# toggle

# 容器

length 表示一个容器, 'length' 表示字符串, length <- 'length' 表示将字符串放入容器。

# 排序

1. 两两比较排序（冒泡排序）
2. 一个个选择排（选择排序）
3. 起扑克牌法（插入排序）
4. 强迫症收扑克牌法（计数排序）

缺点：

* 排序速度比快排还快；
* 负数和小数不能排序；

1. 快排
2. 归并排序
3. 堆排序
4. 桶排序

将数字分桶，例如1~10一个桶11~20一个桶，每个桶里再排序。

1. 基数排序

先排个位数，后排十位数，再拍百位数。。。。。，先出先进。

# js中的.style只能取内联样式

# 数据结构

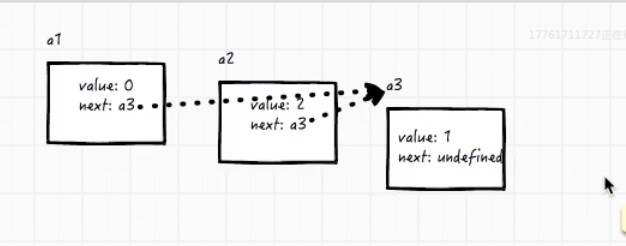
1. 哈希表（Hash Table）
   1. 计数排序中的桶（复杂度 O(n+max)，比快排还快
   2. [桶排序](http://bubkoo.com/2014/01/15/sort-algorithm/bucket-sort/) 与计数排序的区别
   3. [基数排序](http://bubkoo.com/2014/01/15/sort-algorithm/radix-sort/) 与计数排序的区别
2. 队列（Queue）
3. 先进先出
4. 可以用数组实现
5. 举例：排队

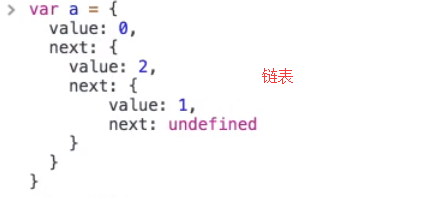
数组.push(“徐涛”)进桶； 数组.shift(“徐涛”)出桶（先进来的先出桶）；

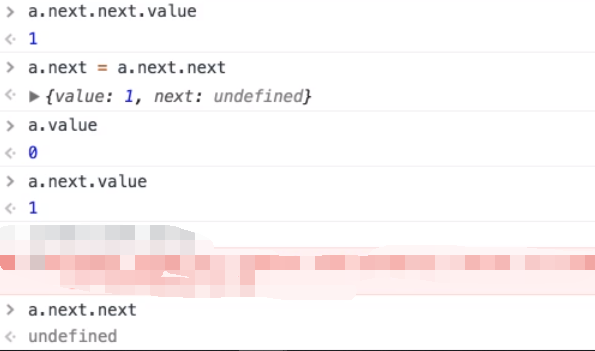
1. 栈（Stack）
2. 先进后出
3. 可以用数组实现
4. 举例：盗梦空间

数组.push(“徐涛”)进栈； 数组.pop(“徐涛”)出栈（从最后开始出栈）；

1. 链表（Linked List）
2. 数组无法**直接**删除中间的一项，链表可以
3. 用哈希（JS里面用对象表示哈希）实现链表
4. head、node 概念







1. 树（tree）
2. 举例：层级结构、DOM
3. 概念：层数、深度、节点个数
4. [二叉树](https://zh.wikipedia.org/wiki/%E4%BA%8C%E5%8F%89%E6%A0%91)
5. 满二叉树
6. 完全二叉树
7. 完全二叉树和满二叉树可以用数组实现
8. 其他树可以用哈希（对象）实现
9. 操作：增删改查
10. 堆排序用到了 tree（和快排差不多）
11. 其他：[B树](https://zh.wikipedia.org/wiki/B%E6%A0%91)、[红黑树](https://zh.wikipedia.org/wiki/%E7%BA%A2%E9%BB%91%E6%A0%91)、[AVL树](https://zh.wikipedia.org/wiki/AVL%E6%A0%91)