C++方向编程题答案

第二周

day9

> **题目ID：25083 --另类加法**
>
> 链接：https://www.nowcoder.com/practice/e7e0d226f1e84ba7ab8b28efc6e1aebc?tpId=8&&tqId=11065&rp=1&ru=/activity/oj&qru=/ta/cracking-the-coding-interview/question-ranking
>
> **【题目解析】**
>
> 本题的意思是自己实现加法，不适用现成的运算符，考察大家对于运算符的灵活运用
>
> **【解题思路】：**
>
> 本题可以通过位运算实现，具体实现如下：
>
> 两个数求和，其实就是 求和后当前位的数据+两个数求和的进位
>
> 例如：
>
> 1 + 2； 00000001 + 00000010
>
> 求和后当前位的数据： 00000011 ； 求和后的进位数据： 没有进位，则 00000000
>
> 两者相加，则得到： 00000011 就是3
>
> 2 + 2； 00000010 + 00000010
>
> 求和后当前位的数据： 00000000， 1和1进位后当前为变成0了
>
> 求和后进位的数据： 00000100， 两个1求和后进位了
>
> 相加后得到： 00000100 就是4
>
> 求和后当前位的数据：简便的计算方法就是两个数进行异或 00000001 ^ 00000010 -> 00000011
>
> 求和后进位的数据：简便的计算方法就是两个数相与后左移一位 (00000010 & 00000010) << 1
>
> 所以这道题使用递归更加容易理解

```
class UnusualAdd {
public:
 int addAB(int A, int B) {
     if (A == 0) return B;
     if (B == 0) return A;
     int a = A ^ B;//求和后当前位的数据
     int b = (A & B) << 1;//求和后进位的数据
     return addAB(a, b);//递归两个数进行相加，任意为0时截止
 }
};
```

题目ID：36915-求路径总数

链接：https://www.nowcoder.com/practice/e2a22f0305eb4f2f9846e7d644dba09b?tpId=37&&tqId=21314&rp=1&ru=/activity/oj&qru=/ta/huawei/question-ranking

【题目解析】：

本题为求取路径总数的题目，一般可以通过递归求解，对于复杂的问题，可以通过动态规划求解。此题比较简单，可以通过递归解答。

【解题思路】：

```
| 1 | 2 | 3 |
-------------
| 4 | 5 | 6 |
-------------
| 7 | 8 | 9 |
-------------
1. 对于上面的n*m(3*3)的格子，有两种情况
a. 如果n或者m为1，则只有一行或者一列，从左上角走到右下角的路径数为n + m
    比如：  1 * 1格子，可以先向下走，再向右走，到达右下角；或者先向右走，
        再向下走，到达右下角，共两条，即 1 + 1 = 2，对于1 * m和 n * m的
        情况同学们自己画一下
b. 如果n,m都大于1，那么走到[n][m]格子的右下角只有两条路径，
    <1>: 从[n - 1][m]格子的右下角向下走，到达
    <2>: 从[n][m - 1]格子的右下角向右走，到达
   所以走到[n][m]格子的右下角的数量为[n-1][m] + [n][m - 1],可以通过递归实现，情况a为递归的终止条件。
```

```
#include<iostream>
using namespace std;
int pathNum(int n,int m)
{
if(n > 1 && m > 1)
//b情况，递归
return pathNum(n-1,m) + pathNum(n,m-1);
else if(((n >= 1)&&(m == 1))||((n == 1)&&(m >= 1)))
// a情况，终止条件
return n + m;
else
//格子为0时， 路径为0
return 0;
}
int main()
{
int n,m;
while(cin>>n>>m)
{
cout<<pathNum(n,m)<<endl;
}
return 0;
}
```

```
#include<iostream>
using namespace std;
int path(int n, int m)
{
    if(n > 1 && m > 1)
    {
        //b情况，递归
        return path(n-1, m) + path(n, m-1);
    }
    else if(((n >= 1) && (m == 1)) || ((n == 1) && (m >=1)))
    {
        //a情况，终止条件
        return n+m;
    }
    else
        //格子为0时，路径为0
        return 0;
}
int main()
{
    int n, m;
    while(cin >> n >> m)
    {
        cout << path(n, m)<< endl;
    }
    return 0;
}
```