//2D Max Sum

//DP, Inclusion Exclusion

//Complexity : O(n^4)

int main()

{

int row\_column, A[100][100]; //A square matrix

scanf("%d", &row\_column);

for(int i = 0; i < row\_column; i++) //input of the matrix/2D array

for(int j = 0; j < row\_column; j++) {

scanf("%d", &A[i][j]);

if(i > 0)

A[i][j] += A[i-1][j]; //take from right

if(j > 0)

A[i][j] += A[i][j-1]; //take from left

if(i > 0 && j > 0)

A[i][j] -= A[i-1][j-1]; //inclusion exclusion

}

int maxSubRect = -1e7;

for(int i = 0; i < row\_column; i++) //i & j are the starting coordinate of sub-rectangle

for(int j = 0; j < row\_column; j++)

for(int k = i; k < row\_column; k++) //k & l are the finishing coordinate of sub-rectangle

for(int l = j; l < row\_column; l++) {

int subRect = A[k][l];

if(i > 0)

subRect -= A[i-1][l];

if(j > 0)

subRect -= A[k][j-1];

if(i > 0 && j > 0)

subRect += A[i-1][j-1]; //due to inclusion exclusion

maxSubRect = max(subRect, maxSubRect);

}

printf("2D Max Sum : %d\n", maxSubRect);

return 0;

}