搜索的进阶与位运算和DP的引入

译题 - numtri

Number Triangles 数字金字塔

描述

考虑在下面被显示的数字金字塔。
写一个程序来计算从最高点开始在底部任意处结束的路径经过数字的和的最大。每一步可以走到左下方的点也可以到达右下方的点。

7
3 8
8 1 0
2 7 4 4
4 5 2 6 5

在上面的样例中,从7 到 3 到 8 到 7 到 5 的路径产生了最大和:30

格式

PROGRAM NAME: numtri
INPUT FORMAT:
(file numtri.in)
第一个行包含 R(1<= R<=1000) ,表示行的数目。
后面每行为这个数字金字塔特定行包含的整数。
所有的被供应的整数是非负的且不大于100。
OUTPUT FORMAT:
(file numtri.out)
单独的一行包含那个可能得到的最大的和。

SAMPLE INPUT

5

7
3 8
8 1 0
2 7 4 4
4 5 2 6 5

SAMPLE OUTPUT

30

解题报告:

简单的动态规划
设dp[i,j]表示到达第i层第j个最大的分数，转移方程：
dp[i,j]=max(dp[i-1,j],dp[i-1,j-1])+num[i,j]（考虑边界情况）
我们只需知道上层的dp情况，所以可以用滚动数组来记录，时间复杂度O(n^2)，空间复杂度O(n)。

译题 - pprime

Prime Palindromes 回文质数

描述

因为151即是一个质数又是一个回文数(从左到右和从右到左是看一样的)，所以 151 号是回文质数。
写一个程序来找出范围[a,b](5 <= a < b <= 100,000,000)间的所有回文质数;

格式

PROGRAM NAME: pprime
INPUT FORMAT:
(file pprime.in)

第 1 行: 二个整数 a 和 b .
OUTPUT FORMAT:
(file pprime.out)
输出一个回文质数的列表，一行一个。

SAMPLE INPUT

5 500

SAMPLE OUTPUT

5
7
11
101
131
151
181
191
313
353
373
383

题解 - pprime

这道题有两种思路：
1. 用筛法求出1..1e8范围内的素数，然后判断每个素数是否是回文数。
2. 生成1..1e8范围内的回文数，然后判断它是否是素数。

思路1的复杂度是O(n)，思路2的复杂度是O(√n*√n)=O(n)，从复杂度来看两种思路没有差别。但思路1用筛法用素数要开O(n)的数组，在n=1e8是就是90M，超出了空间限制，而思路2的空间复杂度是O(1)的，所以我们用思路2。
如何按照从小到大的顺序生成回文数呢？
设生成位数为l的回文数，若l是奇数，那么从小到大枚举(l+1) div 2位的数，然后复制翻转生成一个回文数；若l是偶数，那么从小到大枚举l div 2位的数，然后复制翻转生成一个回文数。上诉两个过程交替进行就可以从小到大生成回文数了。

很有效的优化：任意偶数长度的回文数都不可能为质数（除了11），因为它能被11整除，而11恰好只有自身和1两个因子。

译题 - sprime

Superprime Rib 特殊的质数肋骨

描述

农民约翰的母牛总是生产出最好的肋骨。你能通过农民约翰和美国农业部标记在每根肋骨上的数字认出它们。

农民约翰确定他卖给买方的是真正的质数肋骨,是因为从右边开始切下肋骨,每次还剩下的肋骨上的数字都组成一个质数,举例来说:

7 3 3 1

全部肋骨上的数字 7331是质数;三根肋骨 733是质数;二根肋骨 73 是质数;当然,最后一根肋骨 7 也是质数。

7331 被叫做长度 4 的特殊质数。

写一个程序对给定的肋骨的数目 N (1<=N<=8),求出所有的特殊质数。数字1不被看作一个质数。

格式

PROGRAM NAME: sprime

INPUT FORMAT:

(file sprime.in)

单独的一行包含N。

OUTPUT FORMAT:

(file sprime.out)

按顺序输出长度为 N 的特殊质数,每行一个。

SAMPLE INPUT

4

SAMPLE OUTPUT

2333
2339
2393
2399
2939
3119
3137
3733
3739
3793
3797
5939
7193
7331
7333
7393

题解 - sprime

分析

DFS
以位数为深度搜索，每次枚举当前数的末尾，并检查是不是质数，如果不是就剪枝。
小优化：质数的末尾不可能是(0,2,4,5,6,8)，只可能是(1,3,7,9)，一位数的质数特殊考虑。这样枚举情况就只有4种。
用预处理…否则貌似会超时…先生成1位的质数,再用1位的质数生成2位,然后用2位的生成3位…就这样递归下去….最后一位直接全部输出然后Halt就好了… 下面代码是表示在原质数的基础的后面再加上一个奇数(因为质数除了2不可能是偶数)
tmp:=a[i-1,k]*10+b[l];
生成tmp之后再判断tmp是不是质数,易证循环只要到 根号tmp 过就好了

```
for k:=2 to trunc(sqrt(i)) do
begin
if i mod k=0 then
begin
jump:=true;
break;
end;
end;
```

如果是质数就把这个数记录到长度是这个数的长度的序号的数组里…

```
a[j,c]:=i;
```

这样预处理就不会超时了….最后只要把要求的长度的质数输出来就行了…

译题 - checker

Checker Challenge 跳棋的挑战

描述

检查一个如下的6 x 6的跳棋棋盘，有六个棋子被放置在棋盘上，使得每行、每列只有一个，每条对角线(包括两条主对角线的所有对角线)上至多有一个棋子。

列号

```
1 2 3 4 5 6
-----------------------
1 | | 0 | | | | |
-----------------------
2 | | | | 0 | | |
-----------------------
3 | | | | | | 0 |
-----------------------
4 | 0 | | | | | |
-----------------------
5 | | | 0 | | | |
-----------------------
6 | | | | | 0 | |
-----------------------
```

上面的布局可以用序列2 4 6 1 3 5来描述，第i个数字表示在第i行的相应位置有一个棋子，如下：
行号 1 2 3 4 5 6
列号 2 4 6 1 3 5
这只是跳棋放置的一个解。请编一个程序找出所有跳棋放置的解。并把它们以上面的序列方法输出。解按字典顺序排列。请输出前3个解。最后一行是解的总个数。
特别注意：对于更大的N(棋盘大小N x N)你的程序应当改进得更有效。不要事先计算出所有解然后只输出，这是作弊。如果你坚持作弊，那么你登陆USACO Training的帐号将被无警告删除

格式

PROGRAM NAME: checker
INPUT FORMAT:
(checker.in)
一个数字N (6 <= N <= 13) 表示棋盘是N x N大小的。
OUTPUT FORMAT:
(checker.out)
前三行为前三个解，每个解的两个数字之间用一个空格隔开。第四行只有一个数字，表示解的总数。

SAMPLE INPUT

6

SAMPLE OUTPUT

2 4 6 1 3 5
3 6 2 5 1 4
4 1 5 2 6 3
4

分析

DFS+优化

因为要遍历整棵搜索数，所以用DFS，我们可以在搜索时加入剪枝，记录横行和两条斜线是否被攻击，每次只放在不受任意方向攻击的地方。这个剪枝过最后一个数据需要2s，超时。

考虑对称

如果n是偶数，第一行枚举前一半的数，这样每搜索出一种方案后，沿中线对称过来又是一种方案，并且因为第一行的数小于一半，对称过来的方案总大于原方案（即在搜索树中后被搜索到），不会出现重复的情况。

如果n是奇数，先在中间行和中间列放子，并且位置都不超过半数(<n div 2)，且中间行>中间列，这样每搜索出一种方案，把它对称旋转后一共有8种方案，因为中间行和中间列的的不出现重复，所以8种方案不重复。这样只需枚举原来的1/8就可以了。这样就完全可以过了。{注意放在正中间位置的时候}

使用链表

还可以再优化，用链表（或用数组模拟）记录每行待选的位置，进行插入和删除，这样就不用记录横行是否被攻击了，并且每次枚举位置的个数减少。

最后一位算出来

由于n行每一行都要有一个，那么所有的皇后的位置之和必然为n×(n+1)/2，在dfs的时候记录已经枚举的皇后位置之和，那么最后一个可以直接算出来。这样的话最后一个点0.907s。

DFS+位运算

可以考虑将问题分为两个子问题来处理:

a)求n皇后具体的方案;

b)求n皇后方案的个数;

对于问题a,因为只取前三个,用最简单的DFS即可办到.

对于问题 b,可以考虑采用位运算,来递归模拟试放(考虑的条件与 DFS 类似),

但不用储存具体方案.