301 Remove Invalid Parentheses

Review III

301-1 think about how to DFS.设定last\_j last\_i 来减少之后遍历的内容

301-2 开始count++ count—当count<0 才需要进行异常处理

301-3 注意中断条件 什么时候return

301-4 正面遍历完了没有中断 需要反面遍历。这种情况，char[] {‘(‘, ‘)’}数组需要反过来。

注意运用char[]数组来储存()等经常出现的char

**302 Smallest RectangleEnclosingBlackPixels**

**Review III**

302-1 1 0数组要看清到底是char[]数组还是int[]数组

302-2 bfs图的时候要防止revisit，visit过的点需要更改

302-3 Matrix的最优解就是binary search！！！

本题是利用一个方程同时处理两个维度上binary search的典范！O(nlogm)+O(mlogn)

303 304 Range Sum Query

Review III

303-1 既然关心sum那么就把数组变成sum的数组！

304-1 一维变二维 在写dp关系时候注意用三值运算符排出边界条件

305 Number Of Islands II

Review III

本题是union finding的典型例题。当在矩阵或者图中有交汇合并的情节的时候请使用union finding

UnionFind是一个单独的object 里面有一个id数组用于对数据进行分类。有find函数找dependency，有isConnected() 最重要的是有union函数整合数据。

306 AdditiveNumber

Review I

306-1 Long.parseLong() String.valueOf(integer/long/char/)

307 Range Sum Query – Mutable

Review I

307-1 前n项和：binary index tree 用binary index tree

307-2 如何求一个数以二进制表示最低的一位1的位置i&(-i)

307 308 Binary Index Tree跳过先不看 revisit later

309 Best Time to Buy and Sell Stock with Cooldown\

Review III

309-1 DP with multiple arrays at the same time。三个dp数组对每一个位置有三种描述 互相影响，求得最后结果。

310 Minimum Height Trees

ReviewIII

310-1 从叶子往中间跑，首当其冲就想到了edgeArrayList或者hashMap记录neigbor关系。通过数neighbor个数判断是不是叶子，进一步决定是否删除不断删除不断删除直到hashMap/ArrayList中只剩两个元素

311 Sparse Matrix Multiplication

311-1 注意C[i][k] += A[i][j] \* B[j][k];中的加号！

312 Burst Balloons

ReviewII

312-1 dp[i][j] = Max(dp[i][j], dp[i][x-1] + nums[i-1]\*nums[x]\*nums[j+1] + dp[x+1][j])

312-2 边界条件的处理 可以考虑扩充数组：arr[len] 🡪 arr[len+2] 收尾各加一个1

313 Super Ugly Number

Review II

313-1 dp不断的生成新数，注意remove duplicate.去重非常重要！！！

314 BT Vertical Order Traversal

Review II

314-1 本题水平方向 垂直方向都有顺序。必然想到 1 如果存储用map一定用treemap可以保证key的有序性。 2 重新定义class TreeNode 本身 让其有权重。如果用dfs左右方向的顺序可以在argument中表示。如果用bfs上下方向的权重可以用queue的round表示。所以两种方法均可以

314-2 如果用queue可以每次输入两个值：第一次为值本身 第二次为权重，这样就不需要重新定义TreeNode Class了