86 Опишите основные определения, связанные с темой «Бинарные

деревья». Расскажите про особенности их обходов

Бинарное (двоичное) дерево (binary tree) – древовидная структура данных, в которой каждый узел имеет не более двух потомков (детей).

Бинарное дерево  – это упорядоченное дерево, каждая вершина которого имеет не более двух поддеревьев, причем для каждого узла выполняется правило: в левом поддереве содержатся только ключи, имеющие значения, меньшие, чем значение данного узла, а в правом поддереве содержатся только ключи, имеющие значения, большие, чем значение данного узла(рисунок 1).

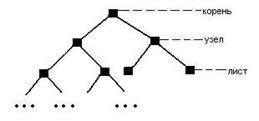


Рисунок 1- Бинарное дерево

Бинарное (двоичное) дерево – это динамическая структура данных, представляющее собой дерево, в котором каждая вершина имеет не более двух потомков ([рисунок](https://intuit.ru/studies/higher_education/3407/courses/504/lecture/11458?page=1#image.31.3) 2). Таким образом, бинарное дерево состоит из элементов, каждый из которых содержит информационное поле и не более двух ссылок на различные бинарные поддеревья. На каждый элемент дерева имеется ровно одна ссылка.

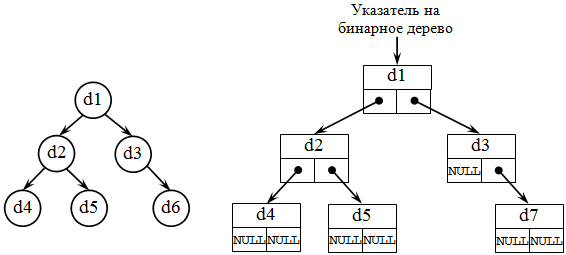


Рисунок 2- Бинарное дерево и его организация

Каждая вершина бинарного дерева является структурой, состоящей из четырех видов полей. Содержимым этих полей будут соответственно:

* информационное поле (ключ вершины);
* служебное поле (их может быть несколько или ни одного);
* указатель на левое поддерево;
* указатель на правое поддерево.

По степени вершин бинарные деревья делятся на [(рисунок 4](https://intuit.ru/studies/higher_education/3407/courses/504/lecture/11458?page=1#image.31.4)):

* строгие – вершины дерева имеют степень ноль (у листьев) или два (у узлов);
* нестрогие – вершины дерева имеют степень ноль (у листьев), один или два (у узлов).

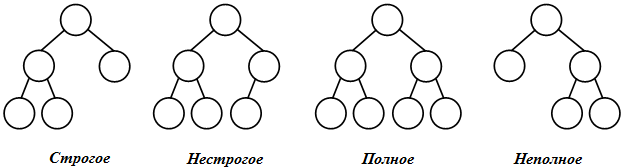


Рисунок 4- Классификация по степени вершин

В общем случае у бинарного дерева на k -м уровне может быть до 2k-1 вершин. Бинарное дерево называется полным, если оно содержит только полностью заполненные уровни. В противном случае оно является неполным.

Дерево называется сбалансированным, если длины всех путей от корня к внешним вершинам равны между собой. Дерево называется почти сбалансированным, если длины всевозможных путей от корня к внешним вершинам отличаются не более, чем на единицу.

Бинарное дерево может представлять собой пустое множество. Бинарное дерево может выродиться в список ([рисунок 5](https://intuit.ru/studies/higher_education/3407/courses/504/lecture/11458?page=1#image.31.5)).

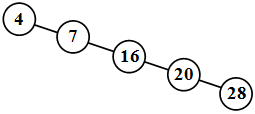


Рисунок 5- Список как частный случай бинарного дерева

Бинарное дерево является рекурсивной структурой, поскольку каждое его поддерево само является бинарным деревом и, следовательно, каждый его узел в свою очередь является корнем дерева. Узел дерева, не имеющий потомков, называется листом.

То есть двоичное дерево либо является пустым, либо состоит из данных и двух поддеревьев (каждое из которых может быть пустым). Очевидным, но важным для понимания фактом является то, что каждое поддерево в свою очередь тоже является деревом.

Каждый узел  в дереве задаёт поддерево, корнем которого он является.

У вершины n=(data, left, right) есть два ребёнка (левый и правый) left и right и, соответственно, два поддерева (левое и правое) с корнями left и right.

Свойство. Строго бинарное дерево с **n** листами всегда содержит **2n-1** узлов.

Уровень узла в бинарном дереве: уровень корня всегда равен нулю, а далее номера уровней при движении по дереву от корня увеличиваются на 1 по отношению к своему непосредственному предку.

Глубина бинарного дерева - это максимальный уровень листа дерева, что равно длине самого длинного пути от корня к листу дерева.

Полное бинарное дерево уровня **n** - это дерево, в котором каждый узел уровня **n** является листом, и каждый узел уровня меньше n имеет непустые левое и правое поддеревья

Почти полное бинарное дерево - это бинарное дерево, для которого существует неотрицательное целое **k** такое, что:

- каждый лист в дереве имеет уровень **k** или **k+1**;

- если узел дерева имеет правого потомка уровня **k+1**, тогда все его левые потомки, являющиеся листами, также имеют уровень **k+1**.

Упорядоченные бинарные деревья - это деревья, в которых для каждого узла **Х** выполняется правило: в левом поддереве - ключи, меньшие **Х**, в правом поддереве - большие или равные **Х**.

**Построение бинарного дерева. Двоичное дерево поиска**

Правило построения двоичного дерева поиска: элементы, у которых значение некоторого признака меньше, чем у корня, всегда включаются слева от некоторого поддерева, а элементы со значениями, большими, чем у корня – справа (рисунок 6).

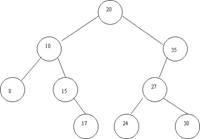


Рисунок 6- Построение двоичного дерева поиска

Этот принцип используется и при формировании двоичного дерева, и при поиске в нем элементов. Поиск места подключения очередного элемента всегда начинается с корня.

Пример: 20, 10, 35, 15, 17, 27, 24, 8, 30.

**Поиск элемента в бинарном дереве.**

При поиске элемента с некоторым значением признака происходит спуск по дереву, начиная от корня, причем выбор ветви следующего шага - направо или налево согласно значению искомого признака - происходит в каждом очередном узле на этом пути.

При поиске элемента результатом будет либо найденный узел с заданным значением признака, либо поиск закончится листом с «нулевой» ссылкой, а требуемый элемент отсутствует на проделанном по дереву пути.

Если поиск был проделан для включения очередного узла в дерево, то в результате будет найден узел с пустой ссылкой (пустыми ссылками), к которому справа или слева в соответствии со значением признака и будет присоединен новый узел.

Двоичное дерево поиска можно определить так:

− двоичное дерево состоит из узлов (вершин) — записей вида (data, left, right), где data — некоторые данные привязанные к узлу, left и right — ссылки на потомков;

− данные (data) обладают ключом (key) на котором определена операция сравнения "меньше", в конкретных реализациях это может быть пара (key, value);

− для любого узла **X** выполняются свойства дерева поиска:

key[left[X]] < key[X] ≤ key[right[X]],

т. е. ключи данных родительского узла больше ключей данных левого сына и нестрого меньше ключей данных правого.

**Основные операции в двоичном дереве поиска**

**FIND**(K) — поиск узла, в котором хранится пара (key, value) с key = K.

**INSERT**(K,V) — добавление в дерево пары (key, value) = (K, V).

**REMOVE**(K) — удаление узла, в котором хранится пара (key, value) с key = K.

**Способы обхода бинарного дерева (TRAVERSE)** (рисунок 7)

1 В симметричном порядке

**INFIX**\_TRAVERSE ( f ) — обойти всё дерево, следуя порядку (левое поддерево, вершина, правое поддерево).

2 В прямом порядке

**PREFIX**\_TRAVERSE ( f ) — обойти всё дерево, следуя порядку (вершина, левое поддерево, правое поддерево).

3 В обратном порядке

**POSTFIX**\_TRAVERSE ( f ) — обойти всё дерево, следуя порядку (левое поддерево, правое поддерево, вершина)

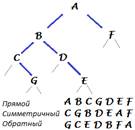


Рисунок 7- Способы обхода бинарного дерева (TRAVERSE) (рисунок 3)

**Поиск элемента в бинарном дереве**

При поиске элемента с некоторым значением признака происходит спуск по дереву, начиная от корня, причем выбор ветви следующего шага - направо или налево согласно значению искомого признака - происходит в каждом очередном узле на этом пути.

При поиске элемента результатом будет либо найденный узел с заданным значением признака, либо поиск закончится листом с «нулевой» ссылкой, а требуемый элемент отсутствует на проделанном по дереву пути.

Если поиск был проделан для включения очередного узла в дерево, то в результате будет найден узел с пустой ссылкой (пустыми ссылками), к которому справа или слева в соответствии со значением признака и будет присоединен новый узел.