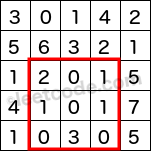
\*\* cumulative sum in 2D array \*\*\* (index starting from 1-N )

\*\*\*

Given a 2D matrix *matrix*, find the sum of the elements inside the rectangle defined by its upper left corner (*row*1, *col*1) and lower right corner (*row*2, *col*2).



\*\*\*

int mat[10][10] ;

int cum[10][10] ;

int n =5 ;

void preCal()

{

memset(cum,0,sizeof(0) ) ;

for(int r=1 ; r<=n ; r++)

{

for(int c=1 ; c<=n ; c++)

{

cum[r][c] = cum[r][c-1] + cum[r-1][c] + mat[r][c] - cum[r-1][c-1] ;

}

}

}

int solve(int row1 , int col1 , int row2 , int col2 )

{

int sum = 0 ;

sum = cum[row2][col2] - cum[row1-1][col2] - cum[row2][col1-1] +cum[row1-1][col1-1] ;

return sum ;

}

**Complexity analysis**

* Time complexity : O(mn)O(mn)O(mn) time per query. Assume that mmm and nnn represents the number of rows and columns respectively, each *sumRegion* query can go through at most m×nm \times nm×n elements.
* Space complexity : O(1)O(1)O(1). Note that data is a *reference* to matrix and is not a copy of it.

Source link :: https://leetcode.com/articles/range-sum-query-2d-immutable/#

#include<bits/stdc++.h>  
  
using namespace std ;  
#define ll long long  
  
  
#define Fix( x ) setprecision( x ) <<fixed  
#define SQ(x) ((x)\*(x))  
  
  
int mat[102][102] ;  
int cum[102][102] ;  
int n,m ;  
  
void preCal()  
{  
 memset(cum,0,sizeof(cum) ) ;  
 for(int r=1 ; r<=n ; r++)  
 {  
 for(int c=1 ; c<=m ; c++)  
 {  
 cum[r][c] = cum[r][c-1] + cum[r-1][c] + mat[r][c] - cum[r-1][c-1] ;  
 }  
 }  
}  
  
  
int solve(int row1 , int col1 , int row2 , int col2 )  
{  
 int sum = 0 ;  
  
  
 sum = cum[row2][col2] - cum[row1-1][col2] - cum[row2][col1-1] +cum[row1-1][col1-1] ;  
  
 return sum ;  
}  
  
int calArea(int x1,int y1,int x2,int y2)  
{  
 //do it  
 int a = abs(x2-x1)+1 ;  
 int b = abs(y2-y1)+1 ;  
  
 return (a\*b) ;  
  
}  
  
void printBoard(int n,int m)  
{  
 for(int i=1 ; i<=n ; i++)  
 {  
 for(int j=1 ; j<=m ;j++)  
 {  
 cout<<mat[i][j] ;  
 }  
 cout<<endl ;  
 }  
}  
  
  
  
int main(){  
 //freopen("input.txt","r",stdin);  
  
 while(scanf("%d",&n) && n!=0 ){  
 scanf("%d",&m) ;  
 char ch ;  
 for(int i=1 ; i<=n ; i++){  
 for(int j=1 ; j<=m ; j++){  
 scanf(" %c",&ch) ;  
 if(ch == '1') mat[i][j] = 1;  
 else  
 mat[i][j] = 0 ;  
 }  
  
 }  
  
 preCal() ;  
 ll cnt =0 ;  
 for(int i=1 ; i<=n ; i++){  
 for(int j=1 ; j<=m ; j++){  
 for(int x=i; x<=n ; x++){  
 for(int y=j ; y<=m ; y++){  
  
 int area = calArea(i,j,x,y) ;  
  
 int res = solve(i,j,x,y) ;  
 //cout<<i<<j <<" "<<x<<y <<" area-> "<<area <<" res " <<res <<endl;  
 if(res == area) {cnt++ ; }  
 }  
 }  
 }  
 }  
  
 printf("%lld\n",cnt) ;  
 }  
  
  
  
  
 return 0 ;  
}