第 1 讲　暴力求解法---枚举算法

2018.9.8

- “暴力求解法”：把问题所有的可能情况都列举出来，然后根据要求逐一判断，最后找到问题的解。
- 是一种“没有办法的办法”，看上去很“笨”，但在问题范围不是很大的情况下往往很有效，而且准确率也很高，很多的考试设置的部分分，经常使用这种“暴力求解”，能得部分分。

简单枚举算法

- 一一列举问题所有可能的情况，找出问题的解
- 往往使用循环嵌套实现，所有比较简单。

1.基本思想

- 枚举法的基本思想是根据提出的问题枚举所有可能状态，并用问题给定的条件检验哪些是需要的，哪些是不需要的。能使命题成立，即为其解。
- 一般简单的枚举结构：循环+判断语句。

2枚举法的条件

- 虽然枚举法本质上属于搜索策略，但是它与后面讲的回溯法有所不同。因为适用枚举法求解的问题必须满足两个条件：
- ⑴可预先确定状态的元素个数n；
- ⑵状态元素a_1，a_2，…，a_n的可能值为一个连续的值域。

3枚举法的框架结构

- 设a_{i1}—状态元素a_i的最小值；a_{ik}—状态元素a_i的最大值(1≤i≤n)，即$a_{11}≤a_1≤a_{1k}$，$a_{21}≤a_2≤a_{2k}$，$a_{i1}≤a_i≤a_{ik}$，……，$a_{n1}≤a_n≤a_{nk}$。
- 结构格式：

```
for a1←a11 to a1k do
    for a2←a21 to a2k do
        ……
            for ai←ai1 to aik do
                ……
                    for an←an1 to ank do
                        if 状态(a1，…，ai，…，an)满足检验条件
                                then 输出问题的解；
```

4枚举法的优缺点

- 枚举法的优点：
- (1)由于枚举算法一般是现实生活中问题的“直译”，因此**比较直观，易于理解**；
- (2)由于枚举算法建立在考察大量状态、甚至是穷举所有状态的基础上，所以**算法的正确性比较容易证明**
- 枚举法的缺点:
- 枚举算法的效率取决于枚举状态的数量以及单个状态枚举的代价，因此**效率比较低。**
- 直译”枚举：直接根据题意设定枚举对象、范围和约束条件。
- 注意认真审题，不要疏漏任何条件

5.枚举算法的优化：

- （1）减少枚举的状态（等价于减少循环层数）。
- （2）减少状态的枚举次数（减少每个状态的枚举数量）。

Oj简单题目

- **1.7650:不定方程求解**
- **2.1749:数字方格**
- **3.1812:完美立方**

1.7650:不定方程求解

```
#include<cstdio>
#include<iostream>
using namespace std;
int main(){
    int a,b,c,s=0;
    cin>>a>>b>>c;
    for(int x=0;x<=1000;x++)
        for(int y=0;y<=1000;y++)
            if(a*x+b*y==c)s++;
    cout<<s<<endl;
    return 0;
}
```

- 如果每个数都不大于100000？1000000?
- 输入：20 30 1000000

改进1：减少变量的枚举范围

```
#include<cstdio>
#include<iostream>
using namespace std;
int main(){
    int a,b,c,s=0;
    cin>>a>>b>>c;
    for(int x=0;x<=c/a;x++){
        for(int y=0;y<=(c-a*x)/b;y++){
            if(a*x+b*y==c)s++;
        }
    }
    cout<<s<<endl;
    return 0;
}
```

◆改进2：减少枚举的变量，减少循环层数

```
#include<cstdio>
#include<iostream>
using namespace std;
int main(){
    int a,b,c,s=0;
    cin>>a>>b>>c;
    for(int x=0;x<=c/a;x++){
        if((c-a*x)%b==0)s++;
    }
    cout<<s<<endl;
    return 0;
}
```

3526:最简真分数

描述

- 给出n个正整数，任取两个数分别作为分子和分母组成最简真分数，编程求共有几个这样的组合。
- 输入
- 第一行是一个正整数n（n<=600）。
- 第二行是n个不同的整数，相邻两个整数之间用单个空格隔开。整数大于1且小于等于1000。
- 输出
- 一个整数，即最简真分数组合的个数。
- 样例输入
- 7
- 3 5 7 9 11 13 15
- 样例输出
- 17

选择合适的枚举顺序：

```
    cin>>n;
    for(int i=0;i<n;i++)cin>>a[i];
    sort(a,a+n);
    int s=0;
    for(int i=0;i<n-1;i++)
            for(int j=i+1;j<n;j++)
                    if(gcd(a[i],a[j])==1)s++;
    cout<<s<<endl;
    return 0;
}
```

- **例1：砝码称重**
- http://noi.openjudge.cn/ch0201/
- 【问题描述】设有1g、2g、3g、5g、10g、20g的砝码各若干枚（其总重<=1000），求用这些砝码能称出不同的重量个数。
- 【文件输入】输入1g、2g、3g、5g、10g、20g的砝码个数。
- 【文件输出】输出能称出不同重量的个数。
- 【样例输入】1 1 0 0 0 0
- 【样例输出】3

例2 周长最大三角形

- 有n(<=100)根木棍，已知他们的长度(<=10000)，现在从中选出3根木棍组成周长尽可能长的三角形。
- 请计算出最大周长，如果无法组成三角形输出“no”。
- 输入第一行：n
- 第二行:n根木棍的长度。
- 如：
- 输入：
- 6
- 2 3 10 4 1 6
- 输出：
- 13
- （选3 4 6）

例3 换钱问题

- 要将一张100元的大钞票，换成等值的10元、5元、2元、1元一张的小钞票，每次换成40张小钞票，每种至少1张。如，有一种换法：
- 10元： 1 张
- 5元： 5 张
- 2元： 31 张
- 1元： 3 张
- 问：一共有多少种换法。

◆例4. 排列

- 用1~9组成3个三位数abc,def,ghi，每个数字恰好使用一次，要求abc:def:ghi=1:2:3。
- 请按“abc:def:ghi=1:2:3”的格式输出所有的解。

例5.除法 uva 725

- 给你一个数n(2 <= n <= 79)，输出所有形如abcde/fghij=n的表达式，期中 a~j是0~9的一个排列（可以包含前导0，如02345也算）。
- Sample Input
- 61
- 62
- Sample Output
- There are no solutions for 61.
- 79546 / 01283 = 62
- 94736 / 01528 = 62

例6.分数拆分 uva 10976

- 给你一个数k，求所有使得1/k = 1/x + 1/y成立的x≥y的整数对。0 < k ≤ 10000。
- Sample Input
- 12
- sample Output
- 8
- 1/12 = 1/156 + 1/13
- 1/12 = 1/84 + 1/14
- 1/12 = 1/60 + 1/15
- 1/12 = 1/48 + 1/16
- 1/12 = 1/36 + 1/18
- 1/12 = 1/30 + 1/20
- 1/12 = 1/28 + 1/21
- 1/12 = 1/24 + 1/24

【分析】

- 关键是如何枚举x和y的范围，这里不明显。
- 根据要求，首先y>k；又x>=y，所有1/k<=2/y ，即y<=2k，所以y的枚举区间[k+1，2k]，然后求出x，当1/k - 1/y的结果分子为1即为一组解。

◆例7 [NOIP2008] 火柴棒等式

- 给你n根火柴棍，你可以拼出多少个形如“A+B=C”的等式？等式中的A、B、C是用火柴棍拼出的整数（若该数非零，则最高位不能是0）。用火柴棍拼数字0-9的拼法如图所示：

注意：

1. 加号与等号各自需要两根火柴棍
2. 如果A≠B，则A+B=C与B+A=C视为不同的等式（A、B、C>=0）
3. n根火柴棍必须全部用上

- 【输入】
- 输入文件matches.in共一行，又一个整数n（n<=24）。
- 【输出】
- 输出文件matches.out共一行，表示能拼成的不同等式的数目。

- 如：
- 输入：
- 14
- 输出：
- 2
- 因为：2个等式为0+1=1和1+0=1。

- 输入：
- 18
- 输出：
- 9

9个等式为：

0+4=4

0+11=11

1+10=11

2+2=4

2+7=9

4+0=4

7+2=9

10+1=11

11+0=11

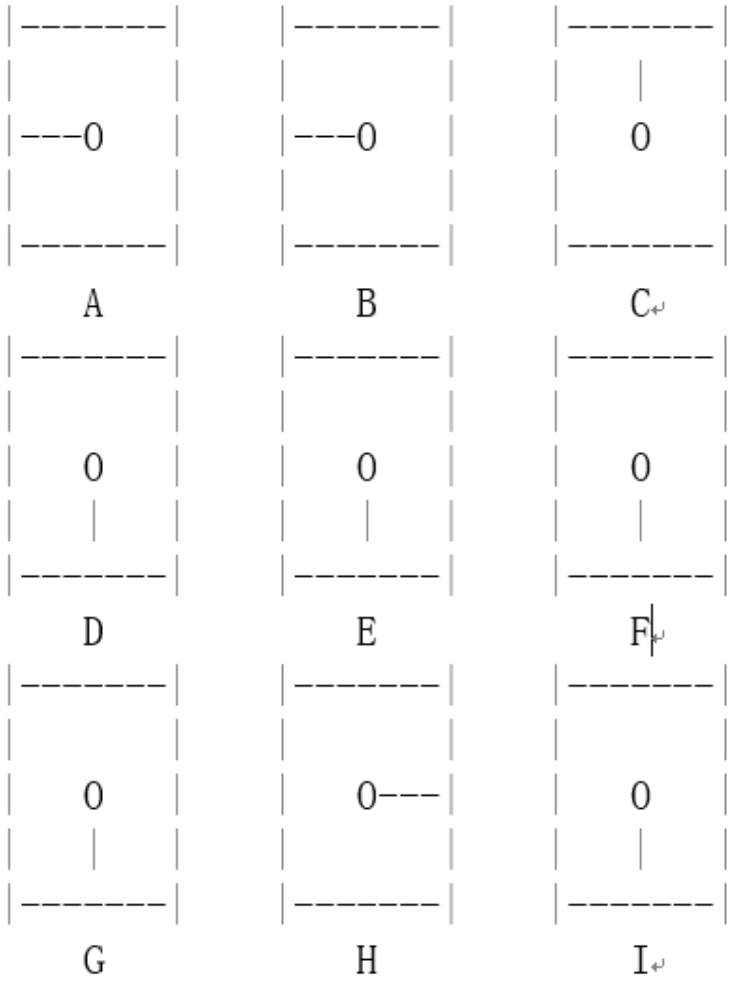
[分析]

- 本题最多24根火柴，等号和加号共用4根火柴，所以A,B,C这3个数字需用20根火柴。我们考查A和B的最大的取值可能：0~9这10个数字所用的火柴数为6,2,5,5,4,5,6,3,7,6，很明显数字1用的火柴棒最少只要2根，不妨让B为1，那么A和C最多可以使用18根火柴，而C>=A，满足条件的A的最大取值为1111。所以枚举A和B的范围是从0~1111，求出c然后计算需要的火柴数量是否为n-4。
- 为了加快速度，可以将0到2222的所有整数需要的火柴棒数目提前算好保存在数组中。

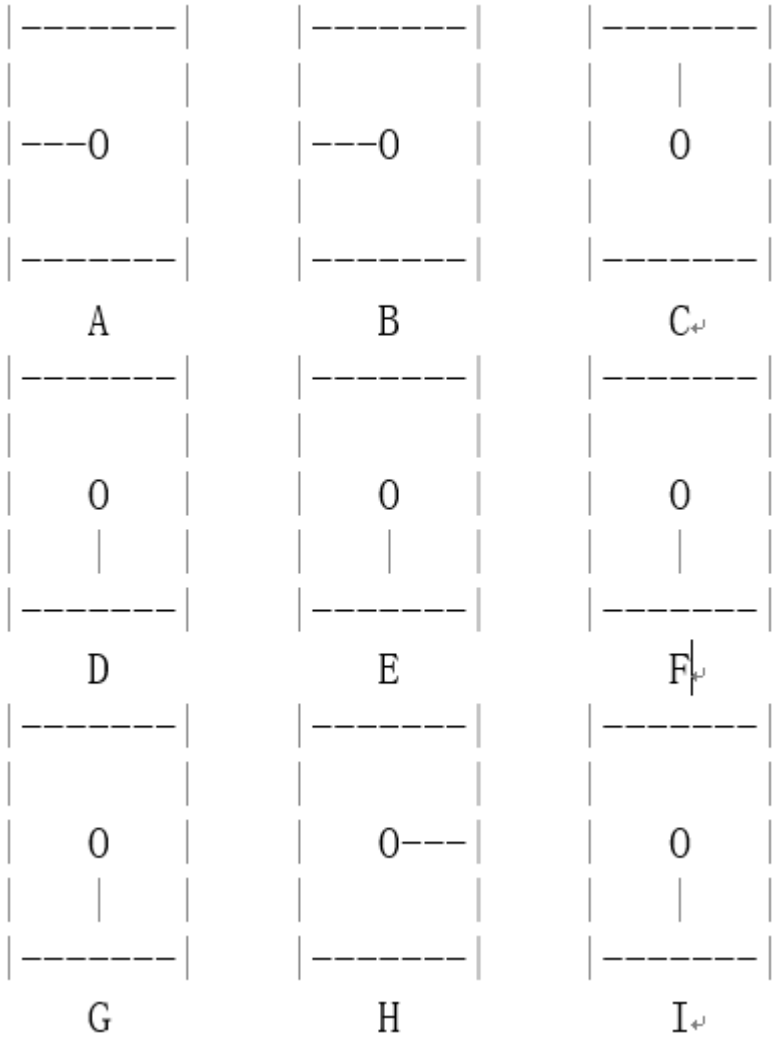
◆例8.拨钟问题（POJ1166，noi1816）

有9个时钟，排成一个3*3的矩阵。

现在需要用最少的移动，将9个时钟的指针都拨到12点的位置。共允许有9种不同的移动。如下表所示，每个移动会将若干个时钟的指针沿顺时针方向拨动90度。



- 移动　影响的时钟
- 1　ABDE
- 2　ABC
- 3　BCEF
- 4　ADG
- 5　BDEFH
- 6　CFI
- 7　DEGH
- 8　GHI
- 9　EFHI



- 输入
- 9个整数，表示各时钟指针的起始位置，相邻两个整数之间用单个空格隔开。其中，0=12点、1=3点、2=6点、3=9点。
- 输出
- 输出一个最短的移动序列，使得9个时钟的指针都指向12点。按照移动的序号从小到大输出结果。相邻两个整数之间用单个空格隔开。

- 样例输入1
- 3 3 0
- 2 2 2
- 2 1 2
- 样例输出1
- 4 5 8 9

样例输入2

1 1 1

2 2 2

3 3 3

样例输出2：

1 1 2 3 3 5 5 7 7 8 8 8 9 9

【分析：】

- 每个时钟最多拨动3次，第4次就回到初始位置了；所以每次移动次数范围：0,1,2,3 。一共9种移动，依次枚举每种移动需要移动几次。
- //x拨动i次的位置是（x+i）%4。

- a[1]—a[9]:9个钟表的初始值；
- c[1]---c[9]:9种移动方式移动的次数；
- b[1]—b[9]:9个表移动后的值。
- d[1]—d[9]:最少的移动次数的方案。

◆例9 四大湖问题

- 上地理课时，四个学生回答我国四个淡水湖大小时说：
- 甲学生：鄱阳湖第3,洞庭湖最大，洪泽湖最小.
- 乙学生：洞庭湖最小，洪泽湖最大，鄱阳湖第2，太湖第3
- 丙学生：洪泽湖最小，洞庭湖第3
- 丁学生：太湖最小，鄱阳湖最大，洪泽湖第2，洞庭第3
- 对于湖的大小，每个学生仅答对一个。
- 请编程判断四个湖的大小。
- 依次输出：鄱阳湖，洞庭湖，洪泽湖，太湖的名次（1,2,3,4表示）。

【分析】

- 如果用abcd分别表示四个湖的名次，那么：
- 怎样描述每个同学答对1个；
- 怎样描述abcd互不相同，正好是1234中的一个。