CH6 Queue

Queue特性：先进先出，与堆叠相反。

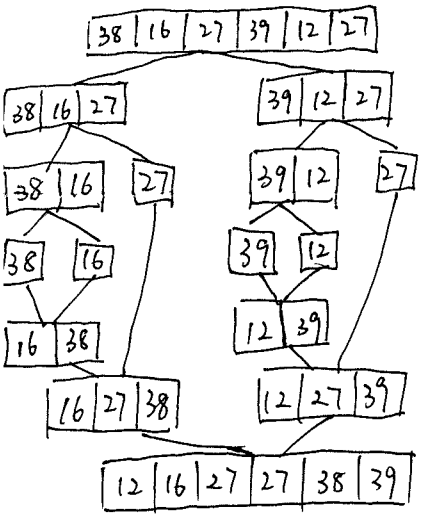
- isEmpty()
- enqueue()
- dequeue()：删除掉一个
- getFront()
- dequeue()：拿一个并删掉。

CH7.

比较演算法效率可看时间效率和空间效率两方面。

简单的：
- 气泡排序：慢。
- 选择排序：比气泡快，还是慢。
- 插入排序：选择的进阶版。
- 合并排序：先分组，后排序。

Stable：气泡、插入、合并、基数。
unstable：选择、快速、heap（堆排序）



worst case: $O(n*\log_2 n)$
average case: $O(n*\log_2 n)$

快速排序：先排序，后分组。
看第1个，大的往前丢，小的不动。排完后以这个数为分界分组继续排序。

基数排序：用Queue。根据某一位数的大小分别放到10个Queue，再依次拿出。即拿完了就排完了。是最快的排序。不需要相互比较。

CH8. 树.

有阶级结构的可以用树状结构.

A是B的父节点 (parent)

B是A的子节点 (child)

叶节点：没有子节点的节点 (Leaf)

兄弟节点：同一个父节点下的节点 (Siblings)

祖先节点：所有可以管到它的节点. (Ancestor).

子孙节点：所有可以由它到达的节点 (Descendant)

二元树：一个节点往下最多只有2个节点.

树不能有循环.

完全树：所有分叉都被填满.

平衡树：对于每一个点的左右树高差距不超过1.

完整树：树高-1的部分是完全树.

最后一层的数据都偏在左边.

遍历二元树：
- 前序：在递回查找下一个节点前先印出，先找左边节点.
- 中序：先找左边节点，再印自己，再找右边.
- 后序：先找左边，再找右边，最后印自己.

二元搜寻树：以二元树为基石而具备排序、搜寻的功能的资料结构.