3 五重复字符的最长子串

Label：动态规划、数组
给你一个整数数组 nums ，请你找出数组中乘积最大的连续子数组（该子数组中至少包含一个数字），并返回该子数组所对应的乘积。
输入: [2,3,-2,4]
输出: 6
解释: 子数组 [2,3] 有最大乘积 6。

- 遍历交换

```
class Solution {
    public int maxProduct(int[] nums) {
        int max = Integer.MIN_VALUE, imax = 1, imin = 1; //一个保存最大的，一个保存最小的。
        for (int i = 0; i < nums.length; i++) {
            if (nums[i] < 0) {
                 int tmp = imax; imax = imin; imin = tmp;
            } //如果数组的数是负数，那么会导致最大的变最小的，最小的变最大的。因此交换两个的值。

            imax = Math.max(imax*nums[i], nums[i]);
            imin = Math.min(imin*nums[i], nums[i]);

            max = Math.max(max, imax);    // Max还是求max
        }
        return max;
    }
}
```

- 遍历 动态规划

```
class Solution {
    public int maxProduct(int[] nums) {
        int length = nums.length;
        int[] maxF = new int[length];
        int[] minF = new int[length];
        maxF[0] = nums[0];
        minF[0] = nums[0];

        for (int i = 1; i < length; ++i) {
            maxF[i] = Math.max(maxF[i - 1] * nums[i], Math.max(nums[i], minF[i - 1] * nums[i]));
            minF[i] = Math.min(minF[i - 1] * nums[i], Math.min(nums[i], maxF[i - 1] * nums[i]));
        }
        int ans = maxF[0];
        for (int i = 1; i < length; ++i) {
            ans = Math.max(ans, maxF[i]);
        }
        return ans;
    }
}
```

- 连乘

```
class Solution {
    public int maxProduct(int[] nums) {
        // 思路：  求最大值，可以看成求被0拆分的各个子数组的最大值。
        // 当一个数组中没有0存在，则分为两种情况：
        // 1.负数为偶数个，则整个数组的各个值相乘为最大值；
        // 2.负数为奇数个，则从左边开始，乘到最后一个负数停止有一个“最大值”，从右边也有一个“最大值”，比较，得出最大值。
        int a = 1;
        int max = nums[0];

        for (int num:nums) {
            a = a*num;
            if (max < a) max = a;
            if (num == 0) a = 1;
        }
        a = 1;
        for (int i = nums.length - 1;i >= 0; i--){
            a = a*nums[i];
            if (max < a) max = a;
            if (nums[i] == 0) a = 1;
        }
        return max;
    }
}
```