problem statement 1: Check whether the given Matrix is balanced or not

import java.util.\*;

public class checkBalancedMatrix{

public static void main(String[]args){

int [][] mat = {{1, 2, 3},{4, 5, 6},{7, 8, 9}} ;

System.out.println(checkBalance(mat));

}

static String checkBalance(int[][] mat){

boolean isBalanced = true;

int n = mat.length;

int m = mat[0].length;

int i,j;

for(i=0;i<n && isBalanced;i++)

{

for(j=0;j<m && isBalanced;j++)

{

//corner elements

if((i==0 || i==n-1) && (j==0 || j==m-1))

{

if(mat[i][j]>=2)

{

isBalanced = false;

}

}

else if (i==0 || i==n-1 || j==0 || j==m-1)

{

if(mat[i][j]>=3)

{

isBalanced = false;

}

}

else

{

if(mat[i][j]>4)

isBalanced = false;

}

}

}

if(isBalanced)

return "Balanced";

else

return "Not Balanced";

}

}

##########################################################################

problem statement 2 - Maximum sum of any submatrix of a Matrix which

is sorted row-wise and column-wise

import java.util.\*;

public class maxSumSubMatrix{

public static void main(String[]args){

int [][] matrix = {{-6,-4,-1},{-3,2,4},{2,5,8}};

System.out.println("Max sum of submatrix : "+maxSubSum(matrix));

}

static int maxSubSum(int[][] matrix){

int r = matrix.length;

int c = matrix[0].length;

int i,j;

//matrix print

for(i=0;i<r;i++)

{

for(j=0;j<c;j++)

{

System.out.print(matrix[i][j]+" ");

}

System.out.println();

}

//dp

int[][] dp = new int[r][c];

dp[r-1][c-1] = matrix[r-1][c-1];

//max sum prin final maximum sum

int maxSum =dp[r-1][c-1];

//submatrix for end row

for(i=c-2;i>=0;i--)

{

dp[r-1][i]= matrix[r-1][i] + dp[r-1][i+1];

maxSum = Math.max(maxSum, dp[r-1][i]);

}

//submatrix for end col

for(i=r-2;i>=0;i--)

{

dp[i][c-1]= matrix[i][c-1] + dp[i+1][c-1];

maxSum = Math.max(maxSum, dp[i][c-1]);

}

//dp matrix

for(i=r-2;i>=0;i--)

{

for(j=c-2;j>=0;j--)

{

dp[i][j] = matrix[i][j] +dp[i][j+1]+dp[i+1][j] -dp[i+1][j+1];

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

}

}

//dp matrix print

// for(i=0;i<r;i++)

// {

// for(j=0;j<c;j++)

// {

// System.out.print(dp[i][j]+" ");

// }

// System.out.println();

// }

//final value which is ans

return maxSum;

}

}