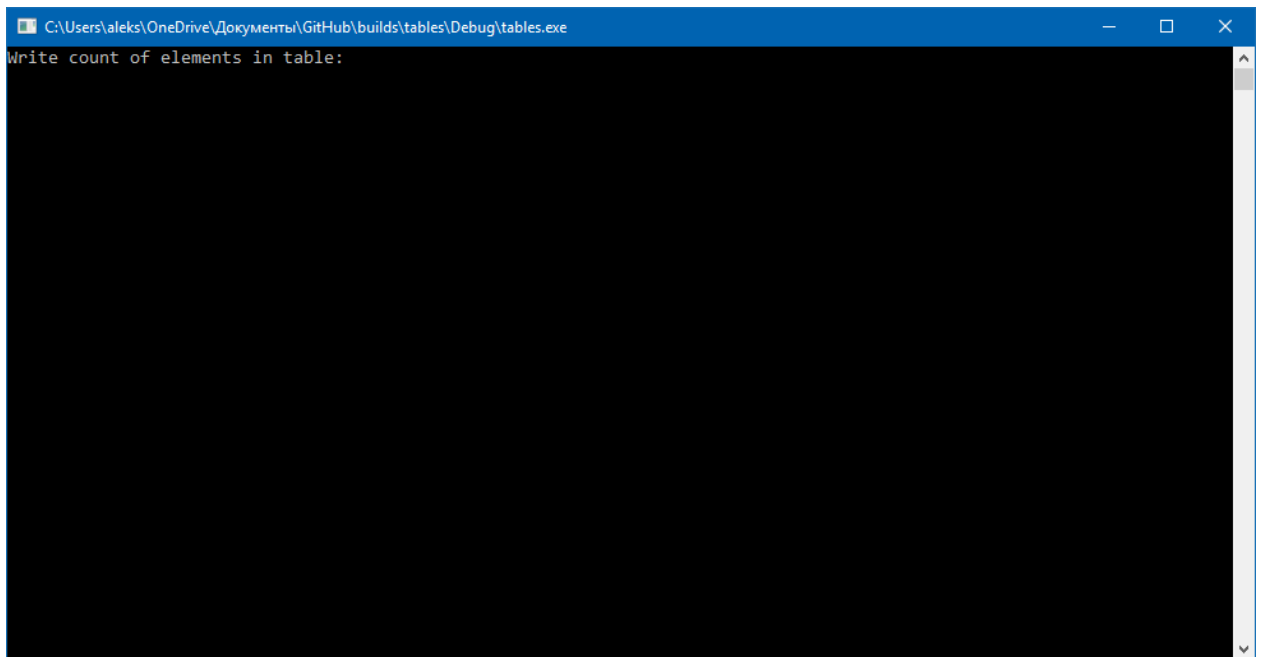


рис. 1 (вывод программы тестирования таблиц на деревьях для пользователя)

в которой сначала от пользователя требуется ввести количество элементов в таблице, а затем заполнить поля этой таблицы (1 поле – ключ, 2 поле – целочисленное число, которое будет хранить эта ячейка) (рис. 2).

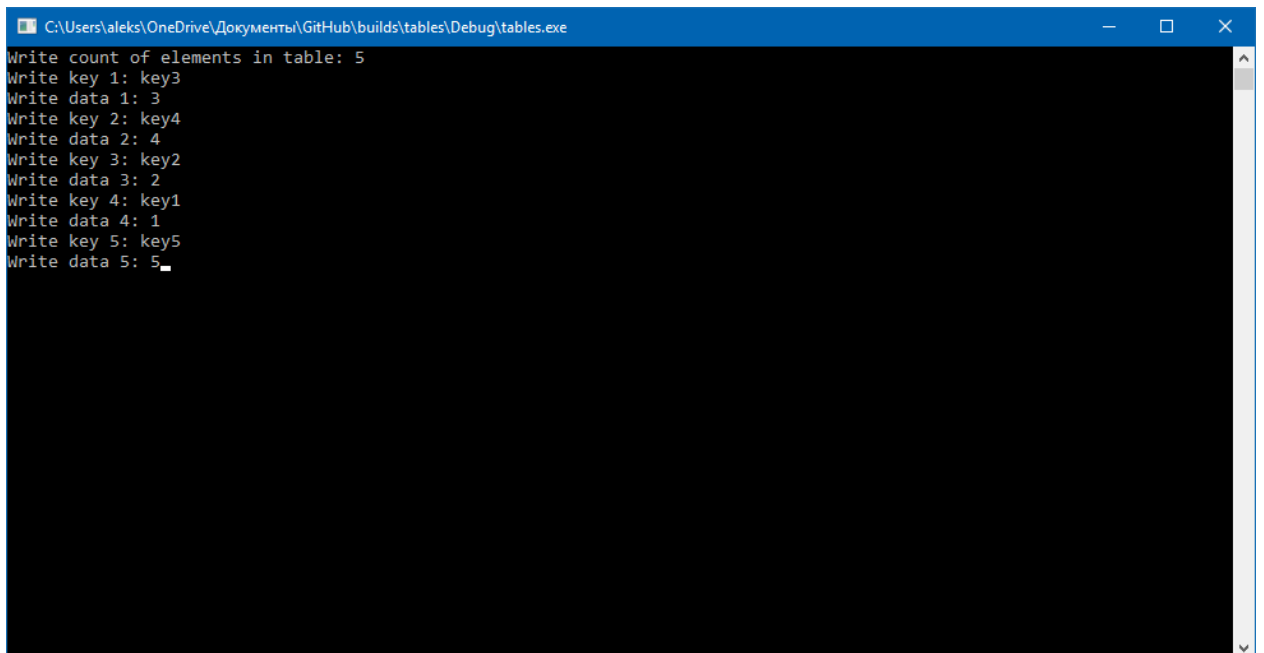
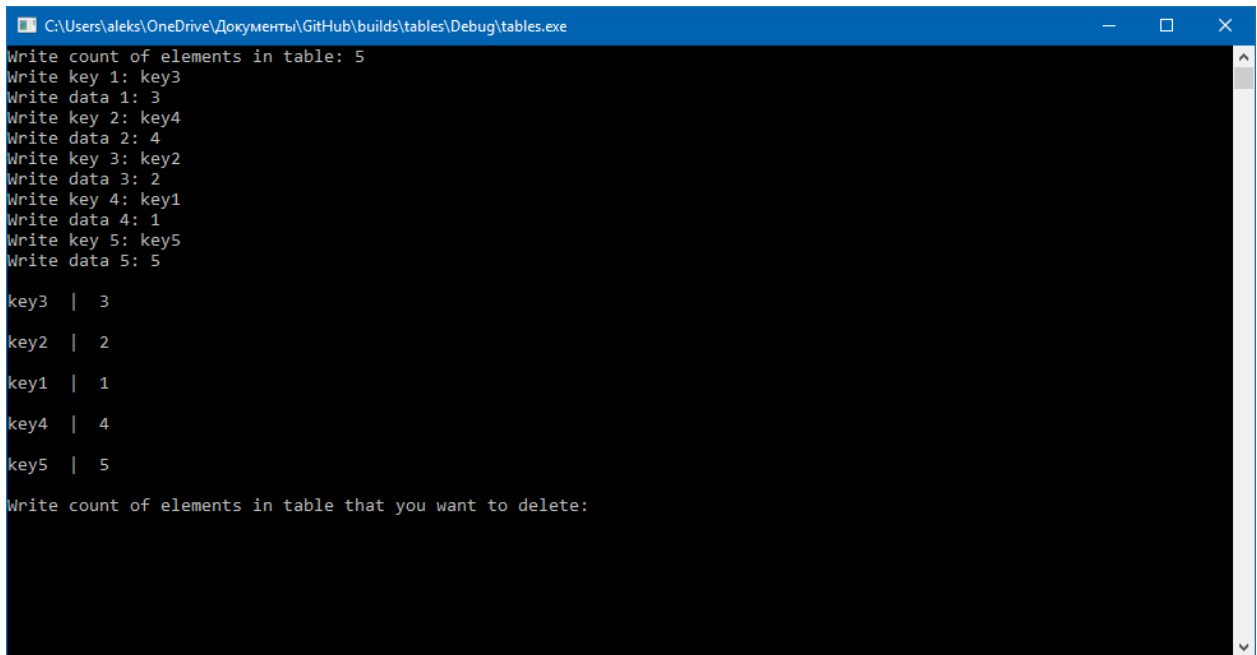


рис. 2 (заполнение полей таблицы)

Далее полученная таблица будет выведена для пользователя и программа предложит удалить некоторое количество элементов, нужно ввести их количество (рис. 3).



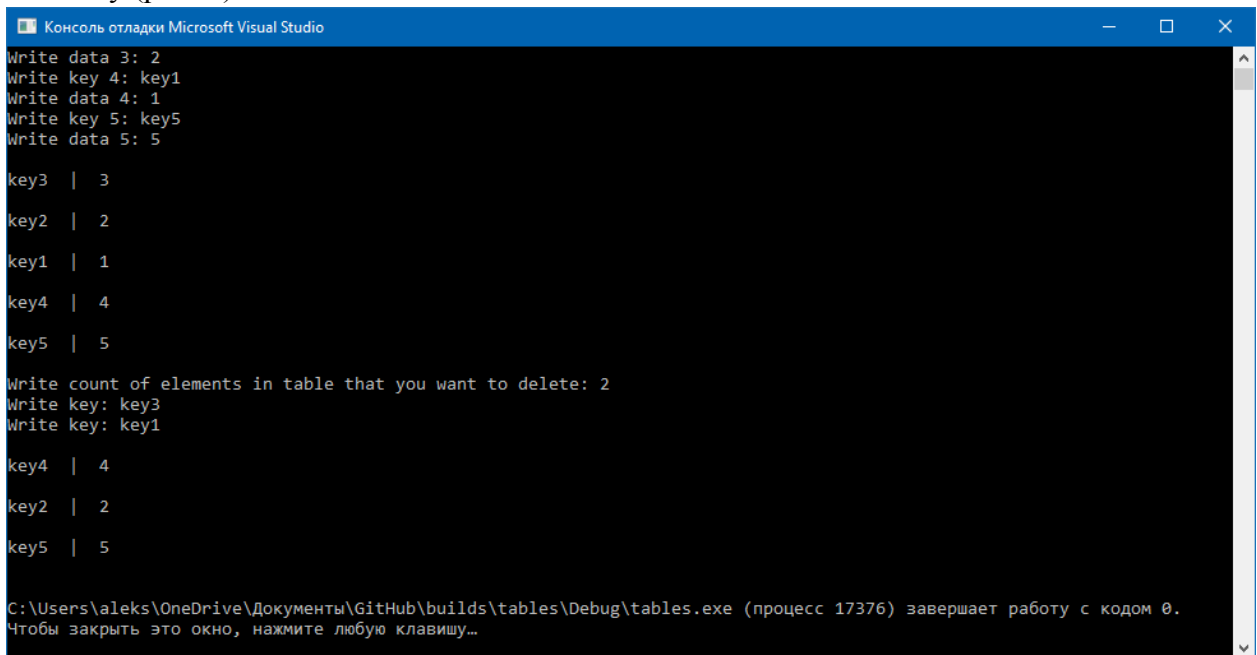
```
C:\Users\aleks\OneDrive\Документы\GitHub\builds\tables\Debug\tables.exe
Write count of elements in table: 5
Write key 1: key3
Write data 1: 3
Write key 2: key4
Write data 2: 4
Write key 3: key2
Write data 3: 2
Write key 4: key1
Write data 4: 1
Write key 5: key5
Write data 5: 5

key3 | 3
key2 | 2
key1 | 1
key4 | 4
key5 | 5

Write count of elements in table that you want to delete:
```

рис. 3 (вывод таблицы на экран и запрос на ввод количества удаляемых элементов)

В конце программа попроси ввести ключи удаляемых элементов и выведет конечную таблицу (рис. 4).



```
Консоль отладки Microsoft Visual Studio
Write data 3: 2
Write key 4: key1
Write data 4: 1
Write key 5: key5
Write data 5: 5

key3 | 3
key2 | 2
key1 | 1
key4 | 4
key5 | 5

Write count of elements in table that you want to delete: 2
Write key: key3
Write key: key1

key4 | 4
key2 | 2
key5 | 5

C:\Users\aleks\OneDrive\Документы\GitHub\builds\tables\Debug\tables.exe (процесс 17376) завершает работу с кодом 0.
Чтобы закрыть это окно, нажмите любую клавишу...
```

рис. 4 (ввод ключей и удаление элементов таблицы)

4.Руководство программиста

4.1.Описание структуры программы

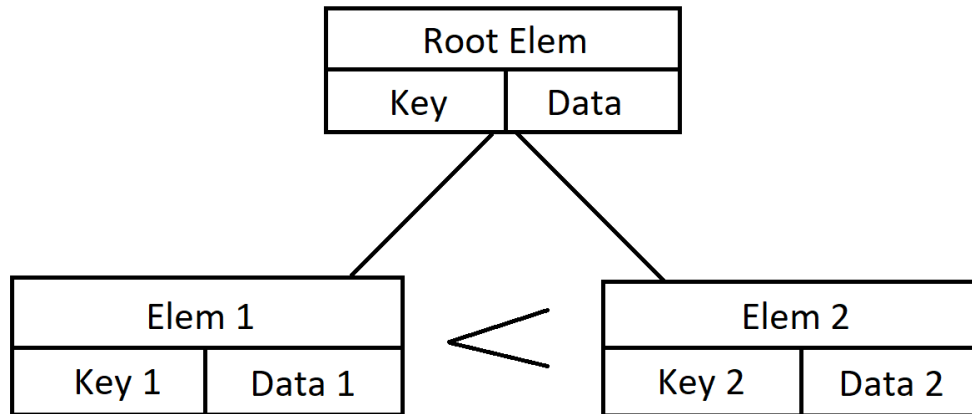


Таблица на деревьях будет реализована как набор элементов таблицы, связанных указателями. Каждый элемент включает в себя ключ, значение, а также указатели на левого и правого потомка и родителя. Каждый ключ левого потомка меньше ключа правого потомка. Это обеспечивает упорядоченность таблицы.

Таким образом для реализации алгоритмов будет использовано 3 класса:

- Класс «Строка» (String).
- Класс «Элемент Таблицы» (TTreeElem), который будет использовать класс String.
- Класс «Таблица» (TTreeTable), который использует класс TTreeElem.

А также проект использующий фреймворк Google Test, для проверки правильности работы этих классов и тестовый проект, который будет показываться пользователю.

Класс String:

Класс строка реализует функции работы с массивом символов, такие как: сравнение, присвоение, доступ к элементам массива.

Класс TTreeElem:

Класс элемент таблицы содержит реализацию работы с элементами. В нем реализованы такие методы, как: сравнение элементов, доступ к ключам и данным.

Класс TTreeTable:

Класс просмотрные таблицы содержит реализацию работы с таблицами. В нем реализованы такие методы, как: положить элемент в таблицу, удалить элемент из таблицы, получить значение по ключу и др.

Класс gtest:

Класс gtest реализует тестирование классов String, TTreeElem и TTable, по средствам фреймворка Google Test. Тесты пишутся для каждого метода классов, каждого ветвления этих методов и для всех возможных исключений этих методов.

Проект table:

В данном проекте реализован пример использования таблиц, показанный пользователю.

4.2.Описание структур данных

Класс String:

`char*` str – указатель на массив для хранения строки;

`int` count – переменная для хранения числа элементов в строке.

Класс TTreeElem:

`template< class T >` - шаблон класса T

`TTreeElem<T>*` left – указатель на левого потомка;

`TTreeElem<T>*` right – указатель на правого потомка;

`TTreeElem<T>*` parent – указатель на предка;

`String` key – ключ элемента;

`T` data – хранимые данные.

Класс TTable:

`template< class T >` - шаблон класса T

`static TTreeElem<T>` st – статический пустой элемент;

`TTreeElem<T>*` node – указатель на корень дерева;

`int` count – текущее количество занятых элементов таблицы.

Описание методов:

Метод:	Описание:
<code>String::String()</code>	Конструктор по умолчанию для класса String.
<code>String::String(const char* _str)</code>	Конструктор с параметром для класса String, который принимает указатель на массив символов.
<code>String::String(const String& _str)</code>	Конструктор копирования класса String.
<code>String::~~String()</code>	Деструктор класса String.
<code>int String::GetCount() const</code>	Метод, который возвращает количество символов в строке.
<code>String& String::operator=(const String& _str)</code>	Перегрузка оператора присваивания.
<code>bool String::operator==(const String& _str) const</code>	Перегрузка оператора равно для строки.
<code>bool String::operator!=(const String& _str) const</code>	Перегрузка оператора не равно для строки.
<code>bool String::operator<(const String& _str) const</code>	Перегрузка оператора меньше для строки.
<code>bool String::operator>(const String& _str) const</code>	Перегрузка оператора больше для строки.
<code>char& String::operator[](const int pos) const</code>	Возвращение элемента массива символов по его номеру.
<code>TTreeElem<T>::TTreeElem()</code>	Конструктор по умолчанию для элемента таблицы.
<code>TTreeElem<T>::TTreeElem(const T& _data, const String& _key, TTreeElem<T>* _left, TTreeElem<T>* _right, TTreeElem<T>* _parent)</code>	Конструктор с параметрами, который принимает хранимый элемент, его ключ, указатели на потомков и предка.
<code>TTreeElem<T>::TTreeElem(const TTreeElem<T>& elem)</code>	Конструктор копирования для элемента таблицы.
<code>TTreeElem<T>::~~TTreeElem()</code>	Деструктор для элемента таблицы.
<code>TTreeElem<T>& TTreeElem<T>::operator=(const TTreeElem<T>& elem)</code>	Перегрузка оператора присваивания для элемента.
<code>bool TTreeElem<T>::operator==(const TTreeElem<T>& elem) const</code>	Перегрузка оператора равно для элемента таблицы.
<code>bool TTreeElem<T>::operator!=(const TTreeElem<T>& elem) const</code>	Перегрузка оператора не равно для элемента таблицы.
<code>T& TTreeElem<T>::GetData()</code>	Возвращает значение, которое хранится в элементе таблицы.
<code>void TTreeElem<T>::SetData(const T& _data)</code>	Метод, который позволяет изменить значение, которое хранится в элементе таблицы.
<code>String& TTreeElem<T>::GetKey()</code>	Возвращает ключ элемента таблицы.
<code>void TTreeElem<T>::SetKey(const String& _key)</code>	Позволяет изменить ключ элемента таблицы.

<code>TTreeElem<T>*</code> <code>TTreeElem<T>::GetLeft()</code>	Метод, который возвращает левого потомка элемента дерева.
<code>TTreeElem<T>*</code> <code>TTreeElem<T>::GetRight()</code>	Метод, который возвращает правого потомка элемента дерева.
<code>TTreeElem<T>*</code> <code>TTreeElem<T>::GetParent()</code>	Метод, который возвращает предка элемента дерева.
<code>void</code> <code>TTreeElem<T>::SetLeft(TTreeElem<T>*</code> <code> _left)</code>	Метод, который позволяет присвоить новое значение левому потомку элемента дерева.
<code>void</code> <code>TTreeElem<T>::SetRight(TTreeElem<T></code> <code> * _right)</code>	Метод, который позволяет присвоить новое значение правому потомку элемента дерева.
<code>void</code> <code>TTreeElem<T>::SetParent(TTreeElem<T></code> <code> >* _parent)</code>	Метод, который позволяет присвоить новое значение предку элемента дерева.
<code>TTreeElem<T> TTreeTable<T>::st;</code>	Инициализация статического поля класса
<code>TTreeTable<T>::TTreeTable()</code>	Конструктор по умолчанию таблицы на деревьях.
<code>TTreeTable<T>::TTreeTable(const</code> <code> TTreeTable<T>& table)</code>	Конструктор копирования для таблицы на деревьях.
<code>TTreeTable<T>::~~TTreeTable()</code>	Деструктор таблицы.
<code>int TTreeTable<T>::GetCount() const</code>	Возвращает текущее количество элементов в таблице.
<code>TTreeElem<T>*</code> <code>TTreeTable<T>::GetNode() const</code>	Метод, который возвращает указатель на корень дерева.
<code>TTreeElem<T>*</code> <code>TTreeTable<T>::Add(TTreeElem<T>&</code> <code> elem)</code>	Метод, который позволяет добавить элемент в таблицу, принимая ссылку на этот элемент.
<code>void</code> <code>TTreeTable<T>::Del(TTreeElem<T>&</code> <code> elem)</code>	Метод, который позволяет удалить элемент из таблицы по ссылке на этот элемент.
<code>bool TTreeTable<T>::Del(const String&</code> <code> key)</code>	Метод, который позволяет удалить элемент из таблицы по ключу.
<code>TTreeElem<T>&</code> <code>TTreeTable<T>::Search(const String&</code> <code> key) const</code>	Метод, осуществляющий поиск элемента в таблице по его ключу.
<code>T& TTreeTable<T>::operator[](const</code> <code> String& key) const</code>	Метод, который возвращает значение элемента по ключу.

4.3.Описание алгоритмов

Подробное описание некоторых методов

Добавление элемента по ссылке на этот элемент:

- Если это первый элемент в таблице, то корню дерева присваивается этот элемент и количество элементов увеличивается на 1, возвращается указатель на корень;
- Если в таблице есть хотя бы 1 элемент, то в цикле для этого элемента ищется нужная позиция и запоминается его предок, если такой элемент уже лежит в таблице, то выдается ошибка.
- Далее в зависимости от того, каким потомком (правом или левом) является новый элемент таблицы, он создается по соответствующему указателю у его родителя, количество элементов в таблице увеличивается на 1, и возвращается указатель на новый элемент.

Удаление элемента таблицы по ссылке на элемент:

- Если у элемента есть и правый и левый потомки, то:
 - Выясняется каким потомком является данный элемент;
 - Ставим соответствующему потомку родителя в соответствии правый потомок элемента;
 - Идем до последнего левого потомка правого потомка исходного элемента, а затем ставим ему в левые наследники левый потомок исходного элемента;
 - Удаляем нужный элемент;
- Если у элемента есть хотя бы один наследник, то:
 - Выясняется каким потомком является данный элемент;
 - Ставим соответствующему потомку родителя в соответствии правый потомок элемента;
 - Удаляем нужный элемент;
- Если потомков нет, то удаляем элемент.
-

5. Заключение

В заключении можно сказать, что все поставленные цели и задачи были выполнены, а именно: созданы классы «String», «TTreeElem» и «TTreeTable» с реализованными методами добавления, удаления и доступа к элементам таблицы, а также написаны к ним тесты, и они успешно пройдены.

6.Литература

- Учебные материалы к учебному курсу «Методы программирования» - Гергель В.П.