USACO 第一章通关总结

全面总结各节题目

2009/8/9

ADVENTopLab
ADVENTop

HTTP://hi.baidu.com/adventop

USACO 第一章通关总结

By ADVENTop

原来用 Pascal 做过一遍第一章,而这次使用 C++有了更深的体会,更能体会到算法使用的精妙,不单是算法的本身,从题目分析,往往能找到更适合的算法.这也是算法设计的重要步骤之一.总的来说第一章我初步学到了基础,与浅层次的分析问题的能力.主要学会的或者说了解的算法有:**枚举,模拟,构造,基本的贪心,DFS,BFS,位运算和最简单的 DP**.

第一章题目总体类型分布:

section 1.0	DONE	2009.05.25	TEXT Introduction
Section 1.1	DONE	2009.07.31	TEXT Submitting Solutions
	DONE	2009.05.25	PROB Your Ride Is Here 模拟
	DONE	2009.05.28	TEXT Contest Problem Types
	DONE	2009.05.28	TEXT Ad Hoc Problems
	DONE	2009.05.28	PROB Greedy Gift Givers 模拟
	DONE	2009.05.29	PROB Friday the Thirteenth 模拟(mod 运算)
	DONE	2009.05.29	PROB Broken Necklace 模拟(数组使用)
Section 1.2	DONE	2009.06.08	TEXT Complete Search
	DONE	2009.06.08	PROB Milking Cows 离散化
	DONE	2009.06.23	PROB Transformations 枚举(矩阵)
	DONE	2009.06.29	PROB Name That Number 枚举(构造法)
	DONE	2009.07.18	PROB Palindromic Squares 枚举
	DONE	2009.07.19	PROB Dual Palindromes 枚举
Section 1.3	DONE	2009.07.19	TEXT Greedy Algorithm
	DONE	2009.07.19	PROB Mixing Milk 贪心
	DONE	2009.07.29	PROB Barn Repair 贪心
	DONE	2009.07.31	TEXT Winning Solutions
	DONE	2009.07.31	PROB Calf Flac 枚举
	DONE	2009.07.31	PROB Prime Cryptarithm 枚举+构造
Section 1.4	DONE	2009.07.31	TEXT More Search Techniques
	DONE	2009.08.02	PROB Packing Rectangles 完全搜索
	DONE	2009.08.02	PROB The Clocks BFS||DFS||Maths
	DONE	2009.08.03	PROB Arithmetic Progressions 枚举(构造)
	DONE	2009.08.03	PROB Mother's Milk DFS

Section 1.5	DONE	2009.08.04	TEXT Introduction to Binary Numbers
	DONE	2009.08.04	PROB Number Triangles DP(BFS)
	DONE	2009.08.04	PROB Prime Palindromes 类 BFS 构造
	DONE	2009.08.05	PROB SuperPrime Rib DFS
	DONE	2009.08.07	PROB Checker Challenge 经典 DFS(位运算)

◎第一节:

第一节的内容简单经典,仅仅四道就考查了应有的基本功, 第一节的算法基本上是模拟,千里之行,始于足下.如果目标长远就永远不要忽略基础,现在的砖,是为将来垒的.

1.1.1:这是 USACO 的入门题目,只要会编程,就可以写出这道题,但是小小的题目却蕴含了最最基本的 Hash 原理,虽然没用接触过 Hash,但是任何一个高深的技术与算法都是从最基本的算法组合,演变过来的.所谓算法设计与分析,就是能从最简单的问题中看出深奥的道理.

1.1.2:这道题目考查了循环与记录的使用,纯粹考编程能力.

1.1.3 虽然第一节的对于 OIer 来说再简单不过了,但我认为这道题目对于初学者仍然是一道好题,他引入了计算机中几乎是最重要的计算方法之一——Mod 运算, 同时也不能向前两题一样瞎模拟, 还需好好分析日子的计数方法, 深刻理解 Mod 运算对于以后的学习受益匪浅.

1.1.4 这是一道考验循环的问题,也就是怎样用一维数组模拟环,道理很简单,只要复制一遍原来的内容就行了,然后是处理方法,如果你的 DP 功底好,使用类似 DP 的方法即可,如果直接模拟的话,普遍要写两遍主要操作,我创了一种一步到位的:

```
for(int i=0;i<num*2;i++)
    {
        char l=beads[i],r=beads[i+1];
        int nl=0,nr=0,lj=0;
        for(int j=i;j>=0;j--)
            if ((beads[j]==l||beads[j]=='w')&&lj<num)
            {
                if (j-1>=0) if (l=='w') l=beads[j-1];
                ++nl;
                ++lj;
            }
                else break;
        for(int j=i+1;j<num*2;j++)
            if ((beads[j]==r||beads[j]=='w')&&lj<num)
            {
                if (j+1<num*2) if (r=='w') r=beads[j+1];
                ++nr;
                ++lj;
            }
                else break;
```

```
            if (nr+nl>maxn) maxn=nl+nr;
    }
```

这样就大大缩短了程序长度.

◎第二节:

第二节的题目是所有算法的奠基之一——完全搜索(也叫穷举,枚举),为什么这么说,因为这是真理,试想一下,有的算法从某种程度上讲,就是穷举的优化,看似效率高的 DP,也要把所有的状态算出,只是不走重复的路.类似的还有很多.这节开始就要进入算法的训练了.

1.2.1:开始我不会做这道题,后来通过这道题,我学会了离散化法,就是对离散的数据的处理,是高效的算法,而这道题目也让我豁然开朗,曾经听起来神秘的算法其实并不难理解,我以前也做过类似的题目,但不知到这就是离散化法.

1.2.2:枚举+模拟:其实还是对基础的考查,计算矩阵翻折的坐标,而且除了翻转不需要交换元素.

1.2.3:这是一道好题,直接看出你对算法时间复杂度的认识,我也领悟了构造的妙处,构造就是减少时间复杂度的金钥匙的一种.好的构造能使问题迎刃而解.

1.2.4:这道题目主要是枚举,以及进制转换,c/c++有一个得天独厚的优点:就是 ASCII 码可以直接运算,同时这道题也训练了我模块化操作的思想.

1.2.5:只要函数编的好,直接枚举即可,int F(int x,int y),将 x 转换为 y 进制.

◎第三节:

第三节也是基本的算法:贪心,主要是是否有局部最优可以反映全局最优,证明也许我还不行,但是通过训练,我也悟出了贪心法的应用方法.

1.3.1:直接贪心,或许还有更直接的办法,从头加到尾,Why?这是一种相互平衡的关系,观察数据可以得知量多数也多,量少数也少.也许这是最直接的贪心.也就是说任何算法不要拘泥于形式,万变不离其宗,到此为止这个宗我觉得应该是:模拟,枚举,贪心,DFS,BFS.

1.3.2:我采用反着想的办法:先遮住一个大板，然后从中除去一定数量的尽量大的板子就行了.我的反思是，有时换一种角度看待问题或许能有更大的收获。

1.3.3:这是一个枚举题目，从中间向两边扩展，同时需要判断奇偶，用 KMP 效率会更高，这也是对字符串处理的考验,通过这道题目,我主要学到了 fin.get(),fin.eof()等成员函数,了解到 string 类的无限长度等。

1.3.4:这是一道构造题目，就是依次枚举，考点主要在数字判定上，只要枚举，筛选（不符合条件的都去掉），然后分离数字判定就行了，可以用 Bool 数组优化判断函数，是这个模块

的时效降为 O(1).

◎第四节：

这是我比较喜欢的一节，因为出现了 DFS,BFS 等常用算法，这节开始题目变得有意思了，算不上难，但题题经典。充分考察了分析能力。

1.4.1:这道题很无敌，很考验逻辑与分析能力，就是枚举，还是我第一次看到能把枚举出成这样的，可以说第六种才是本题的关键，因为它包含了最大边与嵌套的思想，需要交叉匹配，难就难在如何测试这点，所以本题虽出在第一章，但也不失为一道经典题目。

1.4.2:这题解法很多，我比较喜欢 BFS,同时我也更加理解了 BFS 的精妙，因为 BFS 在于层次性的掌握，这是至关重要的一点，层次的掌握也是很多问题的出发点。

```
while (head<tail)
    {
        head++;
        //expand
        for(int i=jl[head][0];i<=9;i++)
        if (jl[head][i]<3)
        {
            tail++;
            for(int j=1;j<=9;j++)
            {
                dl[tail][j]=(dl[head][j]+state[i-1][j-1])%12;
                dl[tail][0]+=dl[tail][j];
            }
            for(int k=1;k<=9;k++)
                jl[tail][k]=jl[head][k];//从上一个状态扩展得到
            jl[tail][i]++;//记录
            jl[tail][0]=i;//非递减扩展
            if(dl[tail][0]==0)
            {
                head=tail;
                break;
            }
        }
    }
```

1.4.3:这是一道构造搜索题目，先构造优化时间复杂度，然后考虑先举 b 还是 a，两个条件都允许的题目就要好好分析一下了，这是的时效一般是不一样的。

1.4.4:经典 DFS,这题有 6 种状态，这些状态的操作简单，主要是退出条件，其实到来到去，最终会形成死锁，所以需要用一个 bool 数组记录所有产生的状态，然后凡是已经有的状态不要再搜索，直到所有状态搜索结束就行了。退出条件的设计是 DFS 的一个重点，每种 DFS

退出条件是不一样的。

◎第五节：

这也是我比较喜欢的一节，因为这节的内容也很有意思，基本上是综合题了，就当是单元测验。

1.5.1:非常经典的 DP,推导出状态转移方程，不难出解。

1.5.2:这道题目，我独创了一种类似 BFS 的方法，是目前这道题目最短的解题方法，但是并不是 BFS,而是同时需要 4 个指针，2 个作为操作指针，2 个作为层次转换指针，我个人认为这道题目层次分明，所以想到了这种办法。

```
int temp=tail,tt=1;
    while(gz<=b)
    {
        head=tt;
        for(int i=0;i<10;i++)
        {
            for(int j=head;j<=tail;j++)
            {
                temp++;
                gz=cheng(10,dl2[j]+1)*tw[i]+dl[j]*10+tw[i];
                if (gz>b) return 0;
                else
                {
                    dl2[temp]=dl2[j]+2;
                    dl[temp]=gz;
                    if(prime(gz)&&(gz>=a)) fout<<gz<<endl;
                }
            }
        }
        tt=tail+1;
        tail=temp;
    }
```

1.5.3:这也是一种构造优化题目，使用 DFS,可以优化各个数位，不需要都进行判断，然后在每个深度先算出来，这样可以减少时间复杂度。

1.5.4:经典的压轴 DFS,使用位运算做。(其实就是用 DFS 可以考虑对称性，奇与偶是不一样的）通过这道题目我也学会了基本的 Bit 运算。可以说每题都有收获。

```
int dfsbit(long bit,long left,long right)
{
    int temp=0,tt=0;//必须是临时变量
    if (bit==last) total++;
```

```
    else
    {
        temp=last&(~(bit|left|right));//检查空位
        while(temp!=0)
        {
            tt=temp&(temp^(temp-1));//取出最右边的 1.
            temp-=tt;
            if(tt!=0) dfsbit(bit+tt,(left+tt)<<1,(right+tt)>>1);
        }
    }
    return 0;
}
```

★章末总结：

到此第一章就彻底通关大吉了，奠定了基础，是一套不错的训练题材，主要学会的不单是算法，更是一种灵魂，一种分析思考的灵魂，这样在以后的道路上才会将思想推到极限。建立一个高塔，基层太重要了，学会了方法，奠定了基础才可以算是入门。(END)

2009/8/9