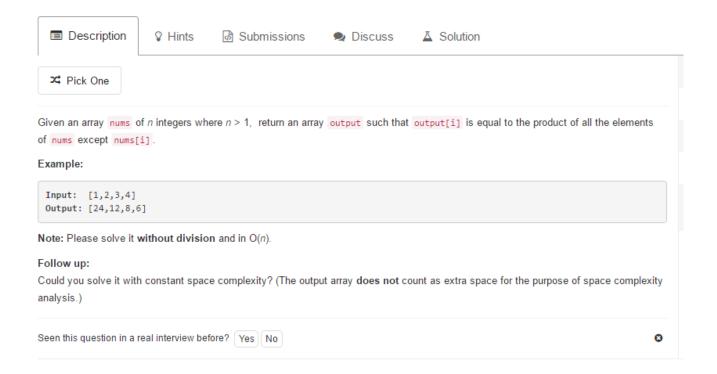
238. Product of Array Except Self



要求

题目比较好理解,但是有几个关键点这里需要明确一下:

- 1. 不能用除法。意思就是:你不能上来先把所有数乘积算出来,然后再逐个除以每个元素,这种思路是无聊、没技术含量而且不被允许的。
- 2. 时间复杂度必须控制到O(n)。 意思是:如果用O(n^2)的方法,那外层一个for循环,内层左右遍历就解决了,也是很无聊的解法。
- 3. 空间复杂度最好是常数,但是重新分配的返回数组不算在内。

思路1

我们以一个4个元素的数组为例, nums=[a1, a2, a3, a4]。 想在O(n)时间复杂度完成最终的数组输出, res=[a2*a3*a4, a1*a3*a4, a1*a2*a4, a2*a3*a4]。

比较好的解决方法是构造两个数组相乘:

- 1. [1, a1, a1*a2, a1*a2*a3]
- 2. [a2*a3*a4, a3*a4, a4, 1]

这样思路是不是清楚了很多,而且这两个数组我们是比较好构造的。

```
public class L238 {
    public int[] productExceptSelf(int[] nums) {
        int len = nums.length;
        int [] pSeq = new int[nums.length];
        int [] nSeq = new int[nums.length];
        pSeq[0] = 1;
        for(int i = 1; i < len; i ++) {</pre>
            pSeq[i] = pSeq[i - 1] * nums[i - 1];
        }
        nSeq[len - 1] = 1;
        for(int i = len - 2; i >= 0; i --) {
            nSeq[i] = nSeq[i + 1] * nums[i + 1];
        }
        for(int i = 0; i < len; i ++) {</pre>
            pSeq[i] *= nSeq[i];
        return pSeq;
    }
}
```