字典序算法

字典序算法用来解决这样一个问题：给定其中一种排列，求基于字典序的下一种排列。

比如给定一种排列为 `abc`，则其基于字典序的下一种排列为 `acb`。

要求下一种排列既要比原排列大，又不能有第三种排列位于他俩之间。即下一种排列为大于原排列的最小排列。

以输入为 `358764` 为例，字典序算法的步骤：
1、从原排列中，从右至左，找到第一个左邻小于右邻的字符，记左邻位置为 a。
示例中 a=1，list[a] = 5。
2、重新从右至左，找到第一个比 list[a] 大的字符，记为位置为 b。
示例中 b=4，list[b] = 6。
3、交换 a 和 b 两个位置的值。
示例变为了 `368754`。
4、将 a 后面的数，由小到大排列。
示例变为了 `364578`。

算法结束，输出 `364578`。

注意：
1、第1步中，如果找不到左邻小于右邻的数，则说明给定的排列已经是全排列的最后一个排列了，则直接返回全排列的第一个排列，即所有排列中最小的排列，形成一个循环。
2、在第3步交换前，a 后面的数是按照从大到小进行排列（否则第1步中就可以找到左邻小于右邻的数了）。
3、在交换之后，a 后面的数仍然是按照从大到小排列的，尽管 b 位置的值变成了 list[a]，但是由于 b 位置是第一个比 list[a] 大的，因此交换之后 list[a] 仍然比左邻小，比右邻大。
4、既然 a 后面的数是从大到小排列的，那么第4步的排序，直接将 a 后面的数倒序即可。

算法的时间复杂度为 O(n) + O(n) + O(n) = O(n)。

`1,2,3` 的全排列的示例：


字典序法找（123）全排列过程
1 2 3
1 2 3
1 2 3
1 3 2
1 3 2
1 3 2
1 3 2
2 3 1
2 1 3
2 1 3
2 1 3
2 3 1
2 3 1
2 3 1
2 3 1
3 2 1
3 1 2
3 1 2
3 1 2
3 2 1
3 > 2 > 1 （退出）
字典序法（算法）
第一步：从右往左，找出第一个左边小于右边的数，设为list[a]（图中红色字体）。
第二步：从右往左，找出第一个大于list[a]的数，设为list[b]（图中浅蓝色字体）。
第三步：交换list[a],list[b]。
第四步：将list[a]后面的数据，由小往大排列（图中淡绿色字体）。
https://blog.csdn.net/HappyRocking