1. 题目
2. 题干

实现获取下一个排列的函数，算法需要将给定数字序列重新排列成字典序中下一个更大的排列。

如果不存在下一个更大的排列，则将数字重新排列成最小的排列（即升序排列）。

必须原地修改，只允许使用额外常数空间。

（二）示例

以下是一些例子，输入位于左侧列，其相应输出位于右侧列。

1,2,3 → 1,3,2

3,2,1 → 1,2,3

1,1,5 → 1,5,1

1. 题解
2. 思路

题目要求下一个更大的排列，这个更大的排列所组成的数是比现在的排列更大的数，且两个数应该是紧挨着的。如果不存在更大的排列，将返回最小的排列，即升序序列。什么时候没有更大的排列呢？当然是降序排列的时候，最大的数都在前面，后面没有更大的数了，因此没有更大的排列了。

先来举几个例子看看：

现有4个数字的排列如下：1， 3， 4， 5，如何求那个紧挨着的更大的排列呢？可以看到，现在4个数字组成的数为1345，要想求紧挨着的更大的数，就要从后面开始找，因为小位数的变动使得新的排列离原有的数更近。

从后向前找，最后一个数是5，前面的数是4，要想得到更大的下一个排列，将4和5调换即可，得到新的排列1354。

2.

5个数的排列如下：1， 3， 4， 5， 1.

同理，从后往前找，第一个数是1，前面的数是5，此时1和5不能调换，因为调换后的数比现在小；那么继续往前找，5的前面是4，4可以和5调换吗？当然可以，调换以后，尾部的3个数就是541，比现在的451要大，但是问题是541不是最贴近451的数，最贴近的数是514，因此需要将调换后的后两位进行升序排列，排列后为514.

3.

6个数的排列为：1， 3， 4， 6， 5， 1

同理，从后往前找，6， 5， 1都是降序排列，无法找到更大的排列，直到6前面是4，就可以进行调换了，那么，4要和后面的哪个数调换呢？当然是要和离4更近的数且大于4的5了，调换了4和5以后，将尾部的6， 4， 1三个数进行升序排列，最后4个数为5，1, 4，6。

由此，可以得出套路，从排列的尾部开始查找，如果从尾部开始向前一直是升序，那么不断向前，当某个数a不再大于或等于后面一个数b时，停止，将这个数a和后面的数中最接近a的且大于a的数字c进行调换，然后将c后面的所有数进行升序排列，可以观察到：c后面的数现在依然是降序的，那么要将降序的排列转为升序的，通过对称地交换数据即可实现。

1. 代码实现

注意：要考虑原有序列为降序的情况，返回升序序列。

Java：

class Solution {

    public void nextPermutation(int[] nums) {

        int i = nums.length- 2;

        while(i >= 0 && nums[i+1] <= nums[i]){

            i--;

        }

        int j = nums.length - 1;

        //i有可能为-1，因此需要判断

        if(i >= 0){

            while(j >= 0 && nums[j] <= nums[i]){

                j--;

            }

            //交换两个数

            swap(nums, i, j);

        }

        //当i为-1时，说明原有的序列为降序，因此掉转接口

        //当i不为-1时，也需要调换

        reverse(nums, i+1);

    }

    public void reverse(int[] nums, int start){

        int i = start;

        int j = nums.length - 1;

        while(i < j){

            swap(nums, i, j);

            i++;

            j--;

        }

    }

    public void swap(int[] nums, int i, int j){

        int temp = nums[i];

        nums[i] = nums[j];

        nums[j] = temp;

    }

}