Министерство науки и образования РФ
Федеральное государственное автономное образовательное
учреждение высшего профессионального образования
«Санкт-Петербургский государственный электротехнический
университет «ЛЭТИ» им. В. И. Ульянова (Ленина)»
(СПбГЭТУ «ЛЭТИ»)
Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

Отчёт
по лабораторной работе № 2
на тему:
“Деревья двоичного поиска”

по дисциплине “Алгоритмы и структуры данных”

Вариант 19

Выполнил студент гр. 4306: Табаков А.В.
Принял: Колинько П.Г.

Санкт-Петербург
2016

Цель

Получить практические навыки работы с деревьями двоичного поиска.

Задание

Переделать программу, составленную по теме «Хеш-таблицы», под использование деревьев двоичного поиска, а именно программу вычисляющую пятое множество по четырём заданным по формуле E=A&B|C&D. Вид дерева двоичного поиска: 2-3 дерево.

Контрольные примеры

Контрольные примеры представлены в таблице 1.

Таблица. 1. Контрольные примеры

№	Исходные множества				Результат
	A	B	C	D	E
1	1 12 17 20 33 48 59	0 1 17 24 25 26 27 50	2 13 14 16 18 19 27 50	0 17 18 19 44 45 46 50	1 17 18 19 50
2	1 2 3 4	3 4 5 6	5 6 7 8	7 8 9 10	3 4 5 6
3	0	0 1	8 5	1 2	0

Демонстрация работы

Демонстрация работы программы представлена на Рис. 1.

Способ представления: 2-3 дерево. Код программы см. приложение.

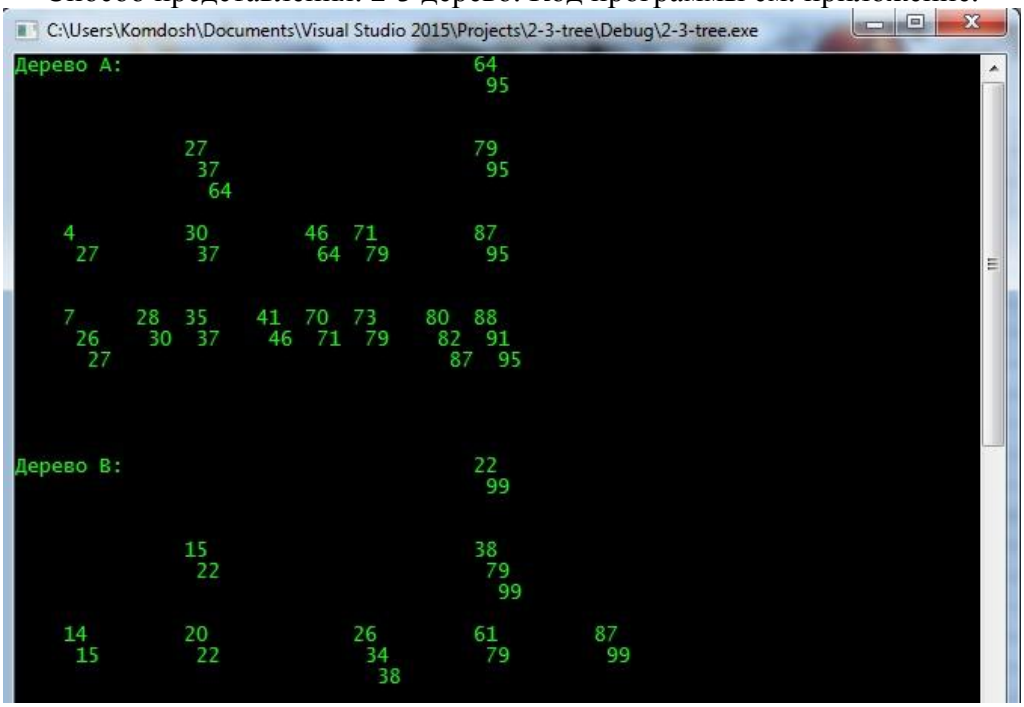

Рис.1. Демонстрация программы(начало)

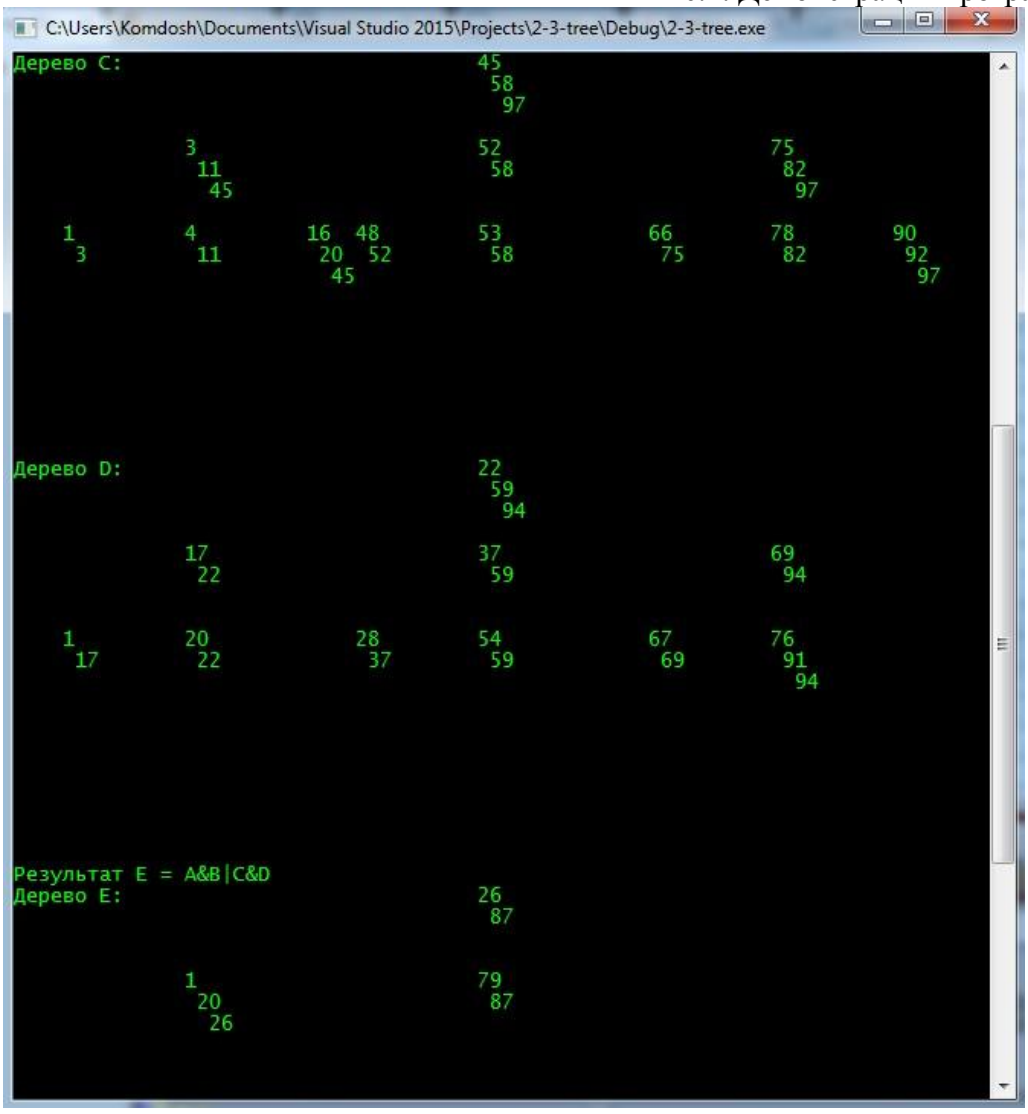

Рис.1. Демонстрация программы(конец)

Временная сложность

Временная сложность представлена в таблице 2.

Таблица. 2. Временная сложность

Функция	Средняя	Худшая
Вставка	O(log n)	O(log n)
Удаление	O(log n)	O(log n)
Поиск	O(log n)	O(log n)
Операция &	O(n)	O(n)
Операция |	O(n)	O(n)

Описание алгоритмов

Поиск:

Пока существует указатель на узел проверяем узел:

Если число совпадает с искомым, возвращаем правду.

Если искомое число меньше, чем в узле, идём вниз, иначе вправо.

Операция &:

Пока есть узел правого и левого дерева выполняем:

Пока ключ узла слева меньше ключа узла справа продолжаем обход по левому дереву.

Пока ключ узла справа меньше ключа узла слева продолжаем обход по правому дереву.

Если ключ левого дерева совпадает с ключом правого дерева добавляем узел временному дереву и делаем шаг по правому и левому дереву.

Операция |:

Пока есть узел правого или левого дерева выполняем:

Если есть узел правого и левого дерева:

Если ключ узла левого дерева совпадает с ключом узла правого дерева, то добавляем узел временному дереву и делаем шаги по левому и правому деревьям. Иначе: пока ключ слева меньше ключа справа, добавляем узлы левого дерева, потом пока ключ справа меньше ключа слева, добавляем узлы правого дерева.

Если нет правого дерева, добавляем все узлы левого дерева и наоборот, если нет левого дерева добавляем все узлы правого дерева.

Ответы на контрольные вопросы

1. Симметричной разметкой, чтобы вес узла слева не превышал вес других узлов.
2. В худшем случае дерево вырождается в линейный список и это происходит в половине случаев, полностью сбалансированное дерево (лучший случай) появляется только в одном из 2^n случаев, где n – количество узлов. Поэтому не имеет смысла рассматривать временную сложность лучшего случая.
3. При поэлементной вставке в дерево упорядоченной последовательности.
4. Можно с помощью автобалансировки.
5. Алгоритмом слияния временная сложность двуместной операции может быть O(n).
6. Обычно время поиска, вставки, удаления – логарифмическое, но в вырожденных деревьях время становится линейным.
7. Может, если применим алгоритм слияния, но не сбалансируем дерево.
8. Сами ключи в дереве лучше не хранить, но можно обойти это ограничение, хранив в узле множитель количества ключей. Также, если дубликаты должны быть представлены явно, в узлах можно хранить указатель на начало списка, как в методе цепочек переполнений хеш-таблиц.

9. Деревья двоичного поиска, т.к. требуют хранения в узлах дополнительной информации для балансировок.
10. Хеш-таблица, т.к. у неё средняя временная сложность O(1), а у ДДП - O(log n).
11. Применять балансировку. Сбалансированное ДДП имеет меньшую временную сложность.

Вывод

При выполнении лабораторной работы были получены практические навыки работы с деревьями двоичного поиска на языке программирования «C/C++».

По моему мнению для вычисления множества A&B|C&D, 2-3 дерево не самый удачный выбор, но и не плохой выбор, т.к. те же хеш-таблицы, в среднем показывают гораздо лучшее время, т.к. в ДДП сложности порядка O(log n), а в хеш-таблицах O(1), но хеш-таблицы проигрывают в худших случаях, а также хеш-таблицы не защищены от вырождения, в то время как 2-3 деревья всегда сбалансированы.

Список используемых источников

- Алгоритмы и структуры данных: методические указания к лабораторным работам, практическим занятиям и курсовому проектированию. Ч.2 Вып. 1601 / сост.: П.Г. Колинько. - СПб.: Изд-во СПБГЭТУ "ЛЭТИ", 2016. - 48 с.
- Освой C++ самостоятельно за 21 день. Сиддхартха Рао. 688 стр., с ил.; ISBN 978-5-8459-1825-3; 7 издание.
- http://stackoverflow.com – Сайт вопросов и ответов по программированию.
- http://cyberforum.ru – Форум программистов и сисадминов.
- http://gubsky.ru/study/5/soad/sh/26.htm - B-деревья

Приложение

Source.cpp – Код программы
Classes.h – Заголовочный файл программы