Leetcode 64最小路径和

class Solution {

public int minPathSum(int[][] grid) {

if(grid==null || grid.length==0 || grid[0].length==0){

return 0;

}

int rows=grid.length,columns=grid[0].length;

int[][] dp=new int[rows][columns];

dp[0][0]=grid[0][0];

for(int i=1;i<rows;i++){

dp[i][0]=dp[i-1][0]+grid[i][0];

}

for(int j=1;j<columns;j++){

dp[0][j]=dp[0][j-1]+grid[0][j];

}

for(int i=1;i<rows;i++){

for(int j=1;j<columns;j++){

dp[i][j]=Math.min(dp[i-1][j],dp[i][j-1])+grid[i][j];

}

}

return dp[rows-1][columns-1];

}

}

Leetcode 221 最大正方形

class Solution {

public int maximalSquare(char[][] matrix) {

int maxSide=0;

if(matrix == null || matrix.length ==0 || matrix[0].length==0){

return maxSide;

}

int rows=matrix.length,columns=matrix[0].length;

int[][] dp=new int[rows][columns];

for(int i=0;i<rows;i++){

for(int j=0;j<columns;j++){

if(matrix[i][j]=='1'){

if(i==0 || j==0){

dp[i][j]=1;

}else{

dp[i][j]=Math.min(Math.min(dp[i-1][j],dp[i][j-1]),dp[i-1][j-1])+1;

}

maxSide=Math.max(maxSide,dp[i][j]);

}

}

}

int maxSquare=maxSide \* maxSide;

return maxSquare;

}

}

总结：我对于动态规划理解的一点都不好，所以，暂时只大概看懂了上面两道题目的代码。我再努力看。