C++方向编程题答案

第三周

day17

> **题目ID：36877-杨辉三角的变形**
>
> 链接：https://www.nowcoder.com/practice/8ef655edf42d4e08b44be4d777edbf43?tpId=37&&tqId=21276&rp=1&ru=/activity/oj&qru=/ta/huawei/question-ranking
>
> **【解题思路】：**
>
> 按照题目意思，可以发现第n行有2n - 1个元素，第i,j元素等于上一行第j - 2,j - 1,j三列元素之和，每一行的第一列和最后一列都为1，如果是第二列，则只是两个元素之和。
>
> **【示例代码】**

```
#include<iostream>
#include<string>
#include<vector>

using namespace std;
int main()
{
    int n, m;
    while (cin >> n)
    {
        m = 2 * n - 1;
        vector<vector<int>> dp(n, vector<int>(m, 0));
        dp[0][0] = 1;
        for (int i = 1; i < n; i++)
        {
      //第一列和最后一列都为1
            dp[i][0] = dp[i][2 * i] = 1;
            for (int j = 1; j < 2 * i; ++j)
            {
                if (j == 1)
              //如果是第二列，则只是两个元素之和
                    dp[i][j] = dp[i - 1][j - 1] + dp[i - 1][j];
                else
              //第i,j元素等于上一行第j - 2,j - 1,j三列元素之和
                    dp[i][j] = dp[i - 1][j - 2] + dp[i - 1][j - 1] + dp[i - 1][j];
            }
        }
        int k;
        for (k = 0; k < m; k++)
        {
            if (dp[n - 1][k] % 2 == 0 && dp[n - 1][k] != 0)
            {
                cout << k + 1 << endl;
```

```
                break;
            }
        }
        if (k == m)
            cout << -1 << endl;
    }
    return 0;
}
```

题目ID:36902-超长正整数相加

链接：https://www.nowcoder.com/practice/5821836e0ec140c1aa29510fd05f45fc?tpId=37&&tqId=21301&rp=1&ru=/activity/oj&qru=/ta/huawei/question-ranking

【题目解析】：

本题是模拟加法运算。

【解题思路】：

加法运算，每一位的值等于当前对应位的两数之和+进位。由于是加法，所以当前位的和最多是19（9 + 9 + 进位1），所以产生的进位最多为1。故

第一步：计算对应位的和，对应位相加 + 上一位的进位

第二步：更新当前位的值， 和 % 10 ， 把值转成字符（和 - '0'）存入字符结果中

第三步：更新进位， 和 / 10， 然后计算下一位的值

最后一步：如果计算完之后，进位为1，说明最高位产生了进位，所以需要再加一位，才是最后的结果。结果产生之后，需要逆置，得到最终结果。

【示例代码】

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

string addStrings(string num1, string num2) {
  //由低位向高位相加
    int i = num1.size() - 1;
  int j = num2.size() - 1;
  string result = "";
    //当前位的相加结果
  int carry = 0;
  while (i >= 0 || j >= 0) {
      if (i >= 0) {
          carry += num1[i] - '0';
      }
      if (j >= 0) {
          carry += num2[j] - '0';
      }
```

```
        //当前为的最大值不大于10
        result += (char)(carry % 10 + '0');
        //如果大于10,向上进一位
        carry /= 10;
        i--;
        j--;
    }
      //相加完之后，如果还有进位，则再加1
    if (carry == 1) {
        result += '1';
    }
      //整体逆置
    reverse(result.begin(), result.end());
    return result;
}

int main()
{
    string s1, s2;
    while(cin>>s1>>s2)
    {
    cout<<addStrings(s1, s2)<<endl;
    }
    return 0;
}
```