ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

*«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»

Институт металлургии, машиностроения и транспорта

**Курсовой проект**

по дисциплине «Программирование на языках высокого уровня»

**«AVL-дерево»**

Пояснительная записка

|  |  |  |
| --- | --- | --- |
| Выполнил  студент гр. 33335/2 | *(подпись)* | Марков М.Е. |
| Работу принял | *(подпись)* | Ананьевский М. С. |

Санкт-Петербург

2018 г.

**Формулировка задачи, которую решает алгоритм.**

АВЛ-дерево — сбалансированное по высоте [двоичное дерево поиска](https://ru.wikipedia.org/wiki/%D0%94%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_%D0%BF%D0%BE%D0%B8%D1%81%D0%BA%D0%B0): для каждой его вершины высота её двух поддеревьев различается не более чем на 1.При работе алгоритма реализуются добавление, удаление и поиск минимального ключа, а также балансировка дерева.

**Особенности алгоритма.**

Ключ любого узла дерева не меньше любого ключа в левом поддереве данного узла и не больше любого ключа в правом поддереве этого узла.

Для любого узла дерева высота его правого поддерева отличается от высоты левого поддерева не более чем на единицу.

В данной работе все значения ключей целочисленные.

**Реализация алгоритма.**

Алгоритм был реализован при помощи языка программирования C++. Узел дерева представлен структурой node, полями которой являются значение ключа в узле, высота дерева, указатель на структуру node для левой и правой ветви. Созданы конструкторы без параметров и с параметром k, которые создают узел со значением 0 и со значением переданного параметра k соответственно.

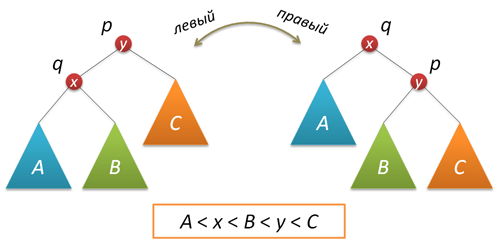
Отсутствие узлов слева или справа будем обнаруживать при помощи нулевого указателя в поле left и right соответственно. Для правильной работы программы необходимо реализовать следующую функцию:

Если на вход подан отсутствующий узел, то она возвращает 0, иначе в поле height узла записывает высоту дерева с корнем в узле.

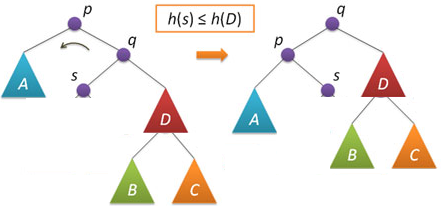
Функция bfactor возвращает разницу между высотой правой и левой ветви. По свойству AVL дерева он может принимать значения -1, 0, 1. При добавлении и удалении узлов может возникать ситуация, когда это условие нарушится. Для этого в программе предусмотрена функция балансировки дерева.

В качестве вспомогательной функции также выступает функция realheight, которая возвращает наибольшее значение высоты правой и левой ветви узла.

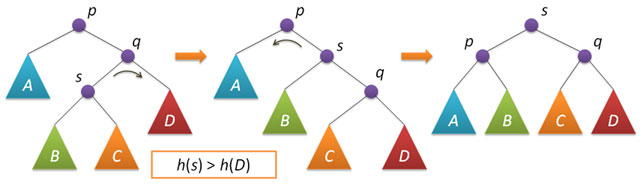
Балансировка узлов может быть осуществлена с помощью поворота вокруг узлов дерева. Правый поворот в программе реализован следующим образом: в функцию передаётся указатель на узел p, узлу q выбирается за левый от узла p. Затем левому узлу от p присваивается значение узла, правого от q. Затем объявляется, что узел p – это узел справа от q. Вычисляются новые высоты узлов p и q и возвращается узел q в качестве указателя на текущий узел. Левый поворот осуществлён аналогичным образом, что показано на рисунке ниже.



Непосредственно для исправления разбалансировки в узле p достаточно выполнить либо простой поворот влево вокруг p, либо большой поворот влево вокруг того же p. Простой поворот выполняется при условии, что высота левого поддерева узла q меньше или равна высоте его правого поддерева.



Большой поворот применяется при условии h(s)>h(D) и сводится в данном случае к двум простым — сначала правый поворот вокруг q и затем левый вокруг p.



Функция балансировки основана на проверке всех этих условий и возвращает тот же узел, который был подан на вход, но со сбалансированными ветвями.

Функция вставки ключа реализована с помощью прохода по дереву, сравнивая значение ключей в каждом из встретившихся узлов с тем, которое вставляется. В случае, если оно меньше, вызывается та же функция для левой ветви, иначе для правой. Если поданного узла не существует, то создаётся в куче новая структура, обозначающая узел, она связана с последним существующим узлом (указатель на неё хранится в поле right или left последнего узла перед тем, как она была создана. При возврате из рекурсии производится балансировка.

Удаление ключей происходит по следующей схеме: находится узел с заданным значением ключа k, затем если узел лист или не имеет правой ветви (указатель поля right нулевой), то в качестве текущего узла возвращается левый узел. Если правая ветвь у узла с заданным значением ключа имеется, то ищется минимальное значение в правой ветви (необходимо спускаться по дереву, держась только левых узлов до конца), узел с текущим значением q удаляется из кучи, на его место вставляется минимальный элемент правой ветви и дерево балансируется.

Анализ алгоритма

1. Время работы алгоритма

Высота h АВЛ-дерева с n ключами лежит в диапазоне от log2(n + 1) до 1.44 log2(n + 2) − 0.328. Основные операции над двоичными деревьями поиска (поиск, вставка и удаление узлов) линейно зависят от его высоты, что гарантирует логарифмическую зависимость времени работы этих алгоритмов от числа ключей, хранимых в дереве.

Расход памяти O(n).

Проверка времени работы алгоритма:

*Проверка функции вставки ключа(insert)*

При 5000 операциях время выполнения: https://pp.userapi.com/c846523/v846523167/140a16/RkLopVgDKps.jpg

При 10000 операциях время выполнения: https://pp.userapi.com/c846523/v846523167/140a0f/4-iffnsWgIQ.jpg

При 50000 операциях время выполнения: https://pp.userapi.com/c845123/v845123167/1446da/pWD90HbTsRE.jpg

При 100000 операциях время выполнения: https://pp.userapi.com/c845123/v845123167/144704/kSYgguQK6Sg.jpg

*Проверка функции удаления ключа(remove):*

При 5000 операциях время выполнения: https://pp.userapi.com/c846523/v846523167/140a16/RkLopVgDKps.jpg

При 10000 операциях время выполнения: https://pp.userapi.com/c845123/v845123167/144720/aDlCJZ0LrHI.jpg

При 50000 операциях время выполнения: https://pp.userapi.com/c845123/v845123167/144727/YI2dmD5N9No.jpg

При 100000 операциях время выполнения: https://pp.userapi.com/c845123/v845123167/144738/eooIffS1c9I.jpg

Применение алгоритма

АВЛ-деревья могут быть применены для упорядоченного хранения элементов ,вставки, поиска и удаления за время от log2(n + 1) до 1.44 log2(n + 2) − 0.328, что требуется, например, для баз данных.