Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления (ИУ)

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6
«Обработка деревьев, хеш-функций»

Студент, группа **Буланый К., ИУ7-36Б**

2020 г.

Описание условия задачи

Построить ДДП, в вершинах которого находятся слова из текстового файла. Вывести его на экран в виде дерева. Сбалансировать полученное дерево и вывести его на экран. Построить хеш-таблицу из слов текстового файла. Использовать метод цепочек для устранения коллизий. Осуществить поиск введенного слова в ДДП, в сбалансированном дереве, в хеш-таблице и в файле. Сравнить время поиска, объем памяти и количество сравнений при использовании различных (4-х) структур данных. Если количество сравнений вхеш-таблице больше указанного (вводить), то произвести реструктуризацию таблицы, выбрав другую функцию.

Техническое задание

Входные данные:

1. **Имя файла с деревом:** строка, содержащая имя файла.
2. **Максимальное значение допустимых коллизий:** целое число.
3. **Слово для поиска:** строка, содержащая слово, которое планируется найти в дереве.

Выходные данные:

1. Изображение бинарного дерева.
2. Изображение сбалансированного бинарного дерева.
3. Хеш-таблица вершин дерева.
4. Характеристика моделирования.

Функция программы:

1. Вывод бинарного дерева.
2. Вывод сбалансированного бинарного дерева.
3. Вывод хеш-таблицы.
4. Вывод характеристика моделирования.

Обращение к программе: запускается из терминала.

Аварийные ситуации:

1. Некорректный ввод имени файла.
   На входе: имя файла, несуществующего в системе.
   На выходе: сообщение «Wrong file.»
2. Пустой файл в качестве аргумента программы.
   На входе: пустой файл.
   На выходе: сообщение «File is empty»

3. Некорректный ввод максимального числа коллизий.
   На входе: буква или, любой другой нечисловой символ или отрицательное число.
   На выходе: сообщение «Wrong number.»

Структуры данных

Реализация линейного односвязного списка:

```
typedef struct list
{
   char *value;
   struct list *next;
} list_t;
```

Поля структуры:

- ***char* *value –*** указатель на массив символов;
- ***struct list* *next –*** указатели на следующий элемент списка.

Реализация листа дерева:

```
typedef struct tree_node
{
   char *value;
   struct tree_node *left;
   struct tree_node *right;
} tree_node;
```

Поля структуры:

- ***char* *value –*** указатель на массив символов;
- ***struct tree_node* *left –*** указатель на левого потомка;
- ***struct tree_node* *right –*** указатель на правого потомка.

Реализация динамического массива (для реализации хеш-таблицы):

```
typedef struct
{
   tree_node **arr;
   int size;
   int mem_size;
} dynarr_t;
```

Поля структуры:

- ***tree_node* **arr –** указатель на массив указателей листов дерева;
- ***int* size –** фактическая ёмкость массива;
- ***int* mem_size –** максимальная ёмкость массива

Алгоритм

Сначала происходит балансировка дерева. Исходное дерево вытягивается в отсортированный односвязный список, затем происходил рекурсивное построение АВЛ-дерева. После постройки дерева строится хеш-таблица с помощью DJB2 хеширования по дереву. В случае, если количество коллизий превосходит допустимое, таблица самоперестраивается на основе Jenkins-At-One-Time хеширования и выводится характеристика моделирования.

Тесты

	Тест	Пользовательский ввод	Результат
1	Некорректный ввод имени файла	tree07.txt	Worng file.
2	Пустой файл	tree06.txt	File is empty.
3	Некорректный ввод максимального количества коллизий	X	Wrong number.
4	Некорректный ввод максимального количества коллизий	0	Wrong number.
5	Некорректный ввод максимального количества коллизий	-100	Wrong number.
6	Ввод несуществующего слова	Kek	Word “Hi” wasn’t found.
7	Ввод количества коллизий, большего, чем текущее максимальное	5	No need to remake the table.
8	Ввод количества коллизий, меньшего, чем текущее максимальное	1	Пересоздание хеш-таблицы

9	Корректный ввод всех характеристик	Корректные данные	Характеристика

Оценка эффективности

Поиск слова (в тактах процессора):

Количество элементов дерева	**Бинарное дерево**	**Сбалансированное бинарное дерево**	**Хеш-таблица**	**Файл**
20	1162	1015	867	11479
40	1239	1020	715	13984
80	1307	1083	638	28294
400	2122	1516	567	35447
800	1894	1541	498	55507

Объём занимаемой памяти (в байтах):

Количество элементов дерева	**Бинарное дерево**	**Сбалансированное бинарное дерево**	**Хеш-таблица (без коллизий)**	**Файл**
20	480	480	952	146
40	960	960	5016	293
80	1920	1920	15864	602
400	9600	9600	202200	2908
800	19200	19200	736792	5828

Контрольные вопросы

1. Что такое дерево?

Дерево – это рекурсивная структура данных, используемая для представления иерархических связей, имеющих отношение «один ко многим».

2. Как выделяется память под представление деревьев?

В виде связного списка.

3. Какие стандартные операции возможны над деревьями?

Обход дерева, поиск по дереву, включение в дерево, исключение из дерева.

4. Что такое дерево двоичного поиска?

Двоичное дерево поиска - двоичное дерево, для каждого узла которого сохраняется условие:
левый потомок больше или равен родителю, правый потомок строго меньше родителя (либо наоборот).

5. Чем отличается идеально сбалансированное дерево от АВЛ дерева?

У АВЛ дерева для каждой его вершины высота двух её поддеревьев различается не более чем на 1, а у идеально сбалансированного дерева различается количество вершин в каждом поддереве не более чем на 1.

6. Чем отличается поиск в АВЛ-дереве от поиска в дереве двоичного поиска?

Поиск в АВЛ дереве происходит быстрее, чем в ДДП.

7. Что такое хеш-таблица, каков принцип ее построения?

Хеш-таблицей называется массив, заполненный элементами в порядке, определяемом хеш-функцией. Хеш-функция каждому элементу таблицы ставит в соответствие некоторый индекс. Функция должна быть простой для вычисления,распределять ключи в таблице равномерно и давать минимум коллизий.

8. Что такое коллизии? Каковы методы их устранения?

Коллизия – ситуация, когда разным ключам хеш-функция ставит в соответствие один и тот же индекс. Основные методы устранения коллизий: открытое и закрытое хеширование. При открытом хешировании к ячейке по данному ключу прибавляется связанны список, при закрытом – новый элемент кладется в ближайшую свободную ячейку после данной.

9. В каком случае поиск в хеш-таблицах становится неэффективен?

Поиск в хеш-таблице становится неэффективен при большом числе коллизий –сложность поиска возрастает по сравнению с O(1). В этом случае требуется реструктуризация таблицы – заполнение её с использованием новой хеш-функции.

10. Эффективность поиска в АВЛ деревьях, в дереве двоичного поиска и в хеш-таблицах.

В хеш-таблице минимальное время поиска O(1). В АВЛ: O(log2n). В дереве двоичного поиска O(h), где h - высота дерева (от log2n до n).

Вывод

Использование хеш-таблицы всегда эффективно по времени, но очень неэффективно по памяти. В случае деревьев, АВЛ дерево немного выигрывает по времени поиска у несбалансированного дерева.