C++方向编程题答案

第三周

day14

题目ID：36897-计算日期到天数转换

链接：https://www.nowcoder.com/practice/769d45d455fe40b385ba32f97e7bcded?tpId=37&&tqId=21296&rp=1&ru=/activity/oj&qru=/ta/huawei/question-ranking

【题目解析】：

本题考察日期类，我们课堂已经深入讲解过日期类。

【解题思路】：

用一个数组存放每月的累积天数
输入的日期天数= 当月的天数 + 当月之前的累积天数
如果包含二月，再去判断是否为闰年，如果是闰年，再加1天即可

【示例代码】

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

int main()
{
 int array[12] = {0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334};
 int year;
 int month;
 int day;
 int sum = 0;
 while(cin >> year >> month >> day)
 {
     sum = 0;
     sum += array[month - 1];
     sum += day;
     if(month > 2)
     {
         if((year % 4 == 0 && year % 100 != 0)
                 || year % 400 == 0)
         {
             sum += 1;
         }
     }
     cout << sum <<endl;
 }
}
```

题目ID:45839-幸运的袋子

链接：https://www.nowcoder.com/practice/a5190a7c3ec045ce9273beebdfe029ee?tpId=85&&tqId=29839&rp=1&ru=/activity/oj&qru=/ta/2017test/question-ranking

【题目解析】：

本题的本质是求符合条件的子集个数。

【解题思路】：

每次从全集中选择若干元素（小球）组成子集（袋子）。对于任意两个正整数a,b如果满足a+b>ab，*则必有一个数为1.可用数论证明：设a=1+x,b=1+y，则1+x+1+y>(1+x)*(1+y)，---> 1>xy，*则x，y必有一个为0，即a,b有一个为1.推广到任意k个正整数，假设a1,a2,...ak，如果不满足给定条件，即和sum小于等于积pi。如果此时再选择一个数b,能使其满足sum+b > pi*b，则，b必然为1，且为必要非充分条件。反之，如果选择的b>1，则sum+b <= pi*b，即a1,a2,...,ak,b不满足给定条件。

因此，将球按标号升序排序。每次从小到大选择，当选择到a1,a2,...,ak-1时满足给定条件，而再增加选择ak时不满足条件（ak必然大于等于max(a1,a2,...,ak-1)），继续向后选择更大的数，必然无法满足！此时不必再继续向后搜索。如果有多个1，即当k=1时，sum(1)>pi(1)不满足，但下一个元素仍为1，则可以满足1+1>1*1，
所以要判断当前ak是否等于1，如果等于1，虽然不能满足，组合的个数不能增加，但是继续向后搜索，仍然有满足条件的可能.对于重复数字，组合只能算一个，要去重。

【示例代码】

```
#include<iostream>
#include<algorithm>
using namespace std;
/*
    getLuckyPacket:从当前位置开始搜索符合要求的组合，一直搜索到最后一个位置结束
    x[]: 袋子中的所有球
    n:   球的总数
    pos: 当前搜索的位置
    sum: 到目前位置的累加和
    multi: 到目前位置的累积值
*/
int getLuckyPacket(int x[], int n, int pos, int sum, int multi)
{
    int count = 0;
    //循环，搜索以位置i开始所有可能的组合
    for (int i = pos; i<n; i++)
    {
        sum += x[i];
        multi *= x[i];
        if (sum > multi)
        {
            //找到符合要求的组合，加1，继续累加后续的值，看是否有符合要求的集合
            count += 1 + getLuckyPacket(x, n, i + 1, sum, multi);
        }
        else if (x[i] == 1)
        {
            //如何不符合要求，且当前元素值为1，则继续向后搜索
            count += getLuckyPacket(x, n, i + 1, sum, multi);
        }
        else
        {
            //如何sum大于multi,则后面就没有符合要求的组合了
            break;
        }
        //要搜索下一个位置之前，首先恢复sum和multi
```

```
        sum -= x[i];
        multi /= x[i];
        //数字相同的球，没有什么区别，都只能算一个组合，所以直接跳过
        while (i < n - 1 && x[i] == x[i + 1])
        {
            i++;
        }
    }
    return count;
}
int main()
{
    int n;
    while (cin >> n)
    {
        int x[n];
        for (int i = 0; i < n; i++)
        {
            cin >> x[i];
        }
        sort(x, x + n);
        //从第一个位置开始搜索
        cout << getLuckyPacket(x, n, 0, 0, 1) << endl;
    }
    return 0;
}

```