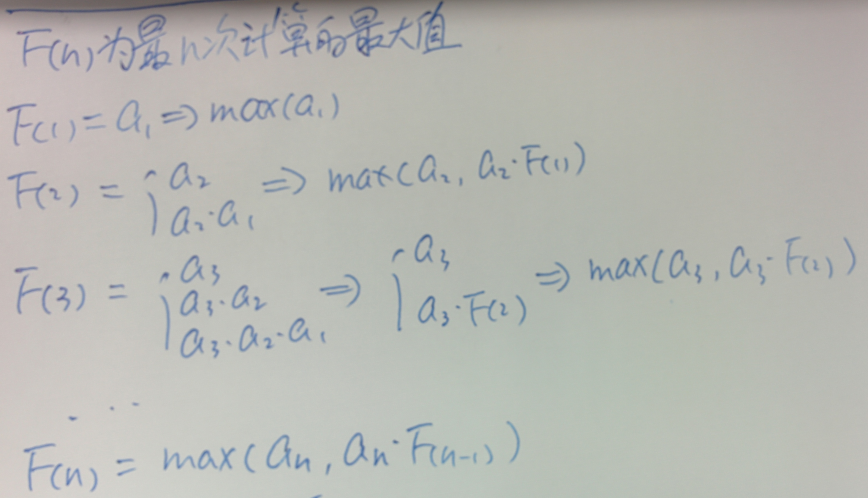
一开始比较容易想到的思路就是第一个值乘后面每一个值，然后再从第二个值开始，以此类推。但是这样的做法很明显的问题就是速度太慢，假设数组有N个数，那么每个数需要做的乘法是：N, N-1, N-2,…, 1那么总共花费的时间是其之和,所以。

上面的方法有一个重要的不足之处是每次都要重新计算每次的值，哪怕那个值之前已经计算过。其实这个问题应该用动态规划来解决，我们可以推算一下：

这样上一次计算出的最大值再下一次计算式会被用到，每次需要重新计算的量就会减少。最后得到的F(n)就是我们的转移方程。有了转移方程，就能比较方便写出代码。对于能用动态规划的问题，都可以尝试用这种不完全归纳法来推导出转移方程。

对于这个问题有点不一样的地方是不能仅仅从一个方向计算乘积，需要从左到右和从右到左同时计算，因为从两边分算可能有不一样的值。从右往左的推导大致一样。

附带的cpp，第一个函数是按照这个方程做的第一版，当然有很大的提高空间，只不过解决思路的流程是先找出解决问题的方法，然后才是优化。后一个是被人写的简洁版，很不错，呵呵。