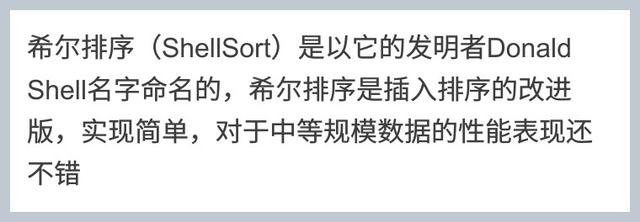
# 希尔排序

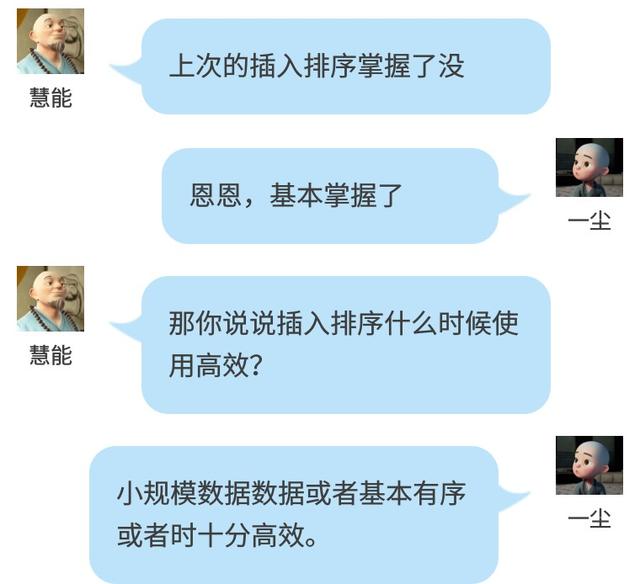
## 图解算法---希尔排序



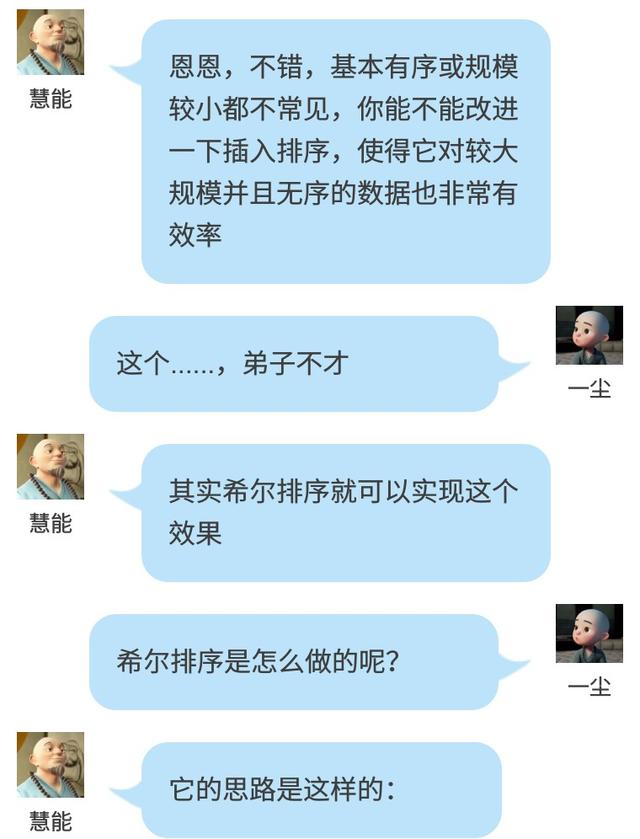


前情回顾：直接插入排序（对插入排序不熟悉的建议先阅读此文）

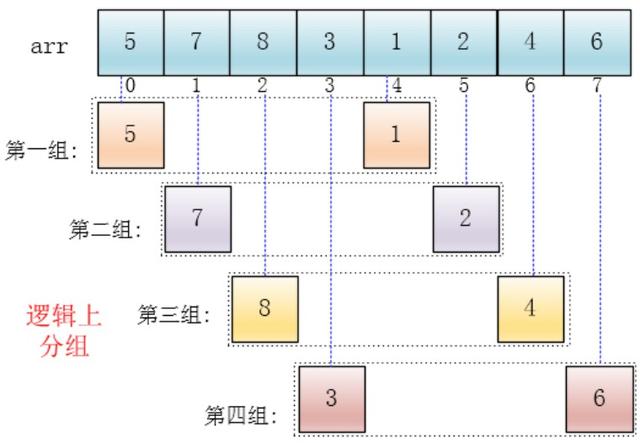
一天，一尘拿着扑克自己在那玩，刚被师傅看见了

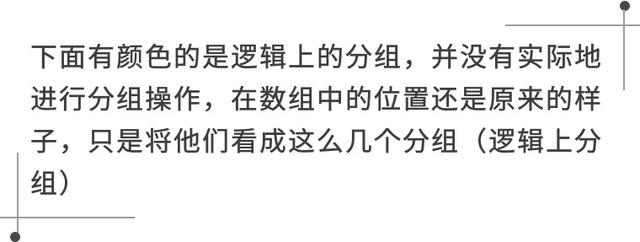




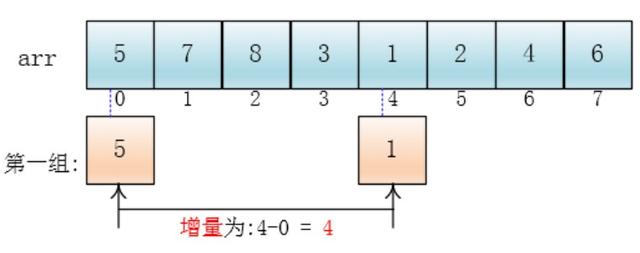


首先它把较大的数据集合分割成若干个小组（逻辑上分组），然后对每一个小组分别进行插入排序，此时，插入排序所作用的数据量比较小（每一个小组），插入的效率比较高

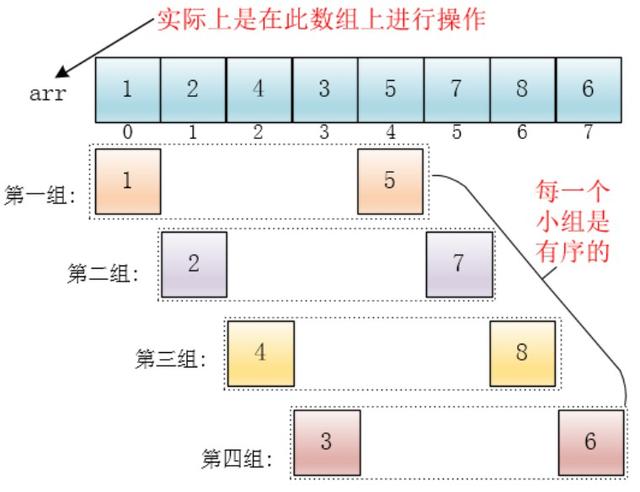




可以看出，他是按下标相隔距离为4分的组，也就是说把下标相差4的分到一组，比如这个例子中a[0]与a[4]是一组、a[1]与a[5]是一组...，这里的差值（距离）被称为增量



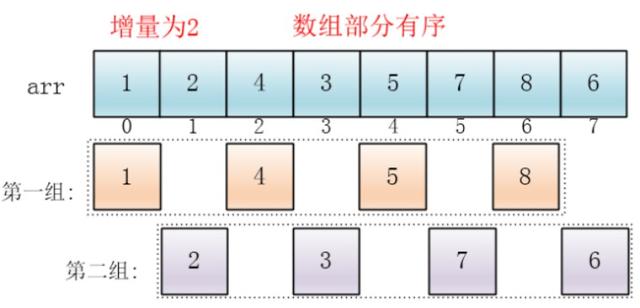
每个分组进行插入排序后，各个分组就变成了有序的了（整体不一定有序）



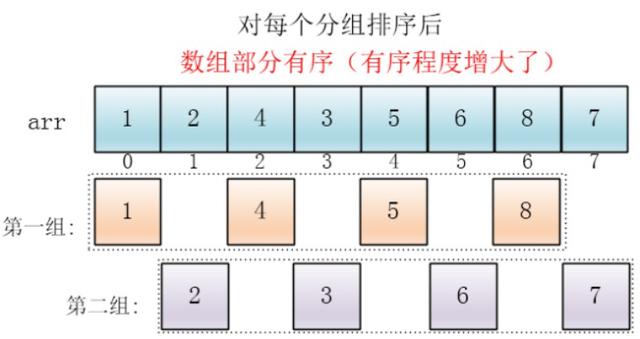
此时，整个数组变的部分有序了（有序程度可能不是很高）



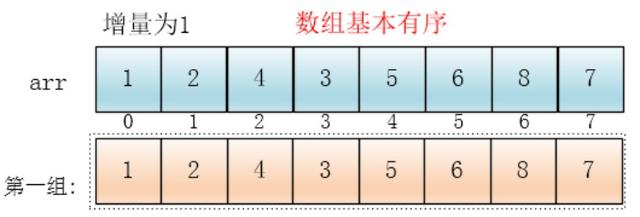
然后缩小增量为上个增量的一半:2，继续划分分组，此时，每个分组元素个数多了，但是，数组变的部分有序了，插入排序效率同样比高



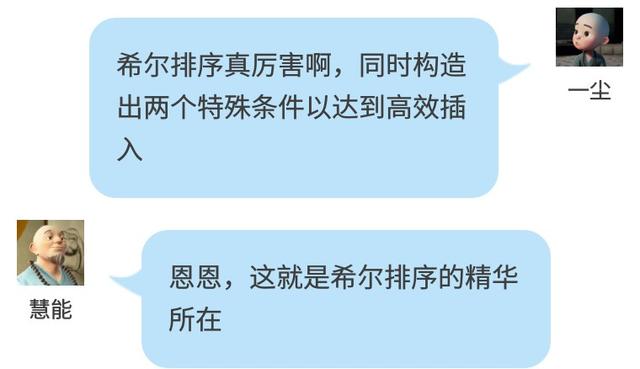
同理对每个分组进行排序（插入排序），使其每个分组各自有序



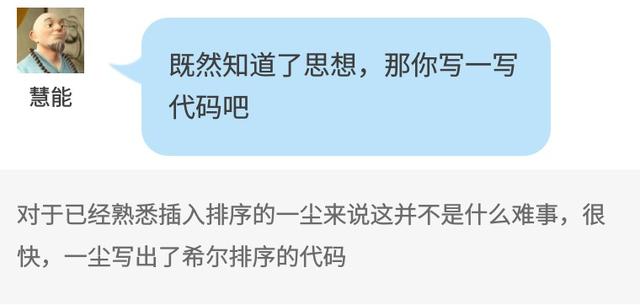
最后设置增量为上一个增量的一半：1，则整个数组被分为一组，此时，整个数组已经接近有序了，插入排序效率高

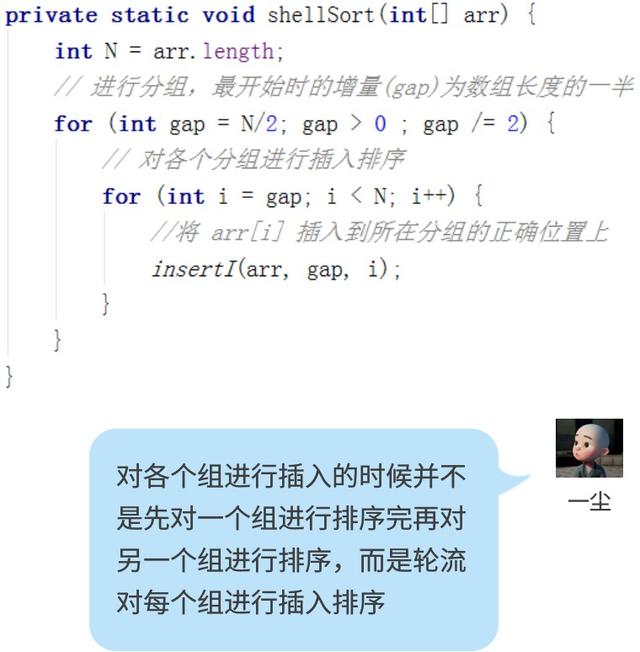


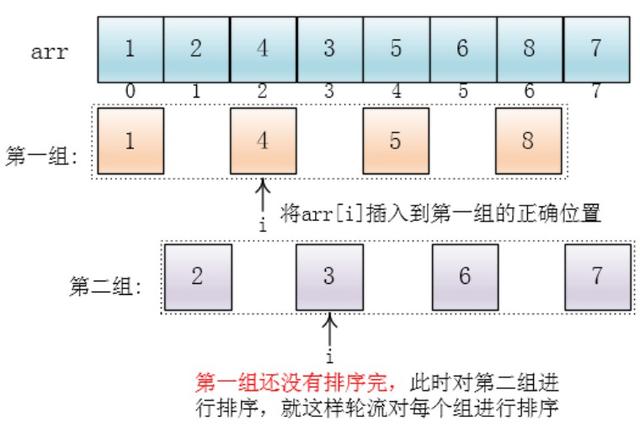
同理，对这仅有的一组数据进行排序，排序完成





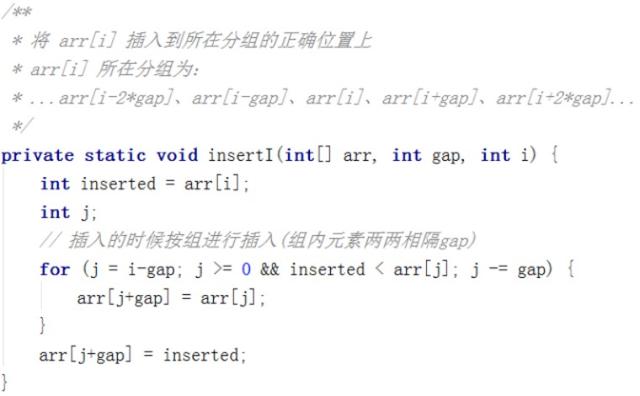


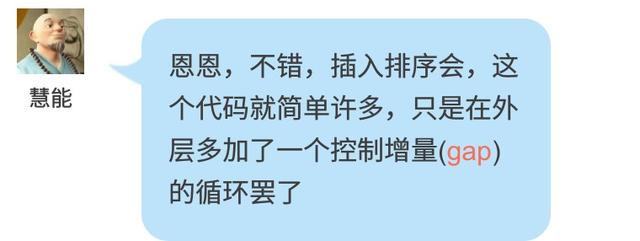




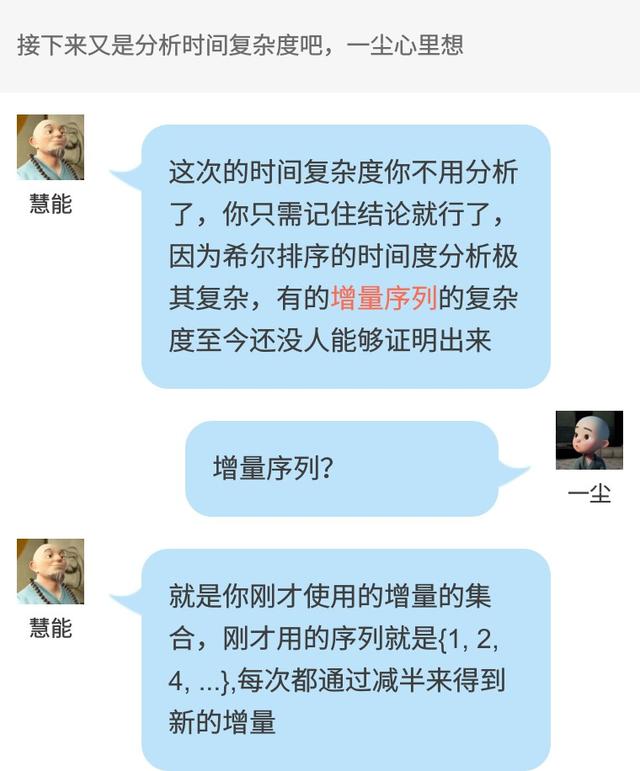
随后一尘写出了插入arr[i]到所在组正确位置的代码（insertI）









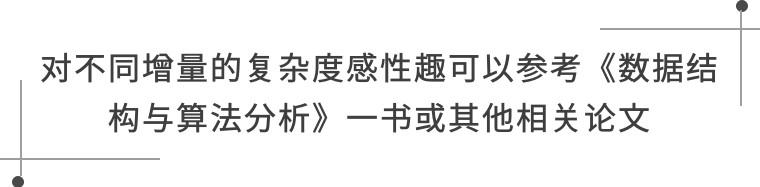


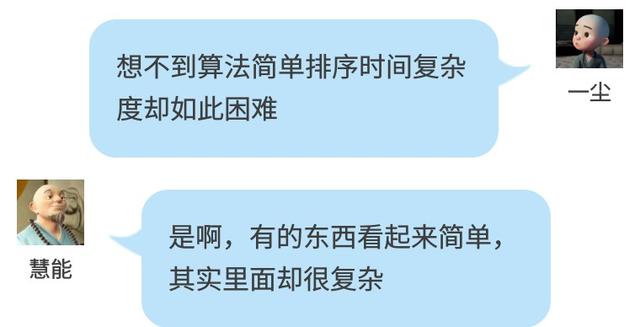
希尔排序的复杂度和增量序列是相关的

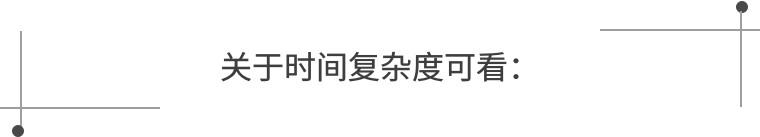
{1,2,4,8,...}这种序列并不是很好的增量序列，使用这个增量序列的时间复杂度（最坏情形）是O(n^2)

Hibbard提出了另一个增量序列{1,3,7，...,2^k-1}，这种序列的时间复杂度(最坏情形)为O(n^1.5)

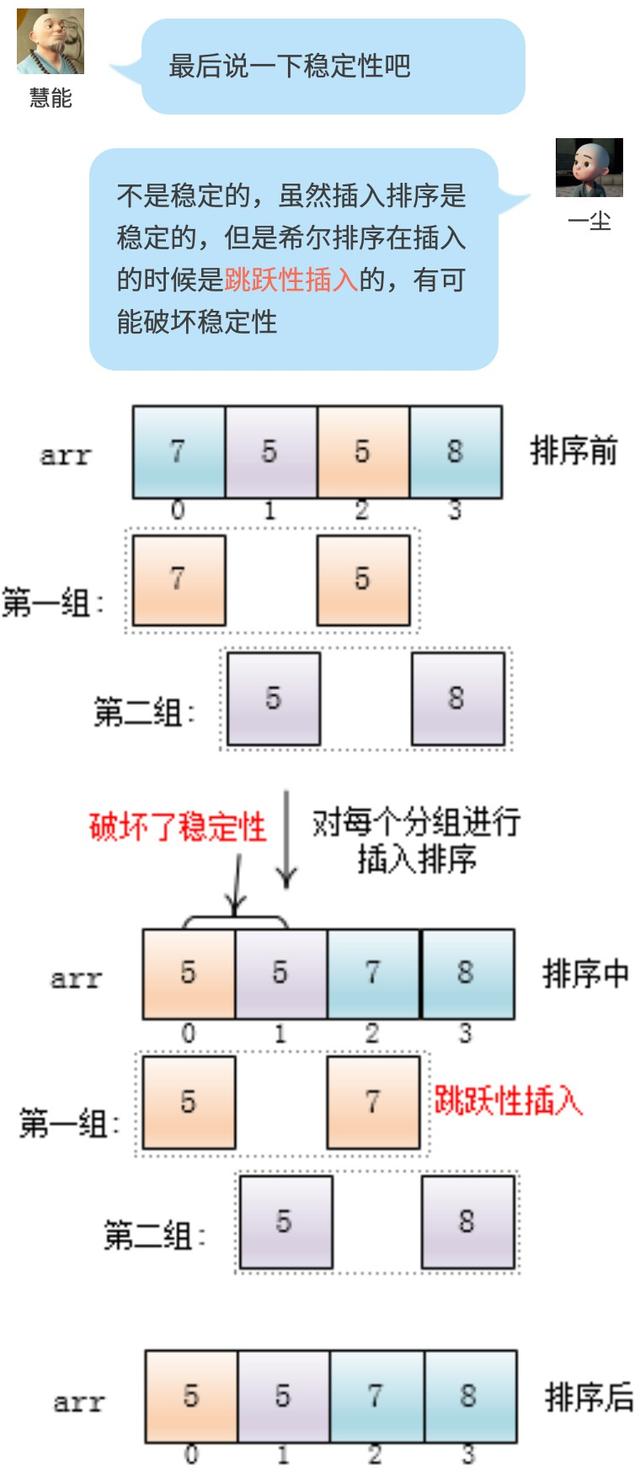
Sedgewick提出了几种增量序列，其最坏情形运行时间为O（n^1.3）,其中最好的一个序列是{1,5,19,41,109,...}

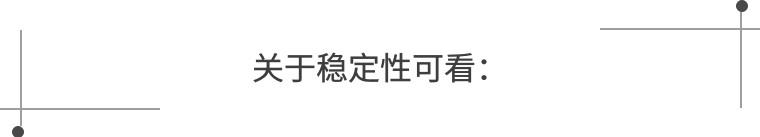












说完，一尘继续玩起了扑克。