C++方向编程题答案

> **答案说明：**
>
> 大家如果对本次题目或者答案有问题，可以联系下方的出题老师答疑。
>
> **出题老师**：
>
> 选择题：鲍松山 qq ： 365690203
>
> 代码题：鲍松山 qq：365690203

第四周

day19

> **题目ID：36846-汽水瓶**
>
> 链接：https://www.nowcoder.com/practice/fe298c55694f4ed39e256170ff2c205f?tpId=37&&tqId=21245&rp=1&ru=/activity/oj&qru=/ta/huawei/question-ranking
>
> **【题目解析】：**
>
> 本题题意明确
>
> **【解题思路】：**
>
> 本题题意简单，每次空瓶的数量除以2，直到最后空瓶的数量少于两瓶，就累加到了课兑换的数量。
>
> **【实例代码】**

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

//概念法
int calcNumber_1(int n)
{
 int sum = 0;
 while(n > 1)
 {
     int res = n / 3;   //所能兑换的个数
     int left = n % 3;  //遗下的个数

     sum += res;
     n = left + res;
     if(n == 2)
     {
         sum++;
         break;
     }
 }
 return sum;
}
```

```
//取巧法
int calcNumber_2(int n)
{
 return n / 2;
}

int main()
{
 int n, res;
 while(cin >> n)
 {
     if(n == 0)
         break;
     res = calcNumber_2(n);
     cout<<res<<endl;
 }
 return 0;
}
```

题目ID:36889-查找两个字符串a,b中的最长公共子串

链接：https://www.nowcoder.com/practice/181a1a71c7574266ad07f9739f791506?tpId=37&&tqId=21288&rp=1&ru=/activity/oj&qru=/ta/huawei/question-ranking

【题目解析】：

本题题意明确

【解题思路】：

本题需要用动态规划求解，MCS[i][j]记录短字符串 s1 前 i 个字符和长字符串 s2 前 j 个字符的最长子串的长度，初始化所有值为 0。当 s1[i-1] = s2[j-1]时，MCS[i][j] = MCS[i - 1][j - 1] + 1，这里使用一个额外的值 start 来记录最长子串在短字符串 s1 中出现的起始位置，maxlen记录当前最长子串的长度，当MCS[i][j] > maxlen 时，maxlen = MCS[i][j]，则start = i - maxlen ； 档s1[i-1] != s2[j-1]时不需要任何操作，最后获取 substr(start, maxlen)即为所求。

【示例代码】

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

string getComSubstr(string &str1, string &str2)
{
    //寻求最短字符串
    if(str1.size() > str2.size())
        swap(str1, str2);

    int len1 = str1.size();
    int len2 = str2.size();
    vector<vector<int>> MSC(len1+1, vector<int>(len2+1, 0));
```

```
    int start = 0, max_size = 0;

    for(int i=1; i<=len1; ++i)
    {
        for(int j=1; j<=len2; ++j)
        {
            if(str2[j-1] == str1[i-1])
                MSC[i][j] = MSC[i-1][j-1] + 1;

            //如果有更长的公共子串，更新长度
            if(MSC[i][j] > max_size)
            {
                max_size= MSC[i][j];
                //以i结尾的最大长度为max，则子串的起始位置为i - max
                start = i - max_size;
            }
        }
    }

    return str1.substr(start, max_size);
}

int main()
{
    string str1, str2;
    while(cin >> str1 >> str2)
    {
        string substr = getComSubstr(str1, str2);
        cout<<substr<<endl;
    }
    return 0;
}
```