国家集训队2015第一次作业
部分题目解析

中山纪念中学

杨乐

Contents

1 Evil(329E) 4

2 Polygon(306D) 5

3 Xenia and String Problem(356E) 5

4 Yaroslav and Arrangements(301E) 6

5 Transferring Pyramid(354D) 7

6 Mystic Carvings(297E) 8

7 Flights(241E) 8

8 Sereja and Squares(314E) 9

9 GCD Table(338D) 9

10 Google Code Jam(277D) 10

11 Tennis Rackets(309D) 11

12 Princess and Her Shadow(317E) 12

13 Mirror Room(274E) 13

14 Dividing Kingdom(260E) 13

15 Doodle Jump(346E) 14

16 Greedy Elevator(257E) 14

17 Cubes(243D) 15

18 Tournament-graph(323B)	16
19 Number Challenge(235E)	16
20 Graph Game(235D)	17
21 Two Permutations(323C)	18
22 Candies Game(341E)	18
23 Dima and Game(273E)	19
24 Friends(241B)	19
25 Reclamation(325D)	20
26 Levko and Game(360E)	20
27 Endless Matrix(249E)	21
28 Biologist(311E)	22
29 Polo the Penguin and Lucky Numbers(288E)	22
30 Binary Key(322E)	23
31 Ciel and Flipboard(321D)	23
32 Maxim and Calculator(261E)	24
33 Ksusha and Square(293D)	24
34 Donkey and Stars(249D)	25
35 Liars and Serge(256D)	25
36 Shaass and Painter Robot(294D)	26
37 Levko and Sets(360D)	26
38 k-Maximum Subsequence Sum(280D)	27
39 Two Sets(251D)	27
40 Ladies' Shop(286E)	28
41 Close Vertices(293E)	28
42 White, Black and White Again(306C)	28

43 Have You Ever Heard About the Word?(319D) 29

44 Piglet's Birthday(248E) 29

45 Rhombus(263E) 30

46 Cow Tennis Tournament(283E) 30

47 Rectangles and Square(335D) 31

48 The Great Julya Calendar(331C3) 31

49 Positions in Permutations(285E) 32

50 Colorful Stones(264D) 32

1 Evil(329E)

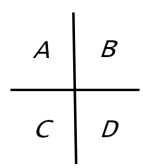
题目简述

给出一个二维平面上的N个点，重新排列点的顺序，使得相邻两点的曼哈顿距离总和（包括第一个与最后一个的曼哈顿距离）最大。

算法讨论

首先考虑可以达到的上界。单独考虑X轴，设N个点的X坐标分别为$X_1, X_2, \cdots, X_N$，以及其排列为$P_1, P_2, \cdots, P_N$，那么其代价为$|X_{P1} - X_{P2}| + |X_{P2} - X_{P3}| + \cdots + |X_{Pn-1} - X_{Pn}| + |X_n - X_1|$。把绝对值展开后，一共会有$N$个正项和$N$个负项，每个元素都会出现两次。显然地，把每一个元素复制一遍得到$2N$个元素并从小到大排序后，后N个数的和减去前N个数的和即为上界。

那么把X轴与Y轴综合起来后还会不会达到这个上界呢？为了方便讨论，我们按X轴与Y轴上中位数把平面划分成4个区域A, B, C, D:



若想达到在X轴上的最优值，必须在AC与BD之间交替；Y轴这要求AB与CD上交替。综合起来，A中的点与D中的交替，B中的与C中的交替才能达到上界。但直观上，是不能达到上界的，因为不能排列使得AD交替转到BC交替。下面分情况讨论：

N为偶数时，满足$|A| = |D|$且$|B| = |C|$（不详细证明），那么是一定不能达到上界的；我们可以找一个中转的位置，使之减少量最小，找中间的两个数显然满足要求。

N为奇数时，满足$||A| - |D|| = 1$且$||B| - |C|| = 1$，若交点处没有点，则可以达到上界。若交点处存在点，与偶数的情况类似，需要一个中转的调整。

但值得注意的是，若$|A| = |D| = 0$或$|B| = |C| = 0$，则不需要交替，可以达到上界。

总时间复杂度为$O(N)$，空间复杂度为$O(N)$。

TAG 分类讨论

2 Polygon(306D)

题目简述

构造一个N边形，要求每个内角相同，并且每条边长互不相同。

算法讨论

本题我使用的是随机调整法。首先生成一个正多边形，枚举一条边，在垂直于边的方向是是可以上下移动的。注意精度问题就可以通过，时间复杂度$O(N)$，空间复杂度$O(N)$。

本题还有较多解法，不再详述。

TAG 几何/随机

3 Xenia and String Problem(356E)

题目简述

定义一个长度为L的字符串是Gray串，满足两个条件中的一个：1)$L = 1$；2)L是奇数，位于字符串的中间位置的字符只出现过一次，并且左半边的字符串与右半边的字符串相等，都是Gray 串。

给出一个长度为N的字符串，每一个长度为L的子串，如果是Gray串，则会给答案贡献L^2。修改至多一个位置上的字符，求出最大的总贡献。

算法讨论

考虑没有修改的情况：Gray串的长度只能是1，3，7，15，$\cdots$，$2^x - 1$，一共只有$logN$ 种，依次枚举每种长度和中央位置即可。

如果修改一个位置上的字符，则会有两种效果：

1)原有的Gary串因为修改而减少

枚举每一个中心位置以及可行的长度，区间赋值。

2)新增一些Gray串

枚举每一个中心位置以及可行的长度，若这个字符串可以通过修改一个字符变成Gray 串，会满足a)中心位置不满足；b)左边或右边不满足。分类讨论即可。

若使用Hash来进行字符串比较，则时间复杂度可达到$O(NLog^2N)$，空间复杂度为$O(NLogN)$。

TAG 分类讨论/字符串/哈希

4 Yaroslav and Arrangements(301E)

题目简述

当一个长度为L的序列A是good的，它满足$|A_i - Ai - 1| = 1, |A_L - A_1| = 1$;

当一个长度为L的序列B是great的，它满足$L \leq N$、$B_i \leq B_{i+1}, B_i \leq M$、对其重新排列后，存在至少一种，但不超过$K$个good的序列。

给出N, M, K，求有多少个序列是great的。

算法讨论

本题最重要的是确定DP的状态，首先明确几点：

1)DP无后效性，由于B序列是递增的，从小确定B序列中每一个数;

2)因为统计方案数时要求不同的序列，所以确定数时，相同的要一起考虑。

3)若当前序列最后一个数为a，加入$a + 1$的方式是找到序列中的一个a，在其后加入两个数$a + 1a$。

4)统计方案数时和插入的方案有关，若有p个a，要插入q个$a + 1$，可以发现总插入方案数为$C(p + q - 1, p - 1)$。

最终得到转移方程：设$F[L][Max][Cnt][Way]$为长度为L的序列，最大值为Max，有Cnt 个，目前的总方案数为Way，这样的序列有多少个。状态可以更新至$F[L + Q * 2][Max + 1][Q][Way * C(Cnt + Q - 1, Cnt - 1)]$。

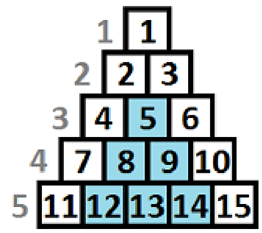
实现时注意前三维数组的大小是可以优化的，同时枚举优化的复杂度就可以通过全部数据。时间复杂度为$O(N^5)$，空间复杂度为$O(N^4)$。

TAG 动态规划/组合数/设计状态

5 Transferring Pyramid(354D)

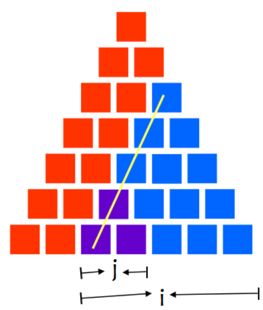
题目简述

给出一个形如下图的金字塔。上有K个格子被标记，要求至少被覆盖一次。覆盖的方式有两种：单个覆盖一个格子，代价为3；覆盖形如蓝色三角形的区域，代价为2+三角形的面积。求出最小的覆盖代价。



算法讨论

先考虑最简单的DP：设$F[i][j]$为覆盖图中蓝色部分内标记格子的所需最小代价。



由于每一个放置的三角形不会出现严格覆盖的情况，所以在第i列上只会放置一个三角形。假设这个三角形的放置在第k 行$(j < k \le i)$，$S[i][j]$为第j 列第i行以上有多少个标记点。

$$F[i][j] = Min\{F[i-1][k-1] + \frac{k(k+1)}{2} + 2 + 3 * S[i][k+1]\}$$

观察$F[i][j]$和$F[i][j+1]$，可得到下列的转移：

$$F[i][j] = Min\{F[i][j+1], F[i-1][j] + \frac{(j+1)(j+2)}{2} + 2 + 3 * S[i][j+2]\}$$

或者在第i列上不放置任何三角形：$F[i][j] = F[i-1][j-1] + 3 * S[i][j+1]$

这样转移的时间复杂度为$O(N^2)$，使用滚动数组的空间复杂度为$O(N)$。

我们发现许多状态是冗余的：由于$F[i][j] <= 3K$，所以$j >= \sqrt{6K}$是不会被转移的。时间复杂度为$O(N\sqrt{K})$。

TAG 动态规划

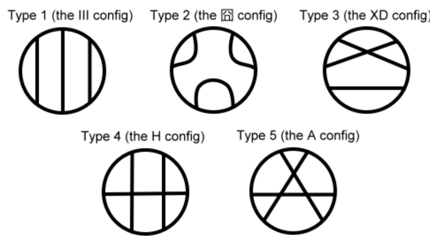
6 Mystic Carvings(297E)

题目简述

给出一个圈，上面有$2n$个点，存在n条边把它们两两不重复的相连。从中选取三条边，标记其所选的6 个点。定义两标记点的距离为从一点到达另一点所经过的标记点的最小值。问有多少种选取方法。

算法讨论

通过讨论，任意选取3条边，所形成的形状共有下列5种形式：



答案即为类型2、5的和。有两种方法计算之：按定义直接算，或者用总方案数减去其余类型的总和，显然这道题用后一种方法更为明智。

逐个考虑每一种形状；枚举形状1的中间边i，通过扫描左边被包含边的数量x_i，以及右边被包含的边的数量y_i，就可以得知这种形状的数量为$x_i * y_i$。

对于形状3和4，特点是至少有一组不相交，但至少有一组相交。我们枚举存在相交的一条边i，计算与它相交边的数目z_i，形状3和4的总数量就为$(x_i + y_i) * z_i$。

x_i, y_i, z_i的计算可以用树状数组维护，总时间复杂度为$O(NLogN)$，空间复杂度为$O(N)$。

TAG 计数问题/补集思想/数据结构

7 Flights(241E)

题目简述

一张有向无权图，给每条边定边权1或2，使得从1到N的每条路径的边权和均相等。

算法讨论

先筛选既可以从1到达也可以到达N的点。再对距离数组进行差分约束求解。时间复杂度为$O(NM)$，空间复杂度为$O(N + M)$。

TAG 差分约束系统

8 Sereja and Squares(314E)

题目简述

定义一个合法的序列：满足括号序列，并且每一对括号中左括号为小写字母，右括号为对应的大写字母。给出一个只有小写字母与'?'的序列，问有多少种方式在'?'中填字母使得序列合法？（不允许填字母x）

算法讨论

由于括号序列并没有特殊的要求，所以我们设$F[i][j]$为前i项含有j个未匹配的左括号的方案数。并得到方程：

$$F[i][j] = \begin{cases} F[i-1][j-1] & S[i] \text{ is a lowercase} \\ F[i-1][j-1]*25+F[i-1][j+1] & S[i] \text{ is '?'} \end{cases}$$

观察转移方程，在情况1下，我们可视整个数组向右移动，维护一个指针代表数组的位移量；情况2 下则需要线性转移来维护。时间复杂度$O(N^2)$，空间复杂度$O(N)$。

TAG 动态规划/动态规划优化

9 GCD Table(338D)

题目简述

给出一个长度为N的数组A_i，找出X与Y使得对于$1 \leq i \leq N$，满足$GCD(X, Y+i) = A[i]$。

算法讨论

记A中所有数GCD为G，X一定是等于G若干倍，更一步，$X = G$。因为如果存在某个位置i使得$GCD(X*p, Y+i) = A[i]$且$GCD(X, Y+i)! = A[i]$，这显然是不可能的。接下来可以解对于Y的若干个同余方程（$Y - i = 0(MODA[i])$），再带入原来的进行检验。总时间复杂度为$O(NLogN)$，空间复杂度为$O(N)$。

TAG 数论/线性同余方程组

10 Google Code Jam(277D)

题目简述

模拟Google Code Jam的比赛机制，问最多能取得多少分，并求出在此前提下最小的罚时时间。

算法讨论

考虑不计罚时的情况：因为分数之间是独立的，那么考虑前i个问题，已经花费时间为j 的情况下的最大得分$F[i][j]$，可以得到递推的方程：

$$F[i][j] = Max \begin{cases} F[i-1][j] & \text{不取问题}i \\ F[i-1][j-tS_i] + sS_i & \text{完成Small部分} \\ F[i-1][j-tS_i-tB_i] + sS_i + sB_i * (1 - Fail_i) & \text{完成Big 部分} \end{cases}$$

考虑确定作答顺序的情况：显然地，我们应完成所有的Small之后再完成Big。Small 部分的顺序无关紧要；对于Big 部分，考虑都需要完成Big的问题i与j，单独地拿出比较：

i在j前: $T_i * (1 - Fail_i) * Fail_j + (T_i + T_j) * (1 - Fail_j)$

j在i前: $T_j * (1 - Fail_j) * Fail_i + (T_i + T_j) * (1 - Fail_i)$

i比j优，那么有

$$\frac{T_i * Fail_i}{1 - Faili} < \frac{T_j * Fail_j}{1 - Failj}$$

我们将上述关键字排序后，顺序依次确定，就可以计算出最小的罚时。

回到原问题，我们可以结合前两个思想来得出解法：先排序，再用DP记录答案二元组求出答案。值得注意的是，当Fail值为1 时，已经没有做Big的需要了，这时我们便可忽略这个问题。

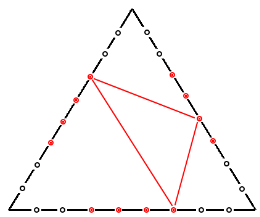
TAG 动态规划/排序

11 Tennis Rackets(309D)

题目简述

给出一个正三角形的球拍。每一条边上有n个洞把这条边$(n+1)$等分，并且左右的m 个洞是不可用的（形如下图）。求有多少种方法，在三边各选一点，形成一个钝角三角形。

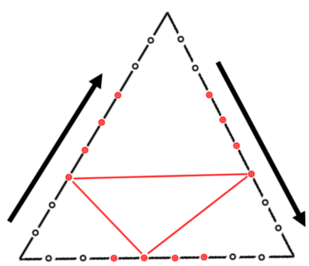
Figure 1: $n = 8, n = 2$



算法讨论

我们枚举钝角的位置，若在一边上枚举，答案乘以3就为最终答案（因为一个三角形最多拥有一个钝角）。同时根据对称性，可以只枚举到中间位置。

在另一边从下往上枚举顶点的位置，则第三边上满足的点的数目在不断减少。



通过构造坐标可以判断三点组成的是否为钝角三角形。实现时应注意优化判断的时间。时间复杂度$O(N^2)$，空间复杂度$O(1)$。

TAG 计数问题/几何/单调性

12 Princess and Her Shadow(317E)

题目简述

A与B在平面直角坐标系上，每当A移动一步时（移动到相邻的一个没有障碍物位置），B 也朝这个位置移动一步，除非下一步的位置存在障碍物。找出一种A 的合法移动序列，使得最后A 与B处在同一个位置上；或者直接输出无解信息。

算法讨论

A与B不能到达或K=0 直接输出无解信息。

A与B只能在有限区域移动

按以下方法构造：每次A都往B道路上最短路上走。因为这样的行走路线，A 与B 之间的路径长度必定是不断地减小的，所以必定能在规定步数内达到同一个位置。

A与B可以到达无穷远的区域

若只有一个障碍物，我们人为地构造一个方案，使得其满足该性质。若有多个障碍物，我们可以借用只有一个障碍物的方法构造答案。

首先把A和B均移动到足够远的地方，再借用一个中心障碍物，并运用之前的方法，途中不碰到任意一个其余的障碍物，分类讨论，构造出方案。

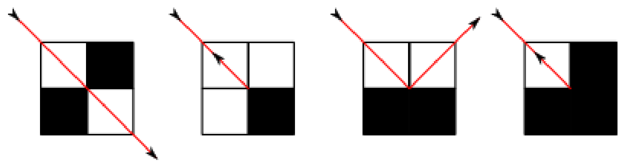
分类情况讨论较多，实现时应该注意细节实现。时间复杂度$O(N^2)$，空间复杂度$O(N^2)$。

TAG 分类讨论

13 Mirror Room(274E)

题目简述

给定一个N*M的网格图，上有若干个格子被障碍物占据。问一束由格点出发，与水平呈45 度的光经过反射无数次后，最多会经过多少个不同的格子。反射的规则如下(我们视网格的边缘全都被障碍物占据)：



算法讨论

光的路线有两种形状：形成一个环或是一条路径，并且，经过的格子是不会重复的。

我们模拟光的行进路线：在光的方向上找到第一个被占据的格子，分别判断格子的相邻两格是否被格子占据，以确定反射的方式。若在步进的过程中发现当前的状态与之前发生重复，则说明形成了一个环，它们之间的长度即为答案。否则把光向两边行进的长度相加即为答案。

我们可以用平衡树完成对格子的查找。时间复杂度为$O(NLogN)$，空间复杂度为$O(N)$。

TAG 数据结构/分类讨论

14 Dividing Kingdom(260E)

题目简述

平面上有N个点，要求分成3*3的井字形，每个井字内的点数排序后和给出的序列相等。

算法讨论

用DFS搜索每个位置放多少个点，这样可以计算出井字形4条分割线的位置。接下来用离线算法判断左上，右上，左下，右下四个位置的点数是否符合。不必判断全9个位置，因为根据方程可以得到判断4个位置必定得到整个图。总时间复杂度为$O((9!+N)LogN)$，空间复杂度为$O(N+9!)$。

TAG 搜索/数据结构

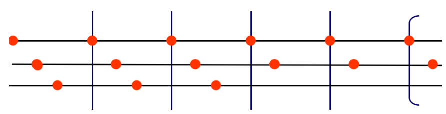
15 Doodle Jump(346E)

题目简述

给出a, n, p, h并定义$x_i = (i * a)\text{Mod } p$。给$x_0, x_1, \cdots, x_n$ 排序后，判断相邻两个数的最大值是否大于h。

算法讨论

首先用数轴表示x(图中相邻两个点的距离为a)，发现其实有许多长度为a 的重复小节，并且可以证明的是，最右的不完整的一小节是不会对答案产生影响：



单独考虑一小节，就可以缩小数据范围：此时的$p' = a, a' = a - p \text{ Mod } a$。$n$的值则要分类：若所有小节都是一样的，则$n' = \lfloor \frac{na}{p} \rfloor - 1$；如果一些小节有缺失（由于最后一行不完整导致的），则$n' = \lfloor \frac{na}{p} \rfloor - 2$。

但转化后的问题有一个不同的地方：在p点处是有一个点的。但可以证明的是，若$na >= p$，p点是对答案没有影响的，从而特判可以解决这点小问题。

但由于a与p的规模减小的速度不确定，时间复杂度还不能保证。每次在a 减小的时候，可以令$a' = min(p \text{ Mod } a, (a - p \text{ Mod } a))$，因为两者本质上没有区别。时间复杂度为$O(LogN)$，空间复杂度$O(1)$。

TAG 数学

16 Greedy Elevator(257E)

题目简述

给出电梯的运行方式，问每个人下电梯的时间。

算法讨论

直接按照题目描述的去贪心去做。用平衡树维护人的信息，用树状数组维护电梯上下的信息。总时间复杂度为$O(NLogN)$，空间复杂度为$O(N)$。

TAG 数据结构/贪心

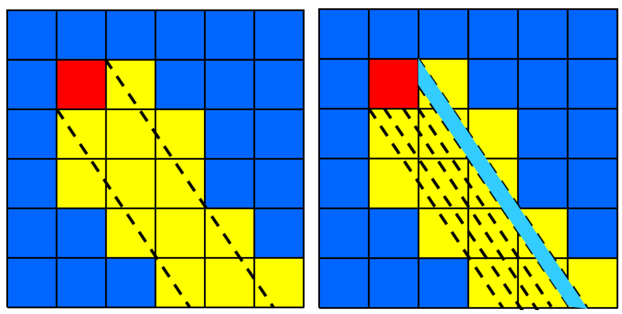
17 Cubes(243D)

题目简述

给定一个大小为$N * M$、单位长度为1的网格图，每个网格上都会摆放若干个1*1*1 的小立方体，构成一个三维空间。问从无穷远的地方，将一束方向为(Vx, Vy)的平行光投射到立方体上，会有多少个立方体被照亮（一个立方体只要有一部分被光投射到就算被照亮）。

算法讨论

首先将抽象的问题数学化：从上往下看，一个方格在$-V$ 方向上，所经过的立方体都会对这个格子有影响。(图中黄色格子的高度对红色格子的视线有影响)



那究竟这些格子会对红色格子的高度有什么影响呢？按照方向V，我们划分成多个条形区域（其中每个区域内部都不包含格点）。显然的是，设一个条形区域中经过的格子高度最大值为h，那么从这一个方向上看能看到高于h的立方体。若红色格子占据的多个条形区域中的最小值为H，那么红色格子高度大于H的立方体都会被照亮。

那么思路就十分显然了：按照V划分出条形区域，按照离光照的距离划分顺序，顺次处理。可以用线段树来维护条形区域的最小值，总时间复杂度为$O(N^2LogN)$，空间复杂度为$O(N^2)$。

TAG 数据结构/转化

18 Tournament-graph(323B)

题目简述

给一个无向完全图定向，使得两个点之间的距离不大于2。

算法讨论

假设我们现在有一个满足条件的无向图，我们可以添加两个点于其中（设为A, B，则每个点向A连一条边(除B点)，B向每个点连一条边）。暴力构造出点数为3，6的答案，逐个添加即可。

TAG 构造

19 Number Challenge(235E)

题目简述

给出a, b, c，定义$d(i)$为i的因子的个数。求

$$\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} d(i \cdot j \cdot k)$$

算法讨论

由于d为积性函数，所以不同因子对答案的影响是独立的。我们设$f[a][b][c][p]$为只使用不大于p的质数时的答案值。我们枚举三个位置取的分别是p^i, p^j, p^k，则可以由$f[a/p^i][b/p^j][c/p^k] * (i+j+k+1)$转移得到。实际转移时使用记忆化搜索，并且枚举质数时应从大到小，因为这样可以避免许多冗余状态出现。期望时间复杂度为$O(N^4)$，空间复杂度为为$O(N^4)$，但两者远远达不到这个上限。

TAG 动态规划/优化动态规划/转移顺序

20 Graph Game(235D)

题目简述

有一个N个顶点N条边的无向连通图，定义过程Apply(G)，其中G是一个连通块：a)给答案加上G的大小，b)随机删去G 中的一个顶点，c)对G中剩余的几个连通块进行过程Apply。问对整个图进行Apply后，答案的期望大小。

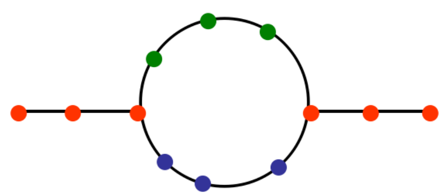
算法讨论

先考虑树的情况：定义事件(A,B)为在删除点A 时，B 点尚与A连通的几率。显然一个事件发生就会给答案加上1，所以答案的期望值为所有事件发生的总和。下面我们来证明(A,B)=在AB路径上的点数的倒数。

设AB路径上的点数为n。下面有两种情况：

- 连通块G的大小为n。这是G为一条链，只有选取A时才会发生事件(A,B)，几率为$1/n$。
- G的大小大于n。选取的点有两种情况：选取在AB链上，这时的几率为$1/x$。选取不在链上，几率为$\frac{x-n}{x} * \frac{1}{n}$，而两者相加得$1/n$，于是得证。

下面来考虑原题中的情况：若AB之间没有经过环，那么结论是一样的；若AB之间经过了环，我们把路径上的点分成三部分：



在选取A之前，不能选取橙色的点，把它们的数量记作X；环上的点被分为两半，数量分别为Y与Z；因为在选取A之前，A 到B之间必须存在通路，所以XY与XZ之间必须要有一条存在。由此可得期望值为$1/(X+Y)+1/(X+Z)-1/(X+Y+Z)$，其中最后减去重复计算的值。

总时间复杂度为$O(N^2)$，空间复杂度为$O(N)$。

TAG 数学推导

21 Two Permutations(323C)

题目简述

给定两个1到N的排列，以及M个询问(L,R,P,Q)，询问有多少个数，在第一个排列L,R 之间出现，又在第二个排列P,Q 之间出现，数据强制要求在线。

算法讨论

为了方便计算，我们把第一个排列变换为1到N，再把第二个排列等价的变化。这样询问就变为：在第二个排列P,Q之间，有多少个数在L,R之间，可以轻松用可持久化线段树解决。时间复杂度为$O(NLogN)$，空间复杂度为$O(NLogN)$。

TAG 数据结构/问题转化

22 Candies Game(341E)

题目简述

现有N个盒子，每个盒子内有$A_i(1 \le i \le N)$颗糖果。我们可以把盒子i的A_j的糖果倾倒到盒子j里（其中$A_j \le A_i$）。请给出一种倾倒方案，使得最后只有两个盒子内存在糖果。

算法讨论

由于题目中最后要求两个盒子内有糖果，所以猜想任意三个盒子都可以通过一系列倾倒使一个盒子为空。如果猜想成立，只需要不断使用这种方法达到目的即可。

下面来证明这个猜想是成立的。设三个盒子的糖果数量分别是$A,B,C(A \le B \le C)$，如果我们每次能达到一个状态$A',B',C'(A' < A)$，使三者的最小值严格递减，通过几次就能使$A=0$ 从而达到目的。

为了达到目的，我们不断B与C中取，增加A的数量，使得最终B或C剩余。若往A中倾倒一次，数量则会变为$2A$；倾倒K次后数量会变为$A*2^K$。设$B=A*Q+R$，我们的目的是让B剩余R，因为$R<A$满足要求。怎样才能在B 中取出恰好Q倍的A呢？注意到我们一次只能从A中取出$A*2^K$，所以我们观察Q的二进制：从低到高，若这一位是1，则从B中取；否则从C中取。显然这样取是合法的(C中不会出现不够取的情况)，并且可以从B中恰好取出Q倍的A(由二进制可以得出)。递归调用这个过程即可。这样时间复杂度为$O(N*A_i)$，空间复杂度为$O(N)$。

TAG 构造/猜想

23 Dima and Game(273E)

题目简述

两人进行博弈游戏。有N对数(L_i, R_i)，满足（$1 \le L_i < R_i \le P$）；两人顺次从中选取一对数，满足（$R_i - Li > 2$），并把这对数替换成（$L_i + \lfloor\frac{R_i-Li}{3}\rfloor, L_i + 2\lfloor\frac{R_i-Li}{3}\rfloor$）或（$L_i, R_i - \lfloor\frac{R_i-Li}{3}\rfloor$）。给出$N, P$，问在两人最优决策下，存在多少种$N$ 对数的方案，使得先手获得胜利。注意到这N对数是有序的。

算法讨论

博弈游戏直观的思路就是计算SG函数，又因为本题中决策只有两种，所以SG 值只有0，1，2三种。但由于本题中P最大值达到10^9，不能简单的线性递推；又打表发现SG值是成段变化的，所以通过成段转移可以减少递推的时间。

推出SG后，余下的问题就很容易解决了：设$F[i][j]$为已确定前i对数，SG值异或后为j的方案数，简单的转移即可。总时间复杂度为$O(P+N)$，空间复杂度为$O(P+N)$。

TAG 博弈/优化/动态规划

24 Friends(241B)

题目简述

N个数$A_i(1 \le i \le N)$，两两异或后，从小到大排序，问前M个数的总和为多少？

算法讨论

对这种问题一般使用二分找到排名第M的数为多少。我们可以将N 个数按二进制为塞进一棵TRIE里，维护大小即可在$O(NLog^2N)$里二分出答案，若直接在树上二分，时间可以减少到$O(NLogN)$。对于求和操作，与前面的操作类似。

总时间复杂度为$O(NLog^2N)$，空间复杂度为$O(NLogN)$。

TAG 二进制/TIRE

25 Reclamation(325D)

题目简述

给出一个$N*M$的网格图，把它卷成圆柱形后，它的第一列与第M 列相邻。每次把其中一个格子删去，若圆柱形的上方与下方仍然连通（四连通），则答案加一，否则保留这个格子。问最后答案为多少。

算法讨论

上下是否连通可以转化为网格图中是否存在一条八连通的路径环绕整个圆柱体。考虑新删除一个格子，只需判断是否存在一条到自身的回路。在实现方面可能有所不便，所以我们把网格复制一遍，成为大小为$N*2M$的图，那样只需要判断在两幅图对应的位置上是否连通，使用并查集可以在时间复杂度$O(NM)$，空间复杂度为$O(NM)$。

TAG 并查集/模型转换

26 Levko and Game(360E)

题目简述

给出一个N个点$M+K$条边的有向带权图，其中K条边的权值是可以改变，范围由L_i到R_i。给出三个点A, B, C，问通过改变权值能否让A到C的最短路径长度小于由B到C的路径，并给出边权的改变方案。

算法讨论

若只考虑可以改变的边权，只有四种可能性：A通过、B通过、AB均通过、AB均不通过。后两种情况，边权是不会对其有影响的；所以我们简单的考虑前两种情况即可。首先把每条边权置为其上限，跑一次最短路。若对于一条边(u, v)，由A到u的最短路径小于B到其的距离，那么就把边权改为其下限。因为这样做不会修改第二种边，所以贪心的修改必定得到最优解。若采用DIJKSTRA 实现，总时间复杂度为$O(KMLogM)$，空间复杂度为$O(N+M+K)$。

TAG 贪心/最短路

27 Endless Matrix(249E)

题目简述

存在一个类似蛇形排列的二维表，问一个矩形区间内的和。

1	2	5	10	17	26
4	3	6	11	18	27
9	8	7	12	19	28
16	15	14	13	20	29
25	24	23	22	21	30
36	35	34	33	32	31

算法讨论

通过二维前缀和，问题可以转化成求从(1,1)到(R,C) 矩形的和。那么根据R与C的大小，我们均可以把整个大矩形划分成两个：一个正方形（内部为1到$min(R,C)^2$的总和），一个长方形（包含若干个等差数列）。

1	2	5	10	17	26
4	3	6	11	18	27
9	8	7	12	19	28
16	15	14	13	20	29
25	24	23	22	21	30
36	35	34	33	32	31

两部分都可以轻松计算出来。还有一个小问题是如何判断答案是否大于10^{10}。我在程序中使用的是高精度；也可以分别模两个大于10^{10}的质数，如果相同则认为答案是小于10^{10}的。时间复杂度为$O(1)$，空间复杂度为$O(1)$。

TAG 数学

28 Biologist(311E)

题目简述

存在标号为$1..N$的N只狗，每一只有其性别S与让其改变性别的代价V。有M个额外任务，每个任务要求若干只狗的性别为同一种S，完成任务有回报W，否则会被扣除G。求最大的收益。

算法讨论

本题存在MALE与FEMALE的关系，于是联想到网络流。本题的构图是比较简单的（由于每个任务都要求同一种性别），不再赘述。时间复杂度$O((M+N)Log(M+N))$，空间复杂度$O(M+N)$。

TAG 网络流

29 Polo the Penguin and Lucky Numbers(288E)

题目简述

一个只含4与7的数字被称作幸运数。给出两个长度相等的幸运数L, R，把L到R中的幸运数进行排序得到一个序列a_i，计算$\sum_i a_i a_{i-1}$。

算法讨论

设数列F_i为所有的i位幸运数排序后相邻两两相乘得到的总和。考虑该如何进行递推：把这些数分为两部分，分别是$4*10^i+x$与$7*10^i+x$。若两个数都在前一部分，相乘的积为$16*10^{2i}+4*10^i+(x+y)+xy$。第一部分可以直接计算，第二部分可以记录一个辅助数列G_i计算所有的i位幸运数排序后相邻两两相加得到的总和，第三部分就是F_{i-1}。若两个数都在后一部分，同理可得。否则两个数只能是$47777\cdots$与$74444\cdots$，加上就好了。G转移同理。

接下来只要运用F与G分别计算出$444\cdots$到L以及到R的答案，相减即可，计算与上述过程类似。总时间复杂度为$O(N)$，空间复杂度为$O(N)$。

TAG 递推/分类讨论

30 Binary Key(322E)

题目简述

一个长度为N的字符串S会与一个长度为K的钥匙串A生成了一个长度M的生成串T：A串中只包含0或1，从小到大枚举位置pos= $0\ldots N-1$，若A[pos Mod K]=1，则把S[pos]加入字符串T中。给定字符串S、生成串T以及长度K，求一个字典序最小的钥匙串。

算法讨论

枚举钥匙串中有多少个1，那我们就能通过简单的贪心得到最小的字典序。

TAG 贪心

31 Ciel and Flipboard(321D)

题目简述

一个大小$N*N$的带权网格图($N=2X-1$)，每次可以翻转大小为$X*X$的子矩阵，使矩阵内的每一个值均变为其相反数。问矩阵可以达到的最大值。

算法讨论

记矩阵$F[i][j](1\le i,j\le N)$为经过若干次翻转之后矩阵达到的状态：$F[i][j]=0$代表没有被翻转，$F[i][j]=1$反之。对于合法的操作，总有性质$F[i][j]\wedge F[i][x]\wedge F[i][j+x]=0$、$F[i][j]\wedge F[x][j]\wedge F[i+x][j]=0$（因为三者中只会被翻转0个或2个）。

有了这条性质后，我们可以枚举$F[x][i](1\le i\le x)$，这样第X行的状态便可确定了。余下的行，根据性质，除了对称的行有关联之外，其余是两两独立的。于是我们可以贪心的枚举每一个元素的F值，得到最优的值。总时间复杂度$O(2^XN^2)$，空间复杂度$O(N^2)$。

TAG 转化/枚举

32 Maxim and Calculator(261E)

题目简述

存在两个数a, b，能进行两种操作：$b = b + 1$或者$a = a * b$。问从初始状态$a = 1, b = 0$经过至多p次操作后a 能达到哪些值($l \leq a \leq r$)，求出其个数？

算法讨论

题目给出的数据范围规定$p \leq 100$，这提示我们a里面不存在大于100的因子。而在小于10^9里的数里面只有$3 * 10^6 = T$个数最大的质因子小于100。

预先筛出这些数，再用动态规划进行求解。具体的，设F[i][j]为目前$a = i, b = j$所需最小的操作数是多少。$F[i][j] = min(F[i/j][j] + 1, F[i][j-1])$。总时间复杂度为O(pr)，空间复杂度为O()

TAG 枚举/动态规划

33 Ksusha and Square(293D)

题目简述

二维平面上存在一个凸多边形，并从凸多边形内部或边上随机取两点（坐标均为整数），求以其连线为对角线构成的正方形的期望面积。

算法讨论

若两个点的坐标为(a, b)与(c, d)，那么正方形的面积为$((a-c)^2+(b-d)^2)/2$，这提醒我们可以把横纵坐标分开考虑，因为他们的互相独立的。

首先考虑X坐标对答案的贡献。在平面上划出若干条垂直与X轴的直线，其中每一条直线上都有若干个点在凸多边形内。从左到右，记每一条直线上的点数为$a_1, a_2, a_3, \cdots, a_K$。因为计算的X坐标的贡献，若考虑直线$x_1$与直线$x_2$，贡献即为$a_{x1} * a_{x2} * (x_1 - x_2)^2$（直线上两点之间贡献都是$(x_1 - x_2)^2$）。记$F_i$为两点的X坐标的较大点在直线$i$上的贡献值，直观地，$F_i = a_i \sum_{j=1}^{i-1} a_j (i-j)^2$。又令$F_i = a_i \sum_{j=1}^{i-1} a_j (i-j)^2 = a_i G_i$，相减得：

$$\begin{aligned}
G_i - G_{i-1} &= a_{i-1} + \sum_{j=1}^{i-2} a_j [(i-j)^2 - (i-j-1)^2] \\
&= a_{i-1} + \sum_{j=1}^{i-2} a_j (2i - 2j - 1) \\
&= a_{i-1} + (2i-1) \sum_{j=1}^{i-2} a_j - 2 \sum_{j=1}^{i-2} a_j * j \\
&= a_{i-1} + (2i-1) S_{i-2} - 2S'_{i-2}
\end{aligned}$$

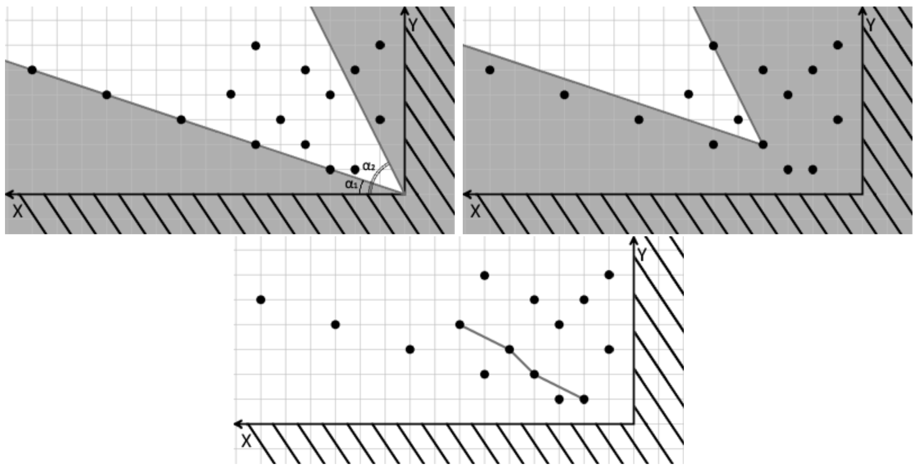
其中S_i与S'_i根据上述定义求出，G_i与F_i均能在线性时间内推出。本题的时间复杂度为$O(N+M)$，空间复杂度为$O(N+M)$。

TAG 递推/几何

34 Donkey and Stars(249D)

题目简述

一个二维平面上有N个点，以及两个方向α_1与α_2，要求从原点出发，找一条最长的链，每次一个点能到达方向严格夹在α_1与α_2内的点。



算法讨论

经过坐标变换后，变成经典的动态规划问题。按一维排序，并用树状数组维护另一维的信息，总时间复杂度为$O(NLogN)$，空间复杂度为$O(N)$。

TAG 动态规划/转化

35 Liars and Serge(256D)

题目简述

已知N个人中有两种人：诚实的和不诚实的。你现在问他们一个问题：你们中有多少人是诚实的？诚实者会回答正确的答案，而不诚实者会在1到N里面随机挑一个不正确的数回答。将N个人的答案有序的排成一个序列A，那么有多少个序列，你能从中明显地发现有K个人在撒谎呢？

算法讨论

如果回答X的人数不为X，那么他们必定是在说谎，于是得到一个状态转移方程：设状态$F[i][j][k]$为只考虑答案小于i的回答，共有j个，其中k个是明显说谎的。枚举回答i的人数，分两种情况考虑。总时间复杂度为$O(N^4)$，空间复杂度为$O(N^3)$。由于N最大为256，这部分打表就可以通过。

TAG 动态规划/打表

36 Shaass and Painter Robot(294D)

题目简述

在一个$N * M$的格子图内，从格子(S, T)引一条与X轴成45或135度的有向射线，遵循镜面反射原则，并且会把所经过的格子染黑。问使整张图黑白染色是否可行，若可行则输出经过多少个格子（重复计算），否则输出-1。

算法讨论

显然如果所有边界上的格子都被经过，那么内部所有格子都被经过。模拟射线的路程，直到在边界上染黑的格子为$N + M - 2$，或者是输出无解过程，时间复杂度为$O(N + M)$，空间复杂度为$O(N + M)$。

TAG 模拟

37 Levko and Sets(360D)

题目简述

给出长度为N的数组A_i、长度为M的数组B_j、质数P。令

$$X = ({A_i}^{\sum_{j=1}^{M} Bj*Cj})ModP$$

其中有$C_j >= 0$，问有多少个不同的X？

算法讨论

因为P是质数，那么必定存在一个原根G，使得$G^{Ri}s = A_i$。那么对于一个确定的i所得到的X，总会有$X = (G^{Ri*\sum_{j=1}^{M} Bj*Cj})ModP$；在有费马小定理可得，$X = (G^{(Ri*K*T)Mod(P-1)})ModP$，其中$K = Gcd(B_j), T >= 0$。那么我们可以推出一个确定的$i$可以得到的$X$的形式总是$G^{R*T}$，其中$R$是常数，且为$P - 1$的约数。我们通过枚举$T$（$P - 1$的约数）可以找出其循环节。

下面问题转变为，对于一个数$P-1$，有若干个数R_i，求在0到$P-1$中，R_i的倍数的并集。由于这里保证R_i是$P - 1$的约数，所以得到下面这个算法：给每个数X分类，找出最大的Y同时是$P - 1$和X的约数，则X归在Y一类。枚举Y，若R中存在一个为Y的约数，那么属于Y一类的数则累加入答案里。这部分的计算可以用欧拉函数加速。综合两部分计算，总时间复杂度为$O(MLogP + NTLogP)$，空间复杂杂都为$O(T)$，其中T为$P - 1$的约数个数。

TAG 数论/欧拉函数/容斥原理

38 k-Maximum Subsequence Sum(280D)

题目简述

给出长度为N的数组A_i和M个修改或询问：修改一个位置的值，或是查询一段区间内的$K-MSS$。从一段区间内选出不多于K个小区间，被这些区间的包含的所有数的总和的最大值就是这段区间的$K-MSS$。

算法讨论

由于题目给出的K不大于20，所以用一个线段树维护整个序列，每一个区间记录其$X-MSS$，总时间复杂度为$O(K^2NLogN)$。遗憾的是这不能通过所有的数据。

出题人给出的方法是，找出区间内的$1-MSS$，将其总和累加进答案，再将其取反，重复K次。直观上，使用最小费用最大流的算法可以解释其正确性，这样优化后，时间复杂度可以优化到$O(KNLogN)$。

TAG 数据结构/转化

39 Two Sets(251D)

题目简述

把N个自然数划分成两个集合，记两个集合内所有值的异或和分别为X与Y。给出一种可行的划分方案，在满足$X+Y$最大的情况下，最小化X。

算法讨论

只满足第一个条件，单独考虑第i位：如果在N个数中第i位为1的数有P个，若$P=0$，第i位不能为$X+Y$做出贡献；若P为奇数，那么不管怎么划分，总会贡献2^i; 如果为大于0的偶数，则有可能为答案贡献2^{i+1}或0。从高到低位贪心的枚举，这样做一定是最优的。

记A_i为i是否属于第一个集合，如果P为偶数且给答案贡献了2^{i+1}，那么会得到形如$A_i \mathrm{xor} A_j \mathrm{xor} A_k$=1的异或方程组。贪心的时候，只用考虑新加入的方程与前面的方程是否有矛盾。注意到我们不必每一次都进行一次高斯消元，只用处理新加入的方程即可。

满足第二个条件，像前面一样贪心的枚举就行了。总时间复杂度为$O(NL^2)$，空间复杂度为$O(N)$。另外，使用Bitset可以优化本题的时间复杂度。

TAG 贪心/高斯消元

40 Ladies' Shop(286E)

题目简述

给出均在M以内的N个数A_i。要求找出不大于M若干个数，这些数的无穷背包在M以内恰好等于A_i。要求选出的数数量最少。

算法讨论

只有A中的数可以被选。如果一个数可以被若干个数相加而得，那么选择它是冗余的。进一步，如果A中存在一个数等于其中两个数相加，它是冗余的。我们可用FFT来优化这一过程，时间复杂度为$O(MLogM)$，空间复杂度为$O(M)$。

TAG FFT

41 Close Vertices(293E)

题目简述

一个棵树上有边权，求有所少个二元对$(u,v)(u < v)$，使得从节点u到节点v上经过的边数不大于L，边权和不大于W。

算法讨论

对整棵树进行点分治，找出此时树的重心，找出经过重心的路径。具体的做法是先把所有的点相互处理，再把同一颗子树内相互处理的值减去即可。相互处理时先按一维排序，再在另一维上建立数据结构维护。时间复杂度为$O(NLogN)$，空间复杂度为$O(N)$。

TAG 分治/数据结构

42 White, Black and White Again(306C)

题目简述

把A个白球和B个黑球放在N个盒子里。要求：每个盒子里至少有一个球且颜色全部相同，记盒子的颜色为球的颜色，则盒子的颜色必须为连续的一段白色+连续的一段黑色+连续的一段白色，恰好三段。盒子之间是有序的，每个盒子内球的放置也是有序的。问有多少种放置方法。

算法讨论

由于球是有序的，我们最后把答案乘上$A! * B!$，并讨论球无序的情况。枚举黑色盒子的长度，那么白色盒子的长度也确定了，并化为两个相似的子问题：在X个盒子里放Y个球的方案数$(X \le Y)$。先在每个盒子里放一个球，剩余$Y - X$个；再自由放置，总方案数为$C(X-1, Y-X+X-1) = C(X-1, Y-1)$。时间复杂度为$O(N^2)$，空间复杂度为$O(N^2)$。

TAG 递推/数学/组合数学

43 Have You Ever Heard About the Word?(319D)

题目简述

一个长度为N的字符串，不断进行下列操作直至不可行：找出其最短的重复子串，如果有多个，找出其最左的，删除其中之一。求最后的字符串。

算法讨论

暴力模拟其删除的过程。从小到大枚举子串的长度L，判断目前的字符串是否存在这样的重复子串，若存在则暴力删除。用哈希判断字符串相同可以减少时间。判断目前的字符串是否存在这样的重复子串的具体方法是，在字符串中每隔L个建立一个标志点，枚举相邻两个标志点，分别向左向右扩展，用最大长度和L比较。时间复杂度为$O(N^2)$，但实际上是达不到的，空间复杂度为$O(N)$。

TAG 字符串/哈希

44 Piglet's Birthday(248E)

题目简述

现有N个架子，每个架子初始都摆放着若干个满的罐子。接下来进行若干次操作，每次从一个架子上取下K 个罐子，把其中满的的罐子掏空，再把放到另一个架子上。问每次操作后，期望有多少个架子上都是空罐子。

算法讨论

题目给定的每个架子上初始不超过100个罐子，而操作不会让满的罐子的数量增加，记$F[i][j]$当前架子i 上有j 个满的罐子的概率。每次操作的时候枚举j，并设$dp[x][y]$ 为选取了x 个罐子，其中y个罐子是满的概率，进行转移。总时间复杂度为$O(NMK^2)$，空间复杂度为$O(NM)$。

考虑到一次操作可以拆分成K次操作，每次移动一个罐子。这样总时间复杂度为$O(NMK)$。

TAG 动态规划/概率

45 Rhombus(263E)

题目简述

在$N*M$的网格图中每个格子中给定权值，同时给出一个数K，定义两个网格的距离为K减去他们的曼哈顿距离，若曼哈顿距离大于K则距离为0。定义一个位置的权值为所有网格的权值乘以到它的距离之和，要求找出拥有最大的权值的位置。

算法讨论

把一个位置分成左上、左下、右上、右下四个位置。通过记录若干个前缀和，面积和可以推出其中之一的权值，其余的位置相似递推得到。总而言之是需要什么设什么。总时间复杂度为$O(N^2)$，空间复杂度为$O(N^2)$。

TAG 递推

46 Cow Tennis Tournament(283E)

题目简述

给出N人，规定编号大的人能打败编号小的人。进行M次修改，一次形如(L,R)的修改可以改变编号在L 到R之间相互的胜负关系（由胜变负，由负变胜）。问最后存在多少组编号(I,J,K)，使得I胜J，J胜K，K胜I。

算法讨论

补集思想，用$C(N,3)$减去不满足的组数。不满足的必定为I 胜J，I胜K，所以如果编号为I胜过X个人，则会产生$C(X,2)$组不满足。

想象一个$N*N$的矩阵，对角线上方为0，下方为1。每个修改会修改一个矩形的区域，而一次询问只会查询一段里面存在多少个1。使用扫描线算法，总时间复杂度为$O(NLogN)$，空间复杂度$O(N)$。

TAG 补集/数据结构

47 Rectangles and Square(335D)

题目简述

平面上存在N个矩形，两两之间没有交集。问是否存在一个正方形，能恰好覆盖若干个矩形（要求这些矩形的面积之和等于正方形的面积，正方形中所有位置都被矩形覆盖）。

算法讨论

题目保证了矩形的坐标为整数并且均在0到3000之间。枚举一个矩形，作为方形的左上角，同时枚举方形的边长。如果左边和上边或中间出现“缺口”，则不必再枚举下去；否则判断右边与下边是否完整。这个用二维数组来维护就行了。总时间复杂度为$O(N^2)$，空间复杂度为$O(N^2)$。

TAG 枚举/递推

48 The Great Julya Calendar(331C3)

题目简述

给出一个数，减去一个0-9内的数，像35能减去3或5，912能减去9或1或2。问最多连续减多少次能达到0。

算法讨论

显然连续的贪心是可行的，每次都减去最大的一个数。接下来比较巧妙的DP。设$F[x][y](0 \le y \le 9)$ 是一个二元组(P, Q)，P为为把x减少到0 最少需要的次数，Q等于会额外减去多少。而y的作用是每一次减去的至少为y。当计算$F[x][y]$ 时，设$x = A * 10^P + B$，先计算$F[B][Max(y, A)] = (P, Q)$，新的x为$x - 10^P - Q$，再进行计算。用记忆化搜索减少重复状态的搜索。

TAG 动态规划

49 Positions in Permutations(285E)

题目简述

对于一个排列P，位置i上的数P_i满足$|P_i - i| = 1$，这个位置是OKAY 的。求有多少个长度为N的排列恰好存在K个位置是OKAY的。

算法讨论

设$F[i][j]$为已经确定前i位置上的数，其中有j个数已经确定，其余的数并没有确定的方案数。这样做的好处是转移有阶段性。由于如果i这个位置是OKAY的，如果选择$i+1$，可能会影响后面的选择。所以附加一个状态s，代表的是这个位置与后一个位置是否被放置。令$G[i] = F[n][i] * (n-i)!$，那么$G[i]$就是OKAY的位置不小于i的排列数。G之间相互之间进行转移可以求出恰好K 个排列数。时间复杂度为$O(N^2)$，空间复杂度为$O(N^2)$。

TAG 动态规划

50 Colorful Stones(264D)

题目简述

两个序列$S[1..N]$与$T[1..M]$，其中都由0，1，2组成，一开始初始状态有(1,1)，对于一个状态$(i,j)(i < N, j < M)$ 和一个数k，可以推到状态$(i + S[i] = k, j + T[j] = k)$。问有多少可达的状态。

算法讨论

如果(i,j)是可达的，那么$S[1..i]$不是$T[1..j+1]$的子序列，$T[1..j]$不是$S[1..i+1]$的子序列，同时$T[1..j]$和$S[1..i]$不是$...XY$与$...YX$的形式。枚举i，同时维护j的范围$L..R$。时间复杂度与空间复杂度均为$O(N)$。

TAG 枚举/证明