实验8 经典算法应用（动态规划）

计科2004 202008010404 赖业智

1. **问题分析**
2. 问题与功能描述
3. 需要处理的是两个int型数字，它们组成了一个方格矩阵的行和列。
4. 实现的功能有，输入两个正整数m，n，找出从图的左下角到图右上角路径数，即从坐标（1，1）到（m，n），蚂蚁只能向上或者向右移动。
5. 输出不同移动路线的数目。
6. 样例分析
7. 输入样例

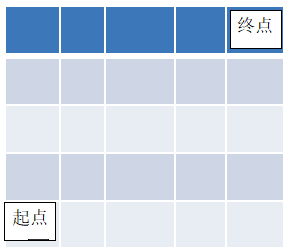
10 10

1. 输出样例

48620

1. 样例说明

我们可以利用组合数学的知识分析一下这个题目



如现在行为5，列为5，那么我们从起点到终点，无论怎么走，都必须向右走4步，向上走4步，从这8步中找更小步的走，因此可以是C(4+4,4)=70。同理，样例是C(9+9,9)=48620。

1. **数据结构分析**
2. 数据对象：本题处理m，n两个int整数。
3. 数据关系：m为行，n为列，数据规模已经确定，因此我们可以开一个二维数组表示这个m行n列的方格矩阵。
4. 物理实现

int a[20][20]={0};

1. **算法分析**

1.算法思想：我们可以直接是组合数学的知识，利用公式去推导出结果，但是我们可以采用动态规划的思想，获得更简便的结果。先将数组a[1][1]起初始化为1，我们要找出状态转移方程，从a[1][1]到a[2][2]的路径数可以分解为是a[1][1]到a[1][2]的路径数和a[1][1]到a[2][1]的路径数的和，因此通过累加可以得到每一个坐标的路径数，状态转移方程为

a[i][j] = a[i - 1][j] + a[i][j - 1];

1. 算法实现：

**static int a[20][20] = { 0 };**

**int main() {**

**int m, n;**

**cin >> m >> n;**

**for (int i = 1; i <= m; i++) {**

**for (int j = 1; j <= n; j++) {**

**if (i == 1 && j == 1)**

**a[i][j] = 1;**

**else**

**a[i][j] = a[i - 1][j] + a[i][j - 1];**

**}**

**}**

**cout << a[m][n];}**

3.性能分析：

【时间复杂度】两个for循环，时间复杂度为O(m\*n)。

【空间复杂度】开辟了一个二维数组，由于数据规模已经固定，空间复杂度为O(20\*20)。