­­­­Лабораторная работа 7

Деревья, хеш-таблицы

Монахов Дмитрий ИУ7-34

Отчет

**Условие задачи:**

Используя бинарное дерево следующего выражения:

9+(8\*(7+(6\*(5+4)-(3-2))+1)),

и процедуру постфиксного обхода дерева, вычислить значение каждого узла и результат записать в его вершину. Получить массив, используя процедуру инфиксного обхода полученного дерева. Построить для этих данных дерево двоичного поиска (ДДП), сбалансировать его. Построить хеш-таблицу для значений этого массива. Осуществить поиск указанного значения. Сравнить время поиска, объем памяти и количество сравнений при использовании ДДП, сбалансированных деревьев и хеш-таблиц.

**Техническое задание**

**Аварийные ситуации**

Ошибка выделения памяти

Некорректная команда пользователя

**Предусмотрено**

Сообщение об ошибке выделения памяти

Сообщение о некорректной команде

**Входные данные**

Команда пользователя

**Выходные данные**

Двоичное дерево поиска

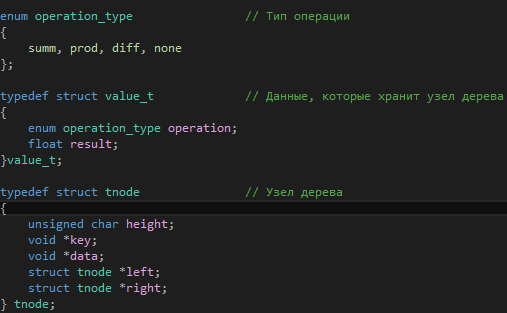
АВЛ-дерево

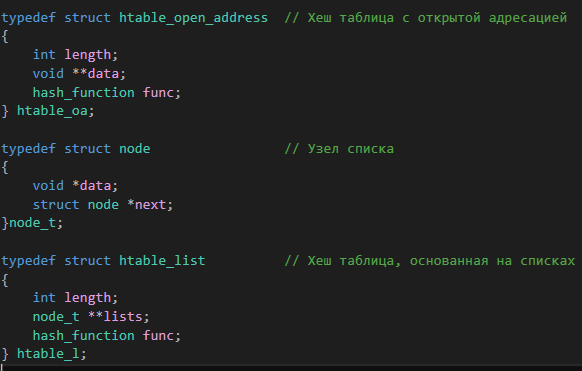
Хеш-таблица, основанная на списке

Хеш-таблица с открытой адресацией

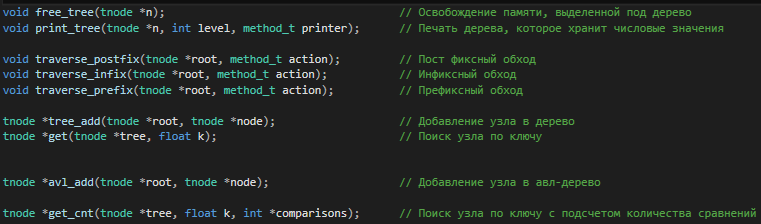
Сообщение об ошибке (если имеется)

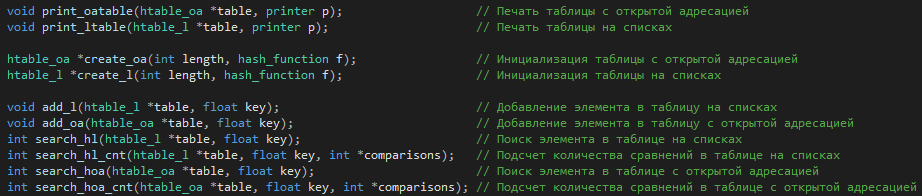
**Структуры данных**





**Функции**



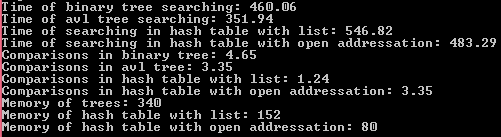


**Тесты**

1. Некорректная команда пользователя

**Анализ**

Сравним время поиска элемента, количество сравнений и занимаемую память в двоичном дереве поиска, АВЛ-дереве и хеш-таблице.

**Вывод программы:** 

Видно, что средние времена поиска у деревьев выше, чем у хеш-таблиц. Это связано с тем, что при использовании таблиц тратится время на вычисление хеш-функции.

Сбалансированность АВЛ дерева заметно снижает среднее время поиска элемента и количество сравнений.

По количеству сравнений выигрывает хэш-таблица, основанная на списках.

Деревья занимают больше памяти чем таблицы, т.к. таблицы проще организованы, а в деревьях некоторое количество памяти уходит на хранение ссылок на потомков каждого узла.

**Вывод**

Идеальная балансировка дает наименьшую высоту дерева, а так как высота дерева определяет длину пути поиска в нем, то, следовательно, и укорачивает поиск. Но поддержание идеальной сбалансированности дерева при включении или исключении элемента – это достаточно сложная процедура, которая требует дополнительных затрат времени и памяти.

Использование деревьев для поиска информации достаточно эффективно (трудоемкость – O(log2n)). Однако есть еще более эффективный метод хранения и поиска информации – хеш-таблицы.

В хеш-таблицах сложность алгоритма поиска выражается как O(1), потому что скорость вычисления хеш-функции не зависит от количества хранимых элементов, а только от сложности самой хеш-функции. Поэтому хеш-таблицы и используются для хранения больших объемов данных.