时间复杂度：T(n)表示时间频度，指一个算法中语句执行次数，T(n) = O(f(n)) 称为O(f(n))为算法的渐进复杂度。简称时间复杂度。其中f(n)表示当n趋近于无穷大时，T(n)/f(n)的极限值为不等于的一个常数。

稳定性：排序后，原先相同的两个值的相对位置有无发生变化？

稳定：冒泡，归并，直接插入，基数

不稳定：简单选择，希尔排序，快排，堆排

# 1 排序：

## 1.1冒泡排序

两个变量i，j。其中i控制趟数，j用来遍历每一趟。

初始值：i=0;i<n-1

j=n-1;j>i

根据是要从大到小排序还是从小到大排序，改变在j的循环里的内容

如果需要找到中途就停下来，设置flag，没有交换直接停止。

## 1.2快速排序

需要用到递归

将第一个位置的值保存下来，temp = A[0];

两个变量i,j 初始值：i=left j=right

先判断j，如果A[j]的值大于temp，则j—

循环停止时：A[i] = A[j] 关键 此时A[j]已经小于temp就应该把它换到前面去

判断循环时需要加上i < j这个条件

最后结束时，将A[i] = temp

返回i，递归排序left到i-1，以及i+1到 right

稳定性：稳定

空间复杂度，需要用到递归，递归需要用到递归工作栈，最好的情况，每次选的最开始的数字都能够把数组平均分为两半，此时工作栈的深度为log(n+1) 最差的情况为每次选的数字都能把数组分为0和n-1，此时工作栈的深度为O(n)。平均情况栈的深度为O(log2)

时间复杂度：最差O(n2) 最好O(nlogn) 平均:O(nlogn)

## 1.3 归并排序

可以用于链表

先拆成两半，使用快慢指针，得到两个链表之后，再对两个链表进行拆分，这是一个递归的过程，拆到每个链表都只剩一个元素时，进行合并，合并时，如果是链表，不需要使用额外的空间来进行复制，新创建一个头节点，再进行比较合并，最后加上还不为空的链表。可以在函数中加上返回值。

用于数组时，拆分变得更容易，只用计算拆分后的位置即可，但是合并需要加上额外的辅助空间，先将A表中的内容复制到B表当中，再改变A表中的内容，比较是比较的B表。

时间复杂度：O(nlogn) 每一趟归并 需要O(n) 总共有O(logn)趟归并

空间复杂度O(n)

稳定性：稳定

## 1.4 堆排序