C++方向编程题答案

day29-22610 求正数数组的最小不可组成和

https://www.nowcoder.com/questionTerminal/296c2c18037843a7b719cf4c9c0144e4

【题目解析】： 求数组的最小不可组成和

arr = {3,2,5} arr的min为2，max为10，在区间[2,10]上，4是不能被任何一个子集相加得到的值中最小的，

所以4是arr的最小不可组成和；

【解题思路】：

这是一个动态规划的01背包问题；

根据承重和已有的重量种类阶段性计算当前承重时能够放入的重量

当数组中只有2重量的时候，背包承重从2-10都可以放入2的数值 当数组中放入2和3重量的时候，背包承重从5-10可以放入5，3-4放入3，2只能放入2 当数组中放入2，3，5重量时，背包承重10放入10，8-9放入8，7放入7，5-6放入5...

w 2 3 4 5 6 7 8 9 10

2 2 2 2 2 2 2 2 2 2 3 2 3 3 5 5 5 5 5 5 5 2 3 3 5 5 7 8 8 10

最终当每个承重与放入的重量不同时，这个承重就是最小不可求和---4

【示例代码】：

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

class Solution {
public:
    int getFirstUnFormedNum(std::vector<int> &arr, int length) {
        int sum = 0,min = arr[0];
        int i, j;
        for(int i = 0; i < length; i++)
        {
            sum += arr[i];
            min = arr[i] < min ? arr[i] : min;
        }
        std::vector<int> dp(sum + 1, 0);
        for(i = 0; i < length; i++){//有length个数据--有length个阶段
            //{2,  3,  5}
            //i=0--d[10]=2 d[9]=2 d[8]=2 d[7]=2...d[2]=2
            //i=1--d[10]=5 d[9]=5...d[5]=5 d[4]=3 d[3]=3
            //i=2--d[10]=10 d[9]=8 d[8]=8 d[7]=7 d[6]=5 d[5]=5
            for(j = sum; j >= arr[i]; j--){
                //逆序判断背包承重中能够放入的数据
                //当数组中只有2的时候，背包承重从2-10都可以放入2的数值
                //当数组中放入2和3的时候，背包承重从5-10可以放入5，3-4放入3，2只能放入2
```

```
                //当数组中放入2，3，5时，背包承重10放入10，8-9放入8，7放入7，5-6放入5...
                //dp[j-arr[i]]意思是背包承重为j时，如果已经放置了arr[i]的重量后还能放置的最大重量
                if (dp[j] < dp[j - arr[i]] + arr[i])//对每个承重计算当前最大能放置重量
                    dp[j] = dp[j - arr[i]]+ arr[i];//更新背包中能够放入的最大值
                else
                    dp [j] = dp[j];
            }
        }
        //最后当承重为n时，放入的重量不为n则认为是最大不可求和
        for (i = min; i <= sum; i++)
        {
            if (i != dp[i])
                return  i;
        }
        return sum + 1;
    }
};
```

day29-773 有假币

https://www.nowcoder.com/questionTerminal/1d18c0841e64454cbc3afaea05e2f63c

【题目解析】：通过对一堆硬币进行均分称重，判断哪一堆中包含假币（因为假币轻）

【解题思路】：

平均分三份是最快的方法，两份进行称重（对比出三个的重量），后对最重的那份再次进行称重，直到称重

的个数不足2个时则结束，获得假币 如果无法平均分3分则余数要么是1要么是2，因为是要最多称几次，n=n/3+1满足每次取最大 分称3份，取两份一样多的过秤，然后把三份中最多的那份继续分，直到硬币剩余0或1时截至

【示例代码】：

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int main(){
    long long n;int cnt;

    while ((scanf("%lld",&n)) != EOF) {
        if (n == 0)
            break;
        cnt=0;
        while (n >= 2) {
            if (n % 3) {
                //不可以整除则取最差情况：最重的一份是 n/3 + 1个金币
                n=n/3+1;
            }else {
                //可以整除在直接整除，能够获取到最重的一份
                n/=3;
            }
            cnt++;
        }
```

```
        printf("%d\n",cnt);
    };
    return 0;
}
```