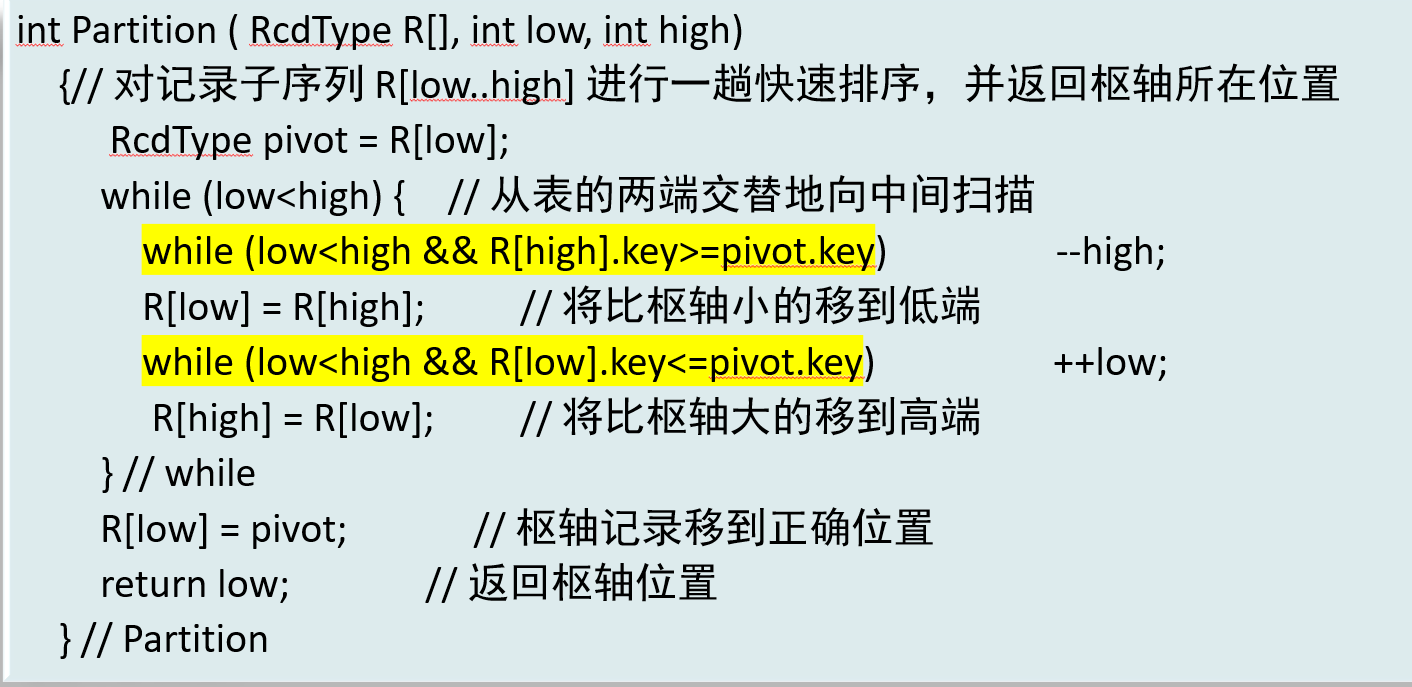


快速排序:





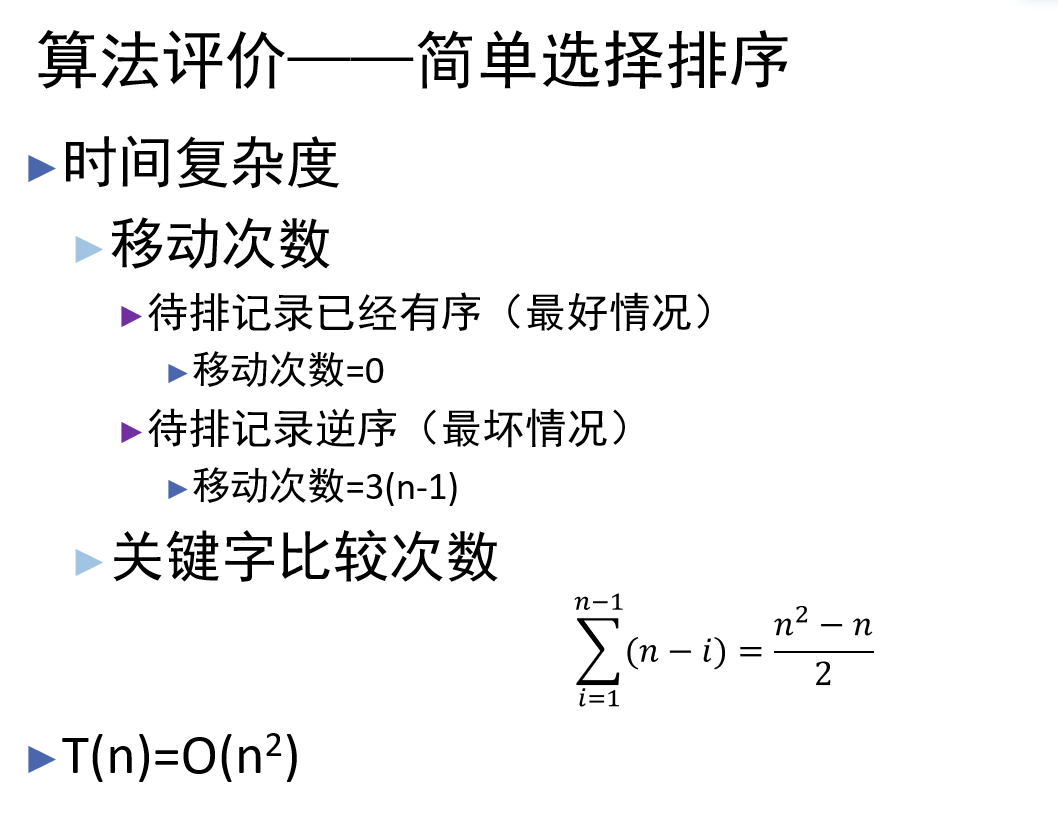
选择排序:不是稳定的排序算法

基本思想

每次从待排序的记录中选出关键字最小的记录

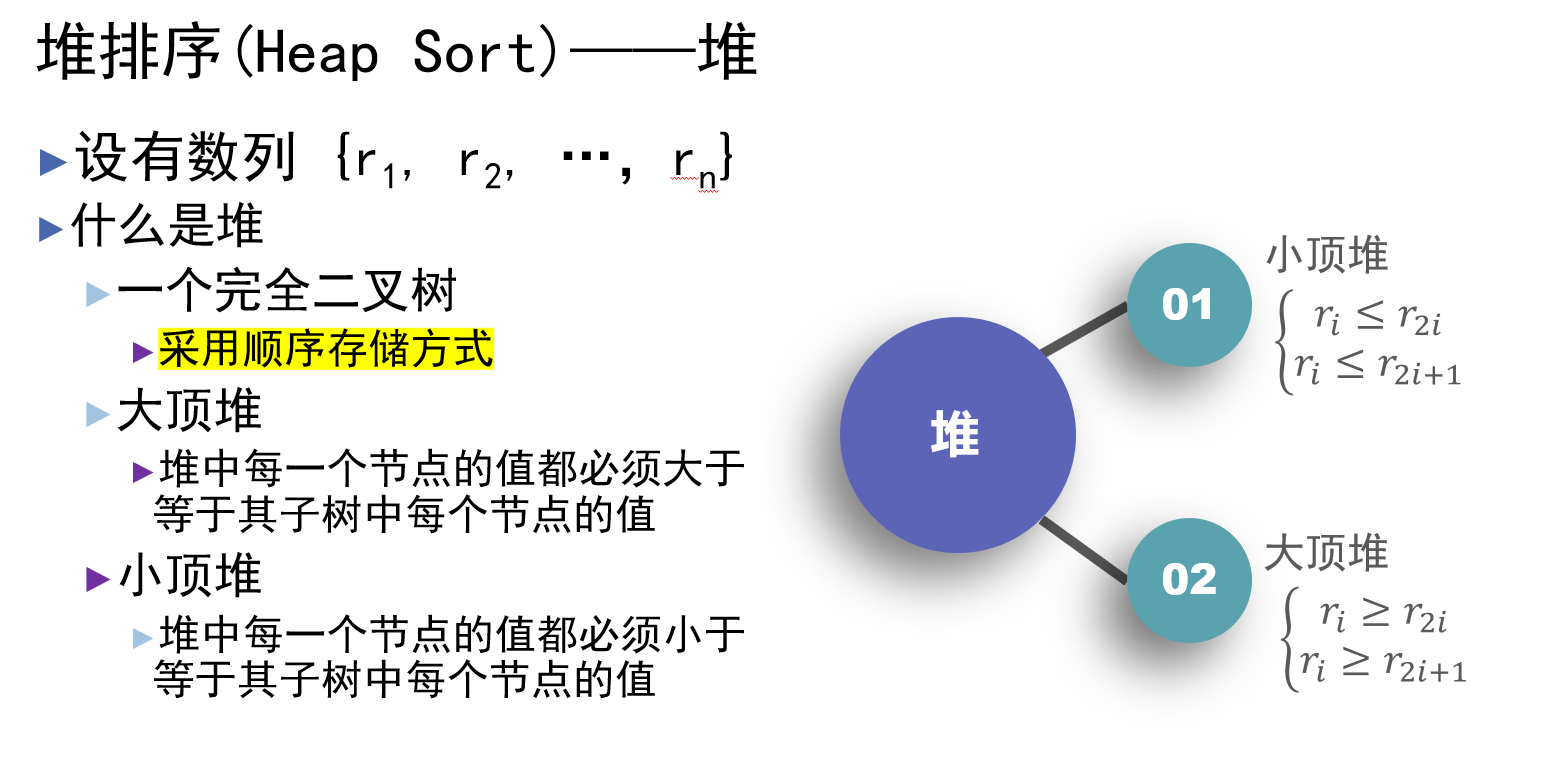
顺序放在已排好序的子表的最后

直到全部记录排序完毕



堆排序

1. 堆的概念:



1. 插入排序
2. 思想:将一个结点插入到已经有序的表里面
3. 直接插入排序:从后往前，将结点一个一个插入。

·技巧:在0位置设置一个哨兵，在从后往前寻找待插结点的位置时，就不用每次都比较判断是否越位

·空间复杂度:原地算法

时间复杂度:

移动和比较都要考虑哨兵那一次

·稳定排序算法

1. 折半插入排序: 减少比较次数，不改变移动次数
2. 二路插入排序:引入一个静态循环数组，以第一个元素为枢轴p，小于p的在p左边插入排序，大于p的在p右边插入排序，最后在一起移动，从而减小了移动次数。
3. 希尔排序: 跳跃式插入排序，减少移动和比较次数，使小的数值基本在前面，大的数值基本在后面，不大不小的基本在中间。

·希尔排序不是稳定的算法

·选取合适的跳跃间隔，可以使时间复杂度优化至

1. 交换排序:两两比较相邻的记录，反序则交换，直到没有反序为止
2. 冒泡排序:从后往前冒泡，小者上浮。

·一个优化: 增加一个flag，每次冒泡初始化为False,若冒泡过程中有交换则置为true, 若一次冒泡没有任何交换，说明已经有序，跳出整个大循环。

·复杂度分析:



1. 快速排序:借用一个枢轴p，比p小的放在p左边，比p大的放在p右边，这样就找到了枢轴的定位。然后再对枢轴的左右两个子表进行快排。

·技巧:partition分割函数:将待排数组作为引用的形式传入partition函数中，paitition依据p对数组进行分割，比p小的在p左边,比p大的在p右边。

1. 堆排序:借助堆这个结构，减少比较次数