GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

- Home
- Algorithms
- DS
- GATE
- Interview Corner
- Q&A
- C
- C++
- Java
- Books
- Contribute
- Ask a Q
- About

Array
Bit Magic
C/C++
Articles
GFacts
Linked List
MCQ
Misc
Output
String
Tree
Graph

# Count all possible paths from top left to bottom right of a mXn matrix

The problem is to count all the possible paths from top left to bottom right of a mXn matrix with the constraints that ***from each cell you can either move only to right or down***

We have discussed a solution to print all possible paths, counting all paths is easier. Let NumberOfPaths(m, n) be the count of paths to reach row number m and column number n in the matrix, NumberOfPaths(m, n) can be recursively written as following.

```
#include <iostream>
using namespace std;

// Returns count of possible paths to reach cell at row number m and column
// number n from the topmost leftmost cell (cell at 1, 1)
int  numberOfPaths(int m, int n)
{
   // If either given row number is first or given column number is first
   if (m == 1 || n == 1)
        return 1;

   // If diagonal movements are allowed then the last addition
   // is required.
   return  numberOfPaths(m-1, n) + numberOfPaths(m, n-1);
           // + numberOfPaths(m-1,n-1);
}

int main()
{
    cout << numberOfPaths(3, 3);
    return 0;
}
```

Output:

```
6
```

The time complexity of above recursive solution is exponential. There are many overlapping subproblems. We can draw a recursion tree for numberOfPaths(3, 3) and see many overlapping subproblems. The recursion tree would be similar to Recursion tree for Longest Common Subsequence problem.

So this problem has both properties (see this and this) of a dynamic programming problem. Like other typical Dynamic Programming(DP) problems, recomputations of same subproblems can be avoided by constructing a temporary array count[][] in bottom up manner using the above recursive formula.

```
#include <iostream>
using namespace std;

// Returns count of possible paths to reach cell at row number m and column
// number n from the topmost leftmost cell (cell at 1, 1)
int numberOfPaths(int m, int n)
{
    // Create a 2D table to store results of subproblems
    int count[m][n];

    // Count of paths to reach any cell in first column is 1
    for (int i = 0; i < m; i++)
        count[i][0] = 1;

    // Count of paths to reach any cell in first column is 1
    for (int j = 0; j < n; j++)
        count[0][j] = 1;

    // Calculate count of paths for other cells in bottom-up manner using
    // the recursive solution
    for (int i = 1; i < m; i++)
    {
        for (int j = 1; j < n; j++)

            // By uncommenting the last part the code calculatest he total
            // possible paths if the diagonal Movements are allowed
            count[i][j] = count[i-1][j] + count[i][j-1]; //+ count[i-1][j-1];

    }
    return count[m-1][n-1];
}

// Driver program to test above functions
int main()
{
    cout << numberOfPaths(3, 3);
    return 0;
}
```

Output:

```
6
```

Time complexity of the above dynamic programming solution is O(mn).

Note the count can also be calculated using the formula (m-1 + n-1)!/(m-1)!(n-1)! as mentioned in the comments of this article.

This article is contributed by **Hariprasad NG**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

## Related Topics:

- Find Union and Intersection of two unsorted arrays
- Pythagorean Triplet in an array
- Maximum profit by buying and selling a share at most twice
- Design a data structure that supports insert, delete, search and getRandom in constant time
- Print missing elements that lie in range 0 – 99
- Iterative Merge Sort
- Group multiple occurrence of array elements ordered by first occurrence
- Given a sorted and rotated array, find if there is a pair with a given sum

Tags: Dynamic Programming

**Writing code in comment?** Please use **ideone.com** and share the link here.

**33 Comments** **GeeksforGeeks** 1 **Login**

Recommend 5 Share Sort by Newest

Join the discussion…

**Aashish Karki** • a month ago

Here is a related problem:

https://orajavasolutions.wordp...

My solution: https://ideone.com/TvW3Y6

• Reply • Share ›

**lkcfree** • a month ago

here is Dynamic programming code

```
/*
opt[i][j] = 1 if i = M-1 and j= N-1 which is destination
opt[i][j] = opt[i+1][j] + opt[i][j+1];
so base case for R to left: opt[M-1][N] =1

*/

private static int computePathsDP(int[][] a, int M, int N){

int[][] opt = new int[M+1][N+1];
opt[M-1][N] =1; //starting point for base case

//bottoms up and top down appraoch

for (int i = M-1; i >=0; i--) {
for (int j = N-1; j >=0; j--){
if (a[i][j] == 1) // '1' for move '0' blocked
opt[i][j] = opt[i+1][j] + opt[i][j+1];

}
}

return opt[0][0];

}
```

• Reply • Share ›

**Harut** • a month ago

1. non recursive solution should also check for n == 0 or m == 0 (ideally <= 0 if arguments are int)

2. for consistency with solution to print all possible paths (and with non recursive version), recursive solution should check if (m == 0 || n == 0)

⌃ | ⌄ • Reply • Share ›

**Dipankar Jana** • a month ago

Why we are checking if ( m == 1 || n == 1 ) instead of if ( m == 0 || n == 0 ) ?

⌃ | ⌄ • Reply • Share ›

**Mission Peace** • 2 months ago

https://www.youtube.com/watch?... Check this video on same topic

⌃ | ⌄ • Reply • Share ›

**anil** • 2 months ago

there are N points (x1,y1),(x2,y2),.........(Xn,Yn).any two points (Xi,Yi) and (Xj,Yj)can be connected by a striaght line.
1)find out how many non-overlapping lines having 30degree inclination can be drawn.
2)locate specifically the shortest and longest line.
write an algorithm for this question..please help me.

⌃ | ⌄ • Reply • Share ›

**anil** • 2 months ago

there are N points in a rectangular grid whose vertices are 4. the N points are (x1,y1),(x2,y2),........(Xn,Yn).count how many recangular grids can be composed of N points,which are inside the bounding rectangular grid. i want write c program ro that as soon as possible....

⌃ | ⌄ • Reply • Share ›

**shinam** • 5 months ago

Can optimize on space complexity from O(n^2) to O(n)

```
for(int i = 1 ; i < m; i++){
for(int j = 1 ; j < n; j++)
dp[1][j] = dp[0][j] + dp[1][j-1];
for(int j = 1 ; j < n; j++)
dp[0][j] = dp[1][j];
}
return dp[1][n-1];
```

⌃ | ⌄ • Reply • Share ›

**Arnab** • 6 months ago

(m+n-2)!/((m-1)!*(n-1)!)

1 ⌃ | ⌄ • Reply • Share ›

**Arnab** • 8 months ago

```
public class allPossiblePaths {

public static void main(String args[])
```

```
{

int m=3,n=3;

int ans=fact(m+n-2)/(fact(m-1)*fact(n-1));

System.out.print("No. of possible paths: "+ans);

}

public static int fact(int n)

{

int t=1;

while(n>1)
```

see more

^ | ˅ • Reply • Share ›

**Priyal Rathi** • 10 months ago

Problem can also be solved in if we keep count of number of paths to reach bottom right of matrix(mat[m-1][n-1]) from any cell in the matrix.

DP[i][j]=DP[i][j+1]+DP[i+1][j]; //DP[i][j]= count of num paths to reach bottom right of matrix from mat[i][j]
Time complexity: O(mn)
link: http://ideone.com/d7ZRiP

^ | ˅ • Reply • Share ›

**Gaurav Gupta** • 10 months ago

how to adjust this solution for values of rows and columns like 100 and 500? long long int doesn't seem to work well.

^ | ˅ • Reply • Share ›

**Karshit Jaiswal** • 10 months ago

@geeksforgeeks
Correct the comment in the code...
it should be row and not column.

1 ^ | ˅ • Reply • Share ›

**MANAS KUMAR** • 10 months ago

You can also use one dimensional array(Code in JAVA)

```
public static int numberSteps(int[][] a,int r,int c){
```

```
int r1 = a.length;

int c1 = a[0].length;

int[] f=new int[100];

if(r>r1)

return 0;

if(c>c1)

return 0;

if(f[r1*c1]!=0) return f[r1*c1];

if((r==r1)&&(c==c1))

return 1;

return f[r1*c1]=numberSteps(a,r+1,c)+numberSteps(a,r,c+1);

//return ((numberSteps(a,r+1,c))+(numberSteps(a,r,c+1)));

}
```

• Reply • Share ›

**Vinod Prabhu** • a year ago

assuming that I have a 2 rows and 3 columns matrix.

then the number of paths according to this program is 3. but if ai draw paths it is 4.

[0,0]->[0,1]->[0,2] ->[1,2]

[0,0]->[0,1]->[1,1] ->[1,2]

[0,0]->[1,0]->[1,1] ->[1,2]

[0,0]->[1,0]->[1,1] ->[0,1]->[0,2]->[1,2]

• Reply • Share ›

**Programmer** ➦ Vinod Prabhu • a year ago

you can either move only to right or down

• Reply • Share ›

**sujeet singh** • a year ago

```
#define ROW 5
#define COLUMN 5
```

```
#define COLUMN 3

using namespace std;

int get_count_paths(int* matrix,int m,int n)
{
return *(matrix+((m-1)*COLUMN)+n-1);
}

void set_count_paths(int* matrix)
{

for(int i =0 ;i < ROW;i++)
for(int j=0;j< COLUMN;j++)
{
if(i==0 ||j ==0)
*(matrix+(i*COLUMN)+j)=1;
else
*(matrix+(i*COLUMN)+j)= *(matrix+((i-1)*COLUMN)+j)+ *(matrix+(i*COLUMN)+(j-1));
}
}
```

• Reply • Share ›

**proton** • a year ago

```
// Count of paths to reach any cell in first column is 1
for (int i = 0; i < m; i++)
count[i][0] = 1;
```

We're starting from top-left with one column at a time...How then to reach any cell in first column is 1 ???

• Reply • Share ›

**Gnanodharan Madhavan** • a year ago

Simple recursion to print all the paths.

```
import java.awt.Point;

import java.util.List;

import java.util.ArrayList;

public class printpathofmxmatrix{

private static int TARGET = 100;

private void printPathABofMatrix(int arr[][],int m, int n, List<point> list){

if(m>=arr.length || n>=arr[0].length)
```

```
return;

Point point = new Point(m,n);

list.add(point);
```

see more

⌃ | ⌄ • Reply • Share ›

**Alok Kumar** • a year ago

The time complexity of O(m*n) is OK, but we can improve the space complexity as O(min(m,n)).

```
#include<stdio.h>
#include<stdlib.h>
int ans(int m,int n)
{
if(m<=0||n<=0) return 0;
if(m>n) return ans(n,m);
int dp1[m];
int loop1,loop2;
for(loop1=0;loop1<m;loop1++) dp1[loop1]="1;" for(loop1="0;loop1&lt;n-1;loop1++)" {=""
for(loop2="m-2;loop2">=0;loop2--)
{
dp1[loop2]=dp1[loop2]+dp1[loop2+1];
}
}
return dp1[0];
}
int main()
{
printf("%d\n",ans(5,8));
return 0;
}
```

⌃ | ⌄ • Reply • Share ›

**Ritesh Mahato** • a year ago

@GeeksForGeeks : In second example, the comment should be 'row' and not 'column'. Pl correct.

```
// Count of paths to reach any cell in first 'column' is 1
for (int j = 0; j < n; j++)
count[0][j] = 1;
```

3 ⌃ | ⌄ • Reply • Share ›

**Sekhar** • a year ago

```
static int printAllPaths(int[][] a, int rowCount, int colCount, int currX, int currY) {

if (currX == rowCount - 1) {
return 1;
}

if (currY == colCount - 1) {
return 1;
}

return printAllPaths(a, rowCount, colCount, currX + 1, currY)
+ printAllPaths(a, rowCount, colCount, currX, currY + 1);
}
```

^ | v • Reply • Share ›

**trojansmith1990** • a year ago

Hi,
Have written here
http://ideone.com/qrYpmc

^ | v • Reply • Share ›

**Ram** • a year ago

There are several variations of this problem which are exhaustively covered at http://n1b-algo.blogspot.com/2...

5 ^ | v • Reply • Share ›

**Subrahmanyan Sankar** ➦ Ram • a year ago

Thank you for sharing this

1 ^ | v • Reply • Share ›

**Hari** ➦ Ram • a year ago

Nice blog thanks for sharing...

^ | v • Reply • Share ›

**Rohit Mitra** • a year ago

It can be written as (m + n -2) C (n - 1)

1 ^ | v • Reply • Share ›

**LK** ➦ Rohit Mitra • a year ago

Could you please explain?

1 ^ | v • Reply • Share ›

**gourav pathak** ➦ LK • a year ago

to reach the final cell you have to take (m-1) steps to the right and (n-1)steps down. So total steps are (m-1)+(n-1)=m+n-2. Out of these (m+n-2) steps any (n-1) steps should be down. So the total number of ways is (m+n-2)C(n-1) or (m+n-2)C(m-1).

10 ⌃ | ⌄ • Reply • Share ›

**h@kumar** • a year ago

A Short and sweet soln->
(2n-2) C (n-1)

3 ⌃ | ⌄ • Reply • Share ›

**sudhakar** ➦ h@kumar • a year ago

this won't work for large matrix like 1000 x 1000

⌃ | ⌄ • Reply • Share ›

**gourav pathak** ➦ h@kumar • a year ago

Even that would take an O(mn) if m was large

1 ⌃ | ⌄ • Reply • Share ›

**Vinod** ➦ gourav pathak • a year ago

assuming that I have a 2 rows and 3 columns matrix.

then the number of paths according to this program is 3. but if ai draw paths it is 4.

[0,0]->[0,1]->[0,2] ->[1,2]

[0,0]->[0,1]->[1,1] ->[1,2]

[0,0]->[1,0]->[1,1] ->[1,2]

[0,0]->[1,0]->[1,1] ->[0,1]->[0,2]->[1,2]

1 ⌃ | ⌄ • Reply • Share ›

✉ Subscribe  Add Disqus to your site  Privacy

- Interview Experiences
- Advanced Data Structures
- Dynamic Programming
- Greedy Algorithms
- Backtracking
- Pattern Searching
- Divide & Conquer
- Mathematical Algorithms
- Recursion
- Geometric Algorithms

## Popular Posts

- All permutations of a given string
- Memory Layout of C Programs
- Understanding “extern” keyword in C
- Median of two sorted arrays
- Tree traversal without recursion and without stack!
- Structure Member Alignment, Padding and Data Packing
- Intersection point of two Linked Lists
- Lowest Common Ancestor in a BST.
- Check if a binary tree is BST or not
- Sorted Linked List to Balanced BST

Follow @GeeksforGeeks

## Recent Comments

- lt_k

  i need help for coding this function in java...

  Java Programming Language · 2 hours ago

- Piyush

  What is the purpose of else if (recStack[*i])...

  Detect Cycle in a Directed Graph · 2 hours ago

- Andy Toh

  My compile-time solution, which agrees with the...

Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 2 hours ago

- lucy

  because we first fill zero in first col and...

  Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago

- lucy

  @GeeksforGeeks i don't n know what is this long...

  Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 3 hours ago

- manish

  Because TAN is not a subsequence of RANT. ANT...

  Given two strings, find if first string is a subsequence of second · 3 hours ago

-