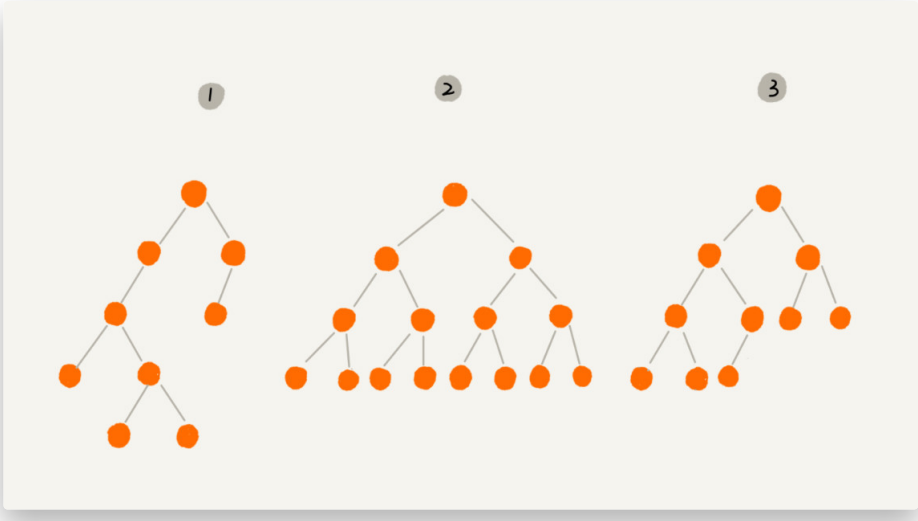
二叉树 （BINARY TREE)

介绍

二叉树，顾名思义，每个结点最多有两个 **叉**, 也就是两个子节点，分别是 **左子节点** 和 **右子节点。**

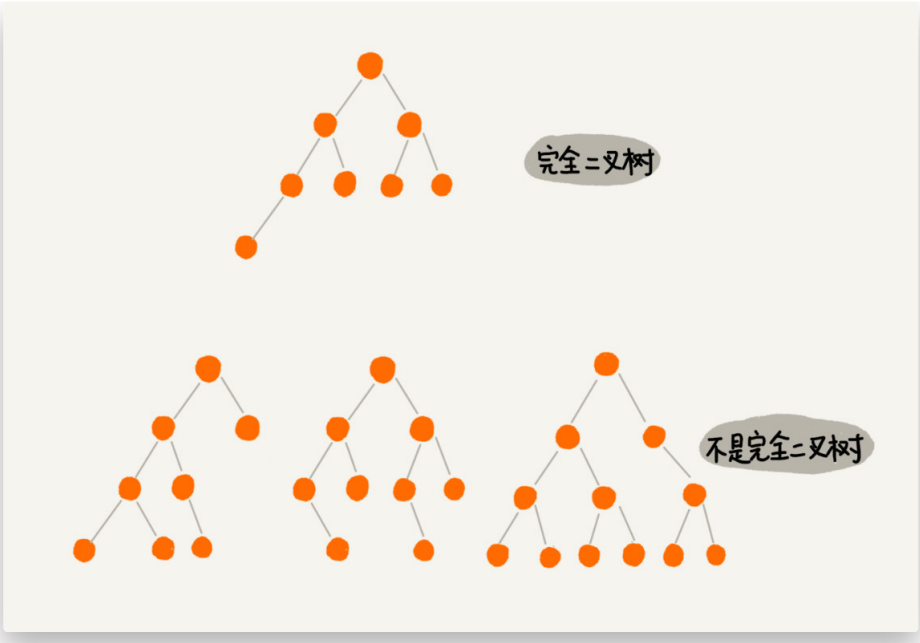
不过，二叉树，并不要求每个节点都有两个子节点，有的节点只有左子节点，有的节点只有右子节点。



特殊二叉树

- `满二叉树`：叶子节点全都在最底层，除了叶子节点之外，每个节点都有左右两个子节点，这种二叉树就叫做满二叉树. (如上图 编号 2)

- `完全二叉树`：叶子节点都在最低下两层，最后一层的叶子节点都靠左排列，并且除了最后一层，其他层的节点个数都要达到最大.(如上图 编号 3)



完全二叉树

H3 疑问

- 为什么要特意把完全二叉树拎出来讲呢?
- 为什么偏偏把最后一层的叶子节点靠左排列的叫完全二叉树?
- 如果靠右排列就不能叫完全二叉树了吗?
- 这个定义的由来或者说目的在哪里?

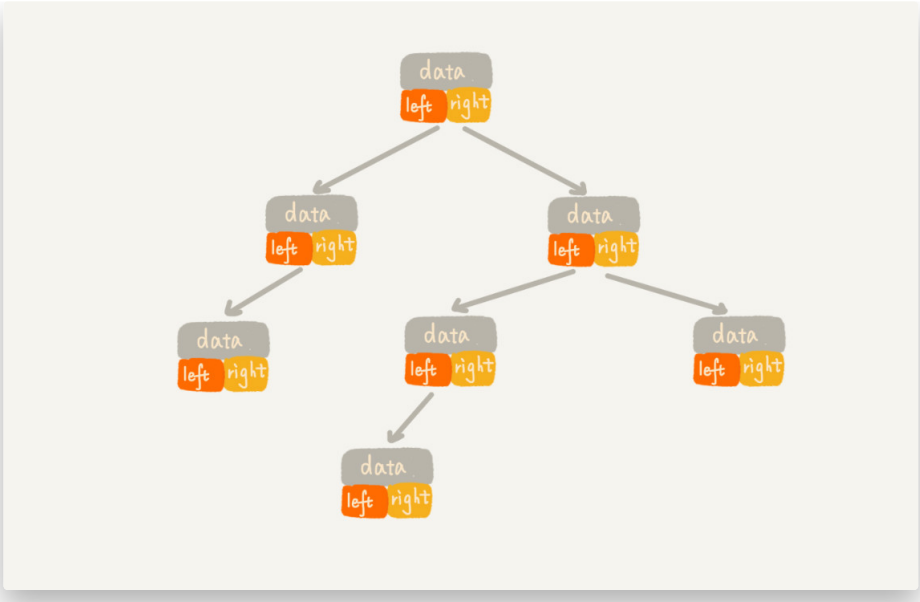
要想理解完全二叉树定义的由来，需要先了解，如何表示（或者存储）一棵二叉树?

H3 如何表示（或者存储）一棵二叉树?

- 两种方法:
  - 方法一： 基于指针或者引用的二叉链式存储法;
  - 方法二：基于数组的顺序存储法。

H4 链式存储法

- 优点：简单、直观



从图中可以很清楚地看到，每个节点有三个字段，其中一个存储数据，另外两个是指向左右子节点的指针.只要拎住根节点，就可以通过左右子节点的指针，把整棵树都串起来.

这种存储方式比较常用，大部分二叉树代码都是通过这种结构来实现的.

H4 顺序存储法

把根节点存储在下标 $i = 1$ 的位置，那左子节点存储在下标 $2 * i = 2$ 的位置，右子节点存储在 $2 * i + 1 = 3$ 的位置.