

## Full Permutation

### 全排列

问题:

求拥有 $n$ 个不同元素的数组 $A = [a_0, a_1, a_2, \dots, a_{n-1}]$ 的所有全排列。

解法:

本文介绍 Steinhaus-Johnson-Trotter 算法。

初始时对于长度为 1 的数组 $A = [a_0]$ ，其全排列只有 1 个，即初始状态 $[a_0]$ 。

在所有长度为 1 的排列的末尾增加新的元素 $a_1$ ，形成新的数组 $A = [a_0, a_1]$ ，并将末尾的元素 $a_1$ 分别与前面的所有元素进行，（加上初始状态的数组）得到的排列有 2 个：

$[a_0, a_1]$

$[a_1, a_0]$

在上面所有长度为 2 的排列的末尾增加新的元素 $a_2$ ，形成新的数组 $A = [a_0, a_1, a_2]$ ，并将末尾的 $a_2$ 分别与前面的所有元素交换，（加上初始状态的数组）得到的排列有 6 个：

$[a_0, a_1, a_2]$

$[a_0, a_2, a_1]$

$[a_2, a_1, a_0]$

$[a_1, a_0, a_2]$

$[a_2, a_0, a_1]$

$[a_1, a_2, a_0]$

重复这样的操作，即可得到长度为 $n$ 的数组 $A = [a_0, a_1, a_2, \dots, a_{n-1}]$ 的全排列。该算法的时间复杂度为 $n!$ 。

StackOverflow 上关于全排列的问题:

<http://stackoverflow.com/questions/9878846/listing-all-permutations-of-a-given-set-of-values>

Steinhaus-Johnson-Trotter 算法:

[https://en.wikipedia.org/wiki/Steinhaus%E2%80%93Johnson%E2%80%93Trotter\\_algorithm](https://en.wikipedia.org/wiki/Steinhaus%E2%80%93Johnson%E2%80%93Trotter_algorithm)