DP

题目描述(p1004)

设有N×N的方格图(N≤9)，我们将其中的某些方格中填入正整数，而其他的方格中则放入数字0。如下图所示（见样例）：

```
A
 0  0  0  0  0  0  0  0
 0  0 13  0  0  6  0  0
 0  0  0  0  7  0  0  0
 0  0  0 14  0  0  0  0
 0 21  0  0  0  4  0  0
 0  0 15  0  0  0  0  0
 0 14  0  0  0  0  0  0
 0  0  0  0  0  0  0  0
                      B
```

某人从图的左上角的AA点出发，可以向下行走，也可以向右走，直到到达右下角的BB点。在走过的路上，他可以取走方格中的数（取走后的方格中将变为数字00）。 此人从AA点到BB点共走两次，试找出22条这样的路径，使得取得的数之和为最大。

输入格式

输入的第一行为一个整数NN（表示N×N的方格图），接下来的每行有三个整数，前两个表示位置，第三个数为该位置上所放的数。一行单独的00表示输入结束。

输出格式

只需输出一个整数，表示22条路径上取得的最大的和。

输入输出样例

```
输入                输出 67
8
2 3 13
2 6  6
3 5  7
4 4 14
5 2 21
5 6  4
6 3 15
7 2 14
0 0  0
```

思路:

dp 设计状态-->转移方程 如dp[i][j]=dp[i-1][j]+dp[i-1][j-1] （好像是杨辉三角形）

优化：记忆化搜索

本题是把走两次当作两个人同时走，转移方程也很明白嘛，主要是特判一下走到同一点是只记一次。

有的dp可以滚动数组优化。

背包问题单独写

code

```cpp
#include <iostream>
#include<cstdio>
#include <cstring>
using namespace std;
int main(){
    int N,m,n,q;
    int dp[11][11][11][11];
    int mapp[11][11];
    cin>>N;
    memset(dp,0,sizeof(dp));
    while(cin>>m>>n>>q){
        if(m==0&&n==0&&q==0) break;
        mapp[m][n]=q;

    }
    for(int i=1;i<=N;i++){
        for(m=1;m<=N;m++){
            for(n=1;n<=N;n++){
                for(q=1;q<=N;q++){
                    if(i==n&&m==q){
                    dp[i][m][n][q]=mapp[i][m]+
                    max(max(dp[i-1][m][n-1][q],dp[i][m-1][n][q-1]),
                    max(dp[i][m-1][n-1][q],dp[i-1][m][n][q-1]));
                    }
                    else{ dp[i][m][n][q]=mapp[i][m]+mapp[n][q]+
                    max(max(dp[i-1][m][n-1][q],dp[i][m-1][n][q-1]),
                    max(dp[i][m-1][n-1][q],dp[i-1][m][n][q-1]));
                    }
                }
            }
        }
    }
    cout<<dp[N][N][N][N];
    return 0;
}
```