Codeforces题目泛做 解题报告

南京外国语学校　许昊然

Contents

1 Solution **2**

1.1 Codeforces 7E Defining Macros . . . 2
1.2 Codeforces 8D Two Friends . . . 3
1.3 Codeforces 8E Beads . . . 4
1.4 Codeforces 10E Greedy Change . . . 4
1.5 Codeforces 15E triangles . . . 5
1.6 Codeforces 17C Balance . . . 7
1.7 Codeforces 17E Palisection . . . 7
1.8 Codeforces 19E Fiary . . . 8
1.9 Codeforces 23D Tetragon . . . 9
1.10 Codeforces 23E Tree . . . 10
1.11 Codeforces 26E Multithreading . . . 11
1.12 Codeforces 28D Do not fear, DravDe is kind . . . 12
1.13 Codeforces 30D Kings Problem . . . 12
1.14 Codeforces 30E Tricky and Clever Password . . . 13
1.15 Codeforces 32E Hide-and-Seek . . . 14
1.16 Codeforces 35E Parade . . . 15
1.17 Codeforces 36E Two paths . . . 16
1.18 Codeforces 37E Trial for Chief . . . 16
1.19 Codeforces 39C Moon Craters . . . 17
1.20 Codeforces 39A C*++ Calculations . . . 18
1.21 Codeforces 39E What Has Dirichlet Got to Do with That? . . . 18
1.22 Codeforces 40E Number Table . . . 19
1.23 Codeforces 43E Race . . . 20
1.24 Codeforces 44J Triminoes . . . 21
1.25 Codeforces 45G Prime Problem . . . 21
1.26 Codeforces 45E Director . . . 22
1.27 Codeforces 46F Hercule Poirot Problem . . . 23
1.28 Codeforces 47E Cannon . . . 24
1.29 Codeforces 49E Common ancestor . . . 24
1.30 Codeforces 51F Caterpillar . . . 25
1.31 Codeforces 53E Dead Ends . . . 26

1.32 Codeforces 57D Journey . . . 27
1.33 Codeforces 226E Noble Knight's Path . . . 28
1.34 Codeforces 217D Bitonix' Patrol . . . 28
1.35 Codeforces 67E Save the City! . . . 29
1.36 Codeforces 67C Sequence of Balls . . . 30
1.37 Codeforces 70D Professors task . . . 31
1.38 Codeforces 70E Information Reform . . . 31
1.39 Codeforces 156E Mrs. Hudson's Pancakes . . . 32
1.40 Codeforces 105D Entertaining Geodetics . . . 33
1.41 Codeforces 193D Two Segments . . . 34
1.42 Codeforces 75E Ship's Shortest Path . . . 35
1.43 Codeforces 76F Tourist . . . 36
1.44 Codeforces 76A Gift . . . 37
1.45 Codeforces 77E Martian Food . . . 38
1.46 Codeforces 79D Password . . . 39
1.47 Codeforces 81E Pairs . . . 40
1.48 Codeforces 82E Corridor . . . 41
1.49 Codeforces 83E Two Subsequences . . . 42
1.50 Codeforces 85E Guard Towers . . . 43
1.51 Codeforces 86E Long sequence . . . 44
1.52 Codeforces 89D Space Mines . . . 45
1.53 Codeforces 91D Grocer's Problem . . . 45
1.54 Codeforces 93D Flags . . . 46
1.55 Codeforces 97C Winning Strategy . . . 47
1.56 Codeforces 97A Domino . . . 47
1.57 Codeforces 98D Help Monks . . . 48
1.58 Codeforces 98C Help Greg the Dwarf . . . 49
1.59 Codeforces 191D Metro Scheme . . . 50
1.60 Codeforces 164D Minimum Diameter . . . 51
1.61 Codeforces 150E Freezing with Style . . . 52
1.62 Codeforces 101E Candies and Stones . . . 53
1.63 Codeforces 103E Buying Sets . . . 53
1.64 Codeforces 105E Lift and Throw . . . 54
1.65 Codeforces 107D Crime Management . . . 55
1.66 Codeforces 113D Museum . . . 56
1.67 Codeforces 115D Unambiguous Arithmetic Expression . . . 57
1.68 Codeforces 120I Luck is in Numbers . . . 58
1.69 Codeforces 123E Maze . . . 59
1.70 Codeforces 125E MST Company . . . 60
1.71 Codeforces 193E Fibonacci Number . . . 61
1.72 Codeforces 145D Lucky Pair . . . 61

1.73 Codeforces 132E Bits of merry old England . . . 62
1.74 Codeforces 138D World of Darkraft . . . 63
1.75 Codeforces 140F New Year Snowflake . . . 64
1.76 Codeforces 147B Smile House . . . 65
1.77 Codeforces 152D Frames . . . 66
1.78 Codeforces 183D T-shirt . . . 66
1.79 Codeforces 217E Alien DNA . . . 67
1.80 Codeforces 135E Weak Subsequence . . . 68
1.81 Codeforces 163D Large Refrigerator . . . 69
1.82 Codeforces 167E Wizards and Bets . . . 70
1.83 Codeforces 232D Fence . . . 70
1.84 Codeforces 175E Power Defence . . . 71
1.85 Codeforces 176D Hyper String . . . 72
1.86 Codeforces 178F Representative Sampling . . . 73
1.87 Codeforces 178E The Beaver's Problem II . . . 73
1.88 Codeforces 180B Divisibility Rules . . . 74
1.89 Codeforces 185D Visit of the Great . . . 75
1.90 Codeforces 187D BRT Contract . . . 76
1.91 Codeforces 176E Archaeology . . . 76
1.92 Codeforces 196D The Next Good String . . . 77
1.93 Codeforces 198E Gripping Story . . . 78
1.94 Codeforces 200E Tractor College . . . 79
1.95 Codeforces 200A Cinema . . . 80
1.96 Codeforces 201E Thoroughly Bureaucratic Organization . . . 81
1.97 Codeforces 201D Brand New Problem . . . 82
1.98 Codeforces 204E Little Elephant and Strings . . . 83
1.99 Codeforces 207B Military Trainings . . . 83
1.100 Codeforces 207A Beaver's Calculator . . . 84
1.101 Codeforces 167D Wizards and Roads . . . 85
1.102 Codeforces 209C Trails and Glades . . . 87
1.103 Codeforces 212B Polycarpus is Looking for Good Substrings . . . 88
1.104 Codeforces 212D Cutting a Fence . . . 88
1.105 Codeforces 212C Cowboys . . . 89
1.106 Codeforces 213E Two Permutations . . . 90
1.107 Codeforces 217C Formurosa . . . 91
1.108 Codeforces 229E Gifts . . . 92

2 Special Thanks **93**

1 Solution

1.1 Codeforces 7E Defining Macros

通过情况

AC

关键词

表达式计算 dp

题目大意

题目给了一系列C++的宏定义，问你一个表达式是否是“安全”的。安全的定义是，展开后的表达式中，所有的宏在计算过程中都被作为一个整体运算。

如#define sum x+y 后， $2 * sum$就会被替换乘$2 * x + y$，此时因为乘号优先级比加号高，导致sum宏在实际计算中被拆开了，可能产生错误。

宏的个数≤ 100, 每个表达式长度≤ 100. 只有四则运算和括号。

算法讨论

我们考虑一个宏是否是“安全”的，经过观察和一些实验，可以发现，只有以下4种状态：

- 状态1($s1$)： 这个宏完全安全，以任何方式使用该宏都没问题。
- 状态2($s2$)： 这个宏不安全，只要表达式中出现该宏，都会导致表达式不安全。
- 状态3($s3$)： 这个宏部分安全，仅当这个宏与'*','/'连接时，或出现在'-'后面时，才会使表达式不安全。
- 状态4($s4$)： 这个宏部分安全，仅当这个宏出现在'/'后面时，才会使表达式不安全。

有了这4个状态，我们只需推出状态之间的转移即可。

- 如果表达式没有使用任何运算符或括号或宏（也就是s仅仅是个单独的变量），那么安全级别显然是$s1$
- 如果表达式s是(t)的形式（被一对括号括起来的表达式t），那么如果t的状态不是$s2$，则s的状态是$s1$，否则s的状态是$s2$
- 我们找到表达式s中，最后一次运算的符号，设其为op，设其两侧表达式分别为$t1$和$t2$。我们进行以下分类讨论：
  - 显然，如果$t1$或$t2$的安全状态是$s2$，则s的状态也是$s2$;
  - 如果op是'+'，那么s的状态是$s3$;
  - 如果op是'-'，那么，如$t2$状态是$s3$，则s状态是$s2$，否则s状态是$s3$
  - 如果op是'*'，那么，如$t1$或$t2$状态是$s3$，则s状态是$s2$，否则s状态是$s4$
  - 如果op是'/'，那么，如$t1$或$t2$状态是$s3$，或$t2$状态是$s4$，则s状态是$s2$，否则s状态是$s4$

于是，此题得到了解决。 时间复杂度$O(n * len^2)$，如果愿意追求更好的复杂度，可以建出表达式树，从而做到$O(N * len)$

特别注意

此题在tsinsen上的版本有多组数据，因此要特判$n = 0$的情况。

1.2 Codeforces 8D Two Friends

通过情况

AC

关键词

计算几何 二分答案

题目大意

平面上有3个点：A，B，C. 现在Alice和Bob都在A，Alice想要走到B，走的路线长度不得超过LA，Bob想经过C然后到B， 走的路线长度不得超过LB。

要求设计他们的路线，使得从A开始的公共部分尽可能长（也就是一旦两人分开，即使重新会合也不计入公共部分的长度了）

算法讨论

首先，如果Alice可以先陪Bob直线走到点C，再陪Bob走到点A，那么Alice显然可以陪伴Bob全程，答案是$min(LA, LB)$.

进一步观察，易发现答案具有单调性。故考虑二分答案。

设我们即将验证答案S的可行性。 因为我们之前已经特判了Alice陪Bob一起去点C的情况，因此我们现在可以假设，走了公共的S路程后，Bob还没去过点C.

因为一旦分离，即使再次重合也不再计入公共路径长度，所以接下来我们可以分开单独考虑Alice和Bob，而且应该让他们尽快到点B（当然Bob还得经过商店）。 分离后，Alice的路线显然应该直线去点B。而Bob的路线显然是直线去点C，然后直线去点B。因此分离点必须满足：

- 分离点必然在以点A为中心，S为半径的圆内。（因为Alice和Bob一起走的长度为S）
- 分离点必然在以B为中心，$LA - S$为半径的圆内。（为了让Alice能在剩下的$LA - S$时间内到达点B）
- 分离点必然在以C为中心，$LB - S - dist(B, C)$的圆内。（为了让Bob能在剩下的$LB - S$时间内到达点C然后去点B）

如果存在任一分离点能同时满足上述3条要求，则S可行。

于是问题被转化为了纯计算几何问题：给定3个圆，判断其是否又公共部分。 这个问题十分经典，解决方法也五花八门，暴力找关键点+分类讨论、二分x轴等方法都是可行的。

特别注意

注意实数精度问题。

1.3 Codeforces 8E Beads

通过情况

AC

关键词

数位dp

题目大意

将所有n位二进制串（允许前导0）中同时满足字典序不小于其逆序串、取反串和逆序取反串的串提取出来，按字典序排序，求第m个。 $n \leq 50, k \leq 10^{16}$

算法讨论

首先显然满足题意的二进制串的首位必须是0.

考虑一位一位地确定答案串。假设已经确定了答案串的前k位，我们假设第$k+1$位是0，则要设法统计出满足条件的串的个数s。

那么如果$s < m$，则答案串第$k+1$位为1，同时$m = m - s$；否则答案串第$k+1$位为0.

于是问题转化为，统计所有长度为n的，前缀为$prefix$的二进制串中，满足题目要求的串的个数。

这是一类与数位有关的统计问题，于是很容易想到数位dp。

状态$dp[i][rev][inv]$表示，当前已经确定了前i位和末i位， rev表示前i位与末i位的逆序是否相等，inv表示前i位与末i位的逆序取反后是否相等。

状态转移比较显然，我们枚举第$i+1$位和第$n-i$位的取值，如果它满足$prefix$的限制，且新的串没有违反题目要求（可以利用rev,inv和取值判断）， 那么更新rev和inv的状态，并累加到对应的新状态上。

时间复杂度$O(16 * N^2)$

特别注意

注意如果n为奇数，那么dp到正中间一位的时候，这一位会同时作为前i位和末i位的组成部分，需要特判。

1.4 Codeforces 10E Greedy Change

通过情况

AC

关键词

论文题 结论题

题目大意

给定n种货币，每种货币数量无限。

现在要求以最少的货币数目表示一个数S。 一种方法当然是DP求一个最优解了， 当然正常人的做法是贪心：每次取最大的不超过当前待表示数的货币。

现在，你的任务是证明正常人的表示法不一定最优：找到最小的S，使得正常人的表示法比理论最优解差，或说明这样的S不存在。

$n \leq 300$

算法讨论

首先这是一道论文结论题。 有一篇论文专门讨论了这个问题（其实CF上的题目描述和论文里的几乎一个字都没变，连举的例子都是一样的）

这篇论文题目是《A Polynomial-time Algorithm for the Change-Making Problem》， 可以在这里下载

其实我们可以感性的猜想出一个“看起来很靠谱”的算法：

- 首先把所有货币从大到小排序。
- 考虑某个大于$w[i]$但小于$w[i-1]$的数的表示方法。我们为了让正常人的贪心算法得到很差的解，应该让这个数恰好超过$w[i]$一点点， 可以想象，必须让人贪心掉$w[i]$后发现剩下的数必须用一大陀小货币才能拼出。
- 我们需要确定这个数S贪心掉$w[i]$后，到底会用多小的货币才能表示出来。于是我们找一个j，使得$w[j+1] < S - w[i] < w[j]$，作为贪心后最大可用货币。 我们枚举这个j，找出利用$i+1$到j种货币能拼出的最大的严格小于$w[i]$的价格是多少，然后把这个值加上$w[j]$。
- 我们检验所有这样的值，找到最小的一个即可。

这个做法直观感觉比较靠谱。事实上我们可以严格证明这个算法的正确性，但比较复杂，有兴趣可以自行参考论文。

特别注意

无

1.5 Codeforces 15E triangles

通过情况

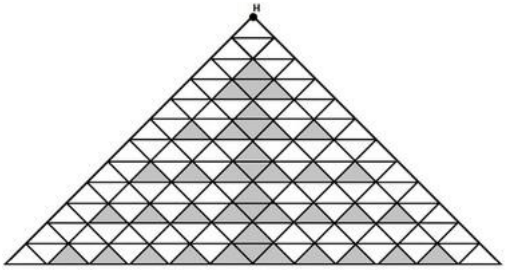
AC

关键词

递推 组合数学 找规律

题目大意

给定一个有规律的金字塔形的图形，其中一部分三角形是黑色的（见图）。



要求统计所有从最顶端出发，沿着黑线走了一条闭合简单路径又回到最顶端，且没有任何黑色三角形在闭合简单路径内部的路径条数。 $n \leq 10^6$

算法讨论

我们显然需要分析可行路线的形式，以得出有用的性质。

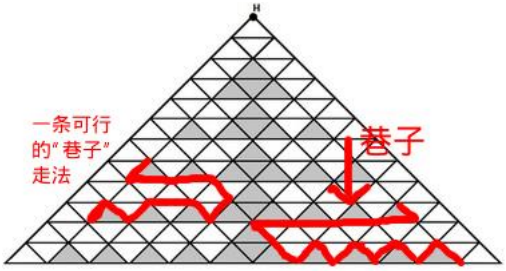
观察发现，金字塔被中间一竖列的黑色三角形分成了左右两个部分，路径只有3种形式：

- 没有进入过左部，也没有进入右部。 这类路径很少，可以手工统计。
- 只进入了左部或只进入了右部。 因为左右对称，这两小类路径条数相等，不妨设其均为S。
- 进入了左部，然后回来到左右部的交汇点（也就是第二条横线的中点），然后进入右部，再回来。 或先进入右部再进入左部。因为左右独立且对称，显然这类路径共有$2*S^2$种。

于是我们只需统计从出发点进入左部，然后回交汇点的方案数。

观察这些路径，可以发现一些共性：

- 一定有一个到达的最深层数。
- 从出发点每次往下走一层时，都有个类似“巷子”的结构，可以选择进去；也可以选择不进去，直接往下走（见下图）。
- 到达最深层后，返回出发点的路线唯一。



于是我们只需计算每层的“巷子”进入后再回来的方案数目。 可以容易的发现，第i层的“巷子”方案数是$f(i) = 2^i - 3$种。

于是我们可以简单的用乘法原理算出对于某个给定的最深层的方案数： $g(i) = f(1) * f(2) * ... * f(i)$

最终答案就是 $g(n) * g(n) * 2 + 2 * g(n) + k$. 分别对应3类路径、2类路径、1类路径的条数。（k值需手算）

特别注意

无

1.6 Codeforces 17C Balance

通过情况

AC

关键词

dp

题目大意

给定一个长度为n的只含字母'a','b','c'的串。可以进行两种操作：

- 选择相邻2个字符，把第二个字符替换成第一个
- 选择相邻2个字符，把第一个字符替换成第二个

问这个串进行任意多次变换后，能得到多少不同的串。 $n \leq 150$

算法讨论

考虑所有可以被得到的串的性质。易发现，可以得到的串中，一段连续的字母必定对应原串的某个字符， 原串的某个字符也必定对应可以得到的串中一段连续的字母（可以长度为0）。 而且，原串中两个字符的位置关系在新串中的依然得到了保持（即，两对应字母区间不相交，且位置关系与原串两字母位置关系相同）。 这个结论证明很简单，用反证即可。有了上述结论，状态便有了序的关系，状态表示就比较显然了。 $dp[l][a][b][c]$表示当前已经有a个字母'a'，b个字母'b'，c个字母'c'，现在匹配的原串字符是第l个时，不同的子串数目。转移很显然： （$next[l][ch]$表示原串在第l个字符之后（包含第l个字符）第一次出现字母ch的位置）

- $dp[l][a][b][c] => dp[next[l][\text{a}]][a+1][b][c]$
- $dp[l][a][b][c] => dp[next[l][\text{b}]][a][b+1][c]$
- $dp[l][a][b][c] => dp[next[l][\text{c}]][a][b][c+1]$
- $dp[l][a][b][c] => ans$ 当 $a+b+c == n$ 且 a,b,c两两差的绝对值不超过1.

因为a,b,c最大都只可能有$\frac{n}{3}+1$，所以实际复杂度是$O(\frac{n^4}{27})$，足以在时限内通过。

特别注意

无

1.7 Codeforces 17E Palisection

通过情况

AC

关键词

字符串 Manacher

题目大意

给定一个长度为n的小写字母串。问你有多少对相交的回文子串（包含也算相交）。$n \leq 2 * 10^6$

算法讨论

题目显然要求$O(N)$算法。

首先使用Manacher算法求出以每个字符为中心的回文串最大延伸长度。考虑计算每个字符为中心的回文串对答案的贡献。

只计算每个字符为中心的回文串与中心在这个字符之前的回文串的相交个数。因为包含也算相交，所以这个条件比较难在$O(N)$内做出来。

但考虑一下就能发现，之前的回文串与当前串不相交，等价于回文串的右边界严格小于当前串的左边界。

考虑用$sum[i]$表示右边界小于等于i的回文串个数，则每个给定中心的回文串集合对sum的贡献是"连续一段数+1"，于是$sum[]$显然可以$O(N)$扫出来。

考虑用$cnt[i]$表示左边界小于等于i的回文串个数，则每个给定中心的回文串集合对cnt的贡献也是"连续一段数+1"，因此$cnt[]$也可以O(N)扫出来。

则答案的形式是 $(cnt[r] - sum[l]) + (cnt[r] - sum[l+1]) + ... + (cnt[r] - sum[r])$ 这样。

化减得 $cnt[r] * (r - l + 1) - (sum[l] + sum[l+1] + ... + sum[r])$

于是维护sum的部分和即可$O(1)$算出答案。

特别注意

无

1.8 Codeforces 19E Fiary

通过情况

AC

关键词

无向图的dfs树 二分图

题目大意

给定一个n点m边无向图。你可以任意删一条边，要求删边后的图是二分图。问可以删哪些边。$n, m \leq 10^4$

算法讨论

这题诡异的规模让人很有暴力+玩常数水过的冲动..... 可是因为判二分图不管用dfs还是并查集都存在一定的常数，暴力水过有难度。还是考虑正经做法吧。

一个图是二分图充要条件是这个图中没有奇环。 考虑dfs这个无向图，得到一棵dfs树。因为是无向图，所以非树边只会是返祖边。

首先，如果图中不存在奇环，那么任意删一条边都是可行的。

接下来我们考虑有奇环的情况。 我们考虑一条返祖边构成的环。

- 如果这个环是偶环，那么不管删的是哪条边，这条返祖边都不会使图变成非二分的。
- 如果这个环是奇环的话，则必须从这个环中删掉一条边，否则整个图必然不是二分图。

首先，如果只存在一个奇环，那么只能删掉环上的任意一条边。

接下来我们考虑图中存在至少2个奇环的情况。

如果存在至少2个奇环，那么删任意一个奇环中的返祖边显然无济于事，因为删返祖边只能破坏这一个奇环，而不会破坏其他的。因此我们必须删树边。

接下来我们观察这条树边必须满足的性质。

首先我们观察出一些结论。

- 拥有2条属于奇环的返祖边的环必然是偶环；
- 拥有2条属于偶环的返祖边的环必然是偶环；
- 而拥有1条属于奇环的返祖边，1条属于偶环的返祖边的环必然是奇环。

我们下面将证明以下条件是可行解的必要条件：

- 这条边必须同时属于所有的奇环
- 这条边不属于任何偶环

证明：

- 如果删掉一条边后，存在一个奇环没有被破坏，那么显然不合法。所以删掉的边必须同时属于所有的奇环。
- 如果这条边属于任何一个偶环，那么必然意味着，有一条属于偶环的返祖边跨越了它。 同时，它属于所有的奇环，所以也必然存在一条属于奇环的返祖边跨越了它。 于是这会导致存在一个『拥有1条属于奇环的返祖边，1条属于偶环的返祖边的环』，由结论3，它是一个奇环。 所以，这条边不能属于任何一个偶环。

反过来，也可以用同样的方法证明，这个约束也是可行解的充分条件。

于是我们得到了最终的结论：

- 一条树边合法，当且仅当它属于所有的奇环，且不属于任何偶环。

利用dfs树我们可以非常容易的在线性时间内统计出跨越某条树边的奇环和偶环的个数。

问题在$O(N+M)$的时间内得到了解决。

特别注意

无

1.9 Codeforces 23D Tetragon

通过情况

AC

关键词

计算几何 分类讨论

题目大意

给定3个点，判定是否存在一个严格凸四边形，使得其中三条边的中点恰好是这3个点。

不超过50000组数据。 坐标范围比较小。

算法讨论

因为答案是四边形，所以被选中的3条边必然是连续的。我们枚举中间那条边。

不妨设中间的边的两个端点分别为p和q，其中点为b，相邻两边的中点为a和c。

则显然，p必须在线段(a,b)的垂直平分线上，q必须在线段(b,c)的垂直平分线上。

问题转化为：

给两条直线和一个点，问是否存在一条过该给定点的直线，使得此直线被两直线所截的线段的中点恰好是给定的这个点。

做法很显然：

直接暴力解析几何算。 设欲求直线与两直线分别交于u和v，设出u的坐标，可以得到v坐标满足的方程。

于是得到了一个二元一次方程组，解这个方程组即可。

特别注意

无

1.10 Codeforces 23E Tree

通过情况

AC

关键词

树形dp 背包dp 高精度

题目大意

给定一个n结点的树，删去若干边，要求最大化得到的所有连通块大小的乘积。

$n <= 700$

算法讨论

这类问题做法大同小异。 用$dp[i][s]$表示考虑以i为根的子树，且i所属的连通块大小是s时的最大值。

转移：对于i的每个孩子j，枚举k，用$dp[j][k] * dp[i][s]$去更新dp[i][s+k]即可。

时间复杂度$O(N^2$*高精度的复杂度)，高精度压位即可在时限内通过。

p.s 做完后想到，如果直接取对数进行比较，就没有高精的复杂度了，但不知道对数精度会不会挂，有时间可以试试。

特别注意

无

1.11 Codeforces 26E Multithreading

通过情况

AC

关键词

分类讨论 构造

题目大意

给定长度为n的正整数序列C. 设$S = 2 * sigma_{1 \le i \le n}(C_i)$. 要求找一个长度为$S$的正整数序列$A$，满足：

- 序列每个数都在1和n之间。
- 序列恰好包含恰好$2 * C_i$个数字i。 对任意$1 \le i \le n$。

这个序列将被当作指令序列执行。具体是这样的：

- 有一个长度为n的数组W，初始值均为0。还有一个特殊变量Y，初始值也是0。
- 令循环变量$i = 1$
- 如果第i个数之前有偶数个数的值也是A_i，执行：$W_{Ai} = Y$。 否则执行：$Y = W_{Ai} + 1$
- 如果$i > n$退出，否则$i = i + 1$，然后跳转到前一步。

求一个适当的序列A使得执行完成后Y的值是某个给定的值T. 或说明不存在解。

$n \le 100, S \le 100000$

算法讨论

看懂题目后就会发现是裸的分类讨论+构造题，要考虑到所有情况下的构造，情况有以下几种： （无解情况自行特判）

- $W < min\{a[i]\}$

此时用 1 2 1 1 1 ... 1 2 这样的pattern来削减$min\{a[i]\}$ （'1'代表$a[i]$最小的那个i）

总共用掉$(min\{a[i]\} - W + 1) * 2$个1和2个2，此时新的$min\{a'[i]\} = W' = W - 1$

- 经过第一步构造，可以保证$W \ge min\{a[i]\}$

此时用如下pattern构造：

... 1 2 2 3 3 ... x x 1

用这个pattern可以把两个1中间的操作全都无效化。 因为$W >= min\{a[i]\}$ ，用$a[i]$最小的i作为'1'即可保证有解。

特别注意

注意细节，不要漏讨论情况。

1.12 Codeforces 28D Do not fear, DravDe is kind

通过情况

AC

关键词

dp

题目大意

给定长度为n的四元组序列 (v_i, c_i, l_i, r_i)

要求选出一个子序列（也就是原序列去掉若干元素后得到的序列）， 使得满足：

- 子序列中所有的四元组$c_i + l_i + r_i$均相等
- 第一个元素的$l_i = 0$， 最后一个元素的$r_i = 0$
- 第i个元素的l_i等于前$i-1$个元素的c_i之和。

我们的任务是，最大化选出的子序列元素v_i之和。要求输出方案。

算法讨论

首先，我们可以把四元组按$c_i + l_i + r_i$归类。 随后只要在每一类里求出最优解即可。

因为题目的约束非常强，可以发现，一个元素j能成为i的后继元素，当且仅当$l_j = l_i + c_i$，而且题目规定了必须选出一个子序列，因此有天然的序的关系.

这时，dp就非常显然了。

我们用$dp[i]$表示：当前考虑到了i，且选择了i时，最大的v值之和

显然有 $dp[i] = max\{dp[j] \text{ 其中 } j > i \text{ 且 } l_j = l_i + c_i\} + v_i$

用个map维护l_j为某个值时最大的$dp[j]$即可做到$O(N \log N)$。

特别注意

顺带提一下，如果错误的把题目理解成了“前面至少有l_i人，后面至少有r_i人”的话，这题也是可做的。 可以证明每次任意删一个不合法元素，最后一定能得到最优解，用线段树维护一下即可做到$O(N \log N)$.

我当时写完这个算法后才发现看错题了..... ><

1.13 Codeforces 30D Kings Problem

通过情况

AC

关键词

贪心

题目大意

有$n+1$个点，其中n个点都在数轴x轴上。

求最短的从第k个点开始的哈密尔顿路。

$n \leq 10^5$

算法讨论

先对x轴上的点按x坐标排序，设排序后为$a_1...a_n$. 设x轴外的点为p

如果人正好在那个x轴外的点，可以证明最优解是 $dist(a_1, a_n)+min(dist(a_1, p), dist(a_n, p))$

否则，可以证明：

一定存在x轴上某点k，使得人先走遍1～k，回来，再走遍$k+1$～n；或者先走遍$k+1$～n，回来，再走遍1～k.

于是我们考虑2个情况：

- 从点p出发，走一个区间l, r。 最佳方案显然是从p到l或r中较近的一个，然后一路走到对面。 距离是$dist(a_l, a_r) + min(dist(p, a_l), dist(p, a_r))$
- 从点k出发，走一个区间l, r，再回到p。 最佳方案可以证明是从k走到l或r中的某一个，然后走到对面，然后走到p。 距离是$abs(a_l - a_r) + min(abs(a_k - a_l) + dist(r), abs(a_k - a_r) + dist(l))$

枚举k，我们可以$O(1)$计算出答案。取最优值。 $O(N)$

特别注意

无

1.14 Codeforces 30E Tricky and Clever Password

通过情况

AC

关键词

字符串 hashing 二分 数据结构

题目大意

给定字符串S. 求一个长度n为奇数的回文串T以及一个不超过$\frac{n}{2}$的正整数x，使得：

$S = S1 + T[1, x] + S2 + T[x+1, n-x] + S3 + T[n-x+1, n]$ $S1$,$S2$,$S3$是任意字符串，可以为空串。 $T[l, r]$表示T的第l到第r字符组成的子串，从1编号。

求满足条件的长度最大的T的长度是多长。

$n \leq 10^5$

算法讨论

这里给出两个不同的、各有优势的算法。

- 做法1：枚举最右侧分界点，我们发现随着右侧分界点向左移动，最左侧分界点单调向右移动，因此用字符串hashing可以均摊$O(1)$找到最左侧分界点。

  问题转化为，给定一个字符串，快速查询其某个子串内的最长奇回文子串。

  我们先预处理某个点作为回文串中心向两侧延伸的最长长度，这个等同于求两个串的lcp，可以部分字符串hashing后二分，$O(\log N)$内求出。

  然后考虑用ST表维护所有延伸的最长长度，$O(N \log N)$

  查询时，直接查询最大值是错误的，因为有可能某些回文串已经超出了所查询的区间。

  正确的做法是二分延伸长度，假设查询区间是$[L, R]$，二分值是M,那么只要查询区间$[L + M, R - M]$的最大值是否大于等于M即可。显然满足二分性质。

  于是我们做到了$O(\log N)$查询。总时间复杂度$O(N \log N)$

- 做法2：为描述方便，不妨设$T[x + 1, n - x]$为Mid串

  首先经过一些分类讨论后可以证明，在给定了Mid的中心的情况下。最优解的Mid一定延伸了尽可能长的长度。

  用Manacher算法可以$O(N)$求出所有点为中心的最长延伸长度。

  接下来利用字符串hashing或扩展KMP在$O(N)$求出某个最右分界点对应的最左的最左分界点在哪里。我们令右端点x对应的最左左端点是$f(x)$。

  然后，我们对每个左端点x，找到最大的y使得$f(y) = x$，也就是定义$g(y) = maxxf(x) = y$

  接下来，我们枚举Mid的中心。然后问题就转化成了查询$g(y)$的前缀最大值。

  注意如果最大值会和Mid重合，那么显然最优的$Right$就是Mid右侧所有字符，否则最优的$Right$的长度就是$g()$的前缀最大值。这一步显然$O(N)$

  总时间复杂度$O(N)$

两个做法均能AC，第二个做法时间复杂度更优秀，第一个做法优势在于思维难度很低。

特别注意

无

1.15 Codeforces 32E Hide-and-Seek

通过情况

AC

关键词

计算几何 分类讨论

题目大意

平面上有两个点A，B、 一堵墙（设为线段W）、一面镜子（设为有向线段M，逆时针方向为反射面）。

问A能不能以适当的方向发出一道光线到达B。 光线碰到墙或镜子的非反射面会被吸收，碰到镜子的反射面会镜面反射。

算法讨论

因为只有一面镜子，因此只可能有以下情况：

- 没有经过任何反射：A直接发射光线到B，中途不被挡住或反射。
- 经过了镜子的一次反射： A向B相对镜子的投影发射光线。中途不被挡住。

因为只有一面镜子，所以不可能发生多于一次反射。

接下来就是纯粹的计算几何实现了。

特别注意

注意精度，注意线段与射线非正规相交时的各种特判。

1.16 Codeforces 35E Parade

通过情况

AC

关键词

扫描线 数据结构

题目大意

平面上有n个矩形 $(Li, 0)$ - (Ri, Hi)

求轮廓线。 轮廓线可以直观理解为最外面的那圈东西。

$n \leq 10^5$

算法讨论

如果不考虑输出时的种种细节和tricks，实际问题就是，得到所有关键点的高度。

我们把一个矩形拆成2个事件点： 在Li处，向集合中插入高度Hi; 在Ri处，从集合中删除高度Hi.

把这些事件按发生的坐标排序，扫一遍。

我们需要支持：插入元素、删除元素、查询最大元素。 这显然可以用堆或其他任何数据结构完成。

接下来就是输出的各种细节，已经没有算法难度了。

特别注意

题目复杂的输出方式值得注意。

1.17 Codeforces 36E Two paths

通过情况

AC

关键词

分类讨论 欧拉回路 构造

题目大意

给定一个无向图，求两条路径，恰好覆盖每条边一次。

总边数$\leq 10^4$

算法讨论

我们需要求两条欧拉迹覆盖所有的边且恰好一次。

分类讨论+构造。情况如下（为简单起见，不考虑边界情况了）：

- 有超过4个奇数度点或超过2个连通块。 显然无解。
- 只有一个连通块，没有奇数度点。 求一个欧拉回路，任意把它分成2份即可。
- 只有一个连通块，2个奇数度点。 把这两个奇数度节点连起来，然后求一个欧拉回路，然后把这条边断开得到一个欧拉迹，任意分成2份即可。
- 只有一个连通块，4个奇数度点。 用两条边把这四个顶点配成2对连接起来。 求一个欧拉回路，删掉这两条边，得到解。
- 有两个连通块。每个连通块没有奇数度点或只有2个奇数度点。 每个连通块求一条欧拉回路或欧拉迹即可。
- 有两个连通块。其中一个连通块有4个奇数度点。无解。

特别注意

注意规模很小的时候的大量边界情况。

1.18 Codeforces 37E Trial for Chief

通过情况

AC

关键词

最短路

题目大意

给一块n*m的黑白木板和一块同样大小的全白木板。 每次，可以选择一个连通块，把它染成一种颜色。 问最少多少次操作可以达成给定的黑白木板。

$n, m \leq 50$

算法讨论

考虑最后一次染色的连通块。我们将证明：

- 假设经过了k次染色，那么必须满足，任何一点距离它的黑白交替层数不超过k。

证明如下：

充分性比较显然：

- 我们第一次把所有距离它黑白交替层数k的点都染黑（显然这必然在一个连通块内）；
- 第二次把所有距离它黑白交替层数不超过$k-1$的点染白，如此下去必然达成目标。

必要性也比较显然：

- 如果存在一个点的黑白交错距离大于k，那么一次染色最多只能增加1的黑白交错距离（如果增加了2，那么必然可以通过一些调整， 通过改变最后一次染色的连通块，得到更优结果）,结果必然最多只能到k，矛盾。

因此我们枚举每个点p，计算其他点到它的最短黑白交错距离（BFS做到$O(NM)$复杂度），取其最大值，记为$f(p)$。

所有$f()$的最小值即为答案。时间复杂度$O(N^2 * M^2)$

特别注意

无

1.19 Codeforces 39C Moon Craters

通过情况

AC

关键词

dp

题目大意

给定n个区间。选出最多的区间使得任意两个区间的关系都是相离、相切和内含之一。

$n \leq 2000$

算法讨论

考虑按右端点从小到大排序，右端点相同的按左端点从大到小排序。

则在区间i内部的区间的序号必定小于i。

考虑区间i在最外面的时候，最多能选出多少区间满足题目要求，设这个值为w_i。

则对于某个给定的i，最大化w_i等价于：从位于区间i内部(或内切)的区间中选出若干相离(或外切)的区间，使得其w和最大。

这个的可以$O(N)$时间内dp出来，并$O(N)$记录下方案： 按照之前排序的结果dp。$dp[r]$表示当前右端点在r（要离散）时最大的w和。

则$dp[r] = max(dp[r-1], max(dp[l] + w_t$ 其中t区间的左端点为l，右端点为r，权值为$w_t))$

于是利用dp结果，有$w[i] = dp[maxr] + 1$. 于是我们成功每次用$O(N)$时间计算出一个新的w值以及它的方案。

问题在$O(N^2)$内得到了解决。

特别注意

无

1.20 Codeforces 39A C*++ Calculations

通过情况

AC

关键词

贪心

题目大意

给一个C++式子，定义模糊的部分按任意次序计算（比如$a++++++a$到底先$a++$还是$++a$）

求最大可能值

算法讨论

我这题看了官方题解才会做。这个贪心真的太诡异了。

题解这么说的： 直接把操作按系数排序，无视掉$++a$和$a++$，然后按那个次序计算。

主要证明思路是：

首先证明对于相同的系数k，计算$a++$和$++a$的次序不会影响最大结果。

当k不同时，应该让较小的a乘以较小的k，$++a$之后得到更大的a，再去和更大的k相乘。

特别注意

无

1.21 Codeforces 39E What Has Dirichlet Got to Do with That?

通过情况

AC

关键词

博弈 dp

题目大意

给定a,b,n，两人博弈，轮流操作，每次可以选择$a = a + 1$或$b = b + 1$. 如果某人操作完后使得$a^b \geq n$，那么此人就输了。 问先手必胜、必输还是必和。

$1 \leq a \leq 10000$, $1 \leq b \leq 30$, $n \leq 10^9$

算法讨论

不妨先考虑a,b均大于1的情况. 易发现，此时显然不会出现和棋。 因为a,b均大于1，所以很显然，$a^b < n$的(a, b)二元组并不多，完全可以暴力dp，用map进行记忆化。

接下来考虑$a = 1$的情况， $a = 1$的特殊性在于，此时可能会出现和局。 可以发现，和局的判定条件是： $a = 1$, 且$2^b >= n$. 此时两人都只能选择$b = b + 1$，而此时a^b始终为1，故和局了。

接下来考虑$b = 1$的情况。 $b = 1$的特殊性在于，a的可能取值可能达到$O(N)$，导致TLE。

其实这个情况也很好解决：当$b = 1$,$a^2 \geq n$时，显然两人都只能选择$a = a + 1$，于是此时$n - a$的奇偶性决定了先手胜还是输， 而$a^2 < n$的a只有$n^{0.5}$个，完全可以直接记忆化。

于是这个问题得到了解决。

特别注意

无

1.22 Codeforces 40E Number Table

通过情况

AC

关键词

组合数学

题目大意

给定$n * m$矩阵，每个元素只能是1或-1. 已经有严格小于$max(n, m)$个元素被确定了。 问有多少种方案填充剩下的元素，使得满足任意一行或一列里所有元素的积都是-1.

$n, m \leq 1000$

算法讨论

首先如果$n + m$为奇数，则易证答案必然为0.

此题有个非常明显的提示： 被确定的元素『严格』小于$max(n, m)$. 这就意味着，必然有某行或某列完全是空的!

不妨假设完全空的是一个行。 我们把这行以外的元素随意填，只需满足每行的所有元素的积为−1. 然后用这行来“收尾”，显然，这一行有且恰有一种方案来填充，以满足每列的所有元素积为−1的条件。

于是，我们只需计算除这行外，每行有多少方案，使得这行的元素积为−1，然后把这些结果乘起来即可。

于是问题被转化为了1维上的问题。 显然，答案只与『还需要填的格子数目』和『已经填上的格子里的数的积』有关。

不妨设需要填s个格子，我们暴力枚举k，表示其中k个格子填了-1（当然必须满足所有数的乘积为−1）， 则填法数目显然是C_s^k，其和即为答案。

时间复杂度$O(nm)$

特别注意

无

1.23 Codeforces 43E Race

通过情况

AC

关键词

数学 细节题

题目大意

给n个分段函数，每个函数每段内都是一次函数。问你总共有多少交点。 可以保证不会有三线共点。

$n \leq 100$, 每个分段函数的段数$S \leq 100$

算法讨论

规模很小，而且不会三线共点。 暴力枚举数对(i, j)，通过某种方式计算$f(i)$和$f(j)$的交点数目，累加进答案即可。

于是任务转化为计算两个分段函数的交点数目。

我们不妨把所有的关键x坐标（也就是每段的端点x坐标）弄出来，排序，通过增加冗余的段，让两个分段函数的每个“段”的x区间对应相等以方便处理。

于是问题转化成了，每次判定两个对应段是否相交。也就是，判定两条线段是否相交。

时间复杂度$O(N^2 * S \log S)$

特别注意

注意在端点相交的特殊情况。

1.24 Codeforces 44J Triminoes

通过情况

AC

关键词

构造

题目大意

有一种$1*3$的骨牌，两端是白色的，中间是黑色的。 有一个$n*m$的国际象棋盘，其中一些格子被挖掉了。 骨牌只能放在棋盘上颜色与其对应的位置上（可以旋转）。 问是否有解。 $n,m \leq 1000$

算法讨论

首先，如果白色格子不恰好是黑格子的2倍，则显然无解。

接下来，发现这是一个匹配模型。 把棋盘中，所有中间夹了一个黑色格子的白色格子对之间连边。 得到的显然是二分图。 存在解当且仅当存在完备匹配。

当然这个图有数十万点，上百万边，显然会TLE。

经过观察，发现其实可以构造解。 考虑最左上的未被覆盖的格子，如果它是黑格子，必然无解。

否则考虑其右侧的格子，如果也没被覆盖，则这个骨牌必须横着放（否则这个白色格子就无法覆盖了），否则必须竖着放。

时间复杂度$O(NM)$

特别注意

无

1.25 Codeforces 45G Prime Problem

通过情况

AC

关键词

构造

题目大意

把1～n分成若干组，使得每组内的数的和是质数。要求组数最少。 $n \leq 6000$

算法讨论

利用歌德巴赫猜想在小范围内正确这个性质进行构造。（其实它所谓的“小范围”也已经是一个非常巨大的大数了）

- 如果所有数的和是质数，那么一组足矣。
- 如果所有数的和是偶数，那么必然可以分成2组。
- 如果所有数的和s是奇数，但$s-2$是质数，那么必然可以以2为一组，其他数为另一组。也只需两组。
- 否则易证至少需要3组。接下来我们构造一个恰好只分成3组的情况。 选择小于n的最大的质数，单独分为一组。 剩下数的和必然是偶数，转化为前一种情况. 而且可以证明删掉一个数后，也依然存在解。
- 不会出现无解的情况。

特别注意

注意N非常小的边界情况。

1.26 Codeforces 45E Director

通过情况

AC

关键词

贪心 构造

题目大意

给定n个姓氏和n个名字，要求把他们匹配起来，满足： 姓和名同字母的组合尽可能多，在满足前一个要求的条件下，字典序尽可能小。 $n \leq 100$

算法讨论

显然同字母开头的能匹配的就应该匹配起来。 剩下的可以找到一个显然的贪心算法：

- 按首字母对字串进行分类，从小到大顺序逐一处理。 设置两个列表，表示左、右还等待匹配的串。 初始时为空。
- 之前左侧有多余的字串的话，应该在保证当前字母匹配最大的前提下，用右侧字典序最小的若干来尽可能匹配左侧。
- 之前右侧有多余的字串的话，应该在保证当前字母匹配最大的前提下，用左侧字典序最小的若干来尽可能匹配右侧。
- 尽可能匹配当前字母类的串（优先匹配字典序小的）
- 未匹配的子串加入多余字串列表。 处理下一个首字母。

可以证明最终的得到的结果必然最优。

特别注意

无

1.27 Codeforces 46F Hercule Poirot Problem

通过情况

AC

关键词

Bellman-ford算法 压位

题目大意

有n个房间，m条双向边，每条边上都有个门。 有s个人，每个人在某个顶点里，有若干钥匙。

- 一个人能打开或关上输入中第k条边的门，当且仅当他有钥匙k。 每种钥匙只有1把。
- 一个人能通过某条边到达另一个顶点，当且仅当这条边的门是开的。
- 如果两个人在一个顶点里，那么他们的钥匙可以互相任意交换。

现在所有门都是关着，给出所有人的位置和钥匙情况。 过了一天，所有门依然是关上的状态。给出新的所有人位置和钥匙状况。

问这是否可能做到。

$n, m, s \leq 1000$

算法讨论

经过观察可以发现，两个状态可以互相转换，当且仅当这两个状态能到达的“极大状态”全等。

所谓“极大状态”，就是此时能开的门都开了，每个连通块的结点、人和钥匙的集合。

所谓“全等”，就是两状态连通块数相等，且存在一个连通块一一对应的方案，使得对应的连通块内结点、人、钥匙的集合均相等。

求“极大状态”实际就是一个类似floodfill的过程，可以用bellman-ford实现。这么做是$O(NM^2)$的，会TLE。

但发现，某个连通块中，某个钥匙或某个人是否存在要么是0要么是1. 我们一个int只存储一个钥匙的信息，实在太浪费了，明明可以存储32个的。

于是用bitset压位，常数小了32倍。新的时间复杂度$O(\frac{NM^2}{32})$，可以通过。

特别注意

要仔细读题，如果没注意到题目中“过了一天后所有门都是关上的”要求，就悲剧了.....

1.28 Codeforces 47E Cannon

通过情况

AC

关键词

数据结构 扫描线 数学

题目大意

有n堵墙，位置分别为$(xi, 0)$ - (xi, yi)。 有m发炮弹，每发从$(0, 0)$以固定的V为初速度（所有炮弹V一样）和不固定的角度（与x正半轴形成的角度，小于45度）发射。 假设炮弹发射后只受重力影响。 问各炮弹落点。 $n, m \leq 10^5$

算法讨论

利用物理知识可得炮弹飞到x坐标为x时的y坐标函数为（$theta$为发射角，v为固定的初速）

$f(x) = x * tan(theta) - g * x^2 * (1 + tan(theta)^2)/(2 * v^2)$

炮弹会打到一堵墙，当且仅当$f(xi) <= yi$（不考虑已经提前落地的情况，因为这种情况可以最后特判）

$f(xi) <= yi$ 移项化减得 （不妨设$tan(angle) = k$）

$(g * x^2) * k^2 - (2 * x * v^2) * k + (2 * y * v^2 + g * x^2) >= 0$

这是一个关于k的一元二次方程。 算出它的解集。

于是问题转化为，给出若干区间（每个区间额外带一个值x），每次查询一个k值，找到包含k值的区间中，x值最小的那个。

这个显然可以提取关键点，离线扫一遍，用一个multiset维护所有当前可行x即可。

$O(N \log N)$

特别注意

注意实数精度问题。

1.29 Codeforces 49E Common ancestor

通过情况

AC

关键词

dp

题目大意

给定若干替换规则 $c->ab$ 表示字符c可以替换成2个字符组成的串ab

给定两个串$s1$,$s2$，求最短的串$s3$，使得$s3$既能替换成$s1$，也能替换成$s2$,或说明这样的$s3$不存在。

替换规则不超过50种. 串长不超过50.

算法讨论

考虑反过来做。 我们试图逆向替换$s1$,$s2$（也就是把ab替换成c）得到一个相同的串$s3$.

我们先考虑计算的某个串的子串能否经过若干逆向替换使得只剩下字符c。

考虑某个串s. 用$dp[l][r][c]$表示s的第l个字符到第r个字符组成的子串能否经过若干逆向替换得到字符c。

则如果存在$l \le k < r$和字符对$(c1, c2)$满足$dp[l][k][c1]$和$dp[k+1][r][c2]$均为真，则$dp[l][r][c]$为真。这一步是$O(n^4)$的

得到了以上信息，我们可以得到，对于某个串s，从位置l开始，存在哪些r，使得区间$[l, r]$能替换成某个字符c. 不妨设vector<int> $v[l][c]$表示这个集合。

我们对$s1$和$s2$做如上处理，得到$v1$和$v2$. 然后就可以dp出最短串了。

设$dp[x][y]$表示$s1$已经匹配到了前x个字符，$s2$已经匹配到了前y个字符，此时最短的$s3$的长度。 则枚举x,y，以及$s3$的下一个字符c，则$dp[x][y]+1$可以更新$dp[x'][y']$, 其中x'属于vector<int> $v1[x+1][c]$, y'属于vector<int> $v2[y+1][c]$

时间复杂度$O(N^5)$，但实际远远达不到这个复杂度，此算法可以秒过此题。

特别注意

无

1.30 Codeforces 51F Caterpillar

通过情况

AC

关键词

边双连通分量缩点 dp

题目大意

定义一个无向图是“毛毛虫”当且仅当：

- 它是连通的，且没有重边（但允许自环以及重自环）
- 存在一条简单链，使得每个点距离链上最近的点距离不超过1

给定一张图，你每次可以把两个点(a, b)合并成一个新点c，然后删除点a,b, 同时所有的边(a, z)和(b, z)都变成(c, z)。

问最少多少次缩点后，图能变成一条毛毛虫。

$n, m \le 10^5$

算法讨论

因为题目定义的“毛毛虫”不允许重边，所以可以发现，所有的双连通分量必须缩成一个点。

于是我们先把双连通分量缩点，得到一个森林。

我们考察森林中的每棵树，考虑它们的最优解。 可以发现，所有的叶子（也就是度数为1的点）都会被保留。 然后删掉这些点后，剩下的树中求一条最长链。

容易证明，把所有非叶子也不属于最长链的点删除后，就能得到毛毛虫，而且这也是最少的删除次数。 于是这仅仅是简单的树形DP找最长链。

当然，最后，毛毛虫要求是连通的。于是我们还需要若干合并操作，把这些零散的毛毛虫合并成一条大的。 显然，我们把这些毛毛虫的“主链”首尾相连即可。

因此，最后还要花费（森林里树的棵树-1）次操作把他们合并起来。

时间复杂度$O(N+M)$

特别注意

注意，在删叶子的过程中，我们选定的根也可能是叶子。这个情况要特判。

1.31 Codeforces 53E Dead Ends

通过情况

AC

关键词

状压dp

题目大意

给定一个n点无向图。问有多少棵生成树使得恰有k个叶子。（叶子被定义为度数是1的结点） $n \leq 10$

算法讨论

规模很小，让我们想到状压dp。

用$dp[s1][s2]$表示已经用的结点状态是$s1$，其中叶子状态是$s2$时的生成树个数（$s2$必须是$s1$的子集）。

则枚举L,k，使得L属于$s1$，而k不属于$s1$。 我们将这条边加入生成树。 于是，此时新的L必然不是叶子，而新的k必然是叶子。

于是我们得到了转移： $dp[s1][s2] => dp[s1$设置第k位$][s2$清空第L位设置第k位$]$ 其中k不属于$s1$，L属于$s1$

但这样会重复计数。于是我们要求，必须从小到大依次加入叶子。于是我们在转移时额外要求$s2$的第$k+1$～n位均为0，以保证当前加入的是最大的叶子.

这样就不重不漏了。 $O(3^n * n^2)$

特别注意

注意dp的边界条件。

1.32 Codeforces 57D Journey

通过情况

AC

关键词

统计

题目大意

给定一个$n*m$的棋盘，其中一些格子被挖掉了。保证每行、每列最多被挖掉一个格子，且不会出现对角格子都被挖掉的情况。

随机选两个没被挖掉的格子，求期望最短距离。

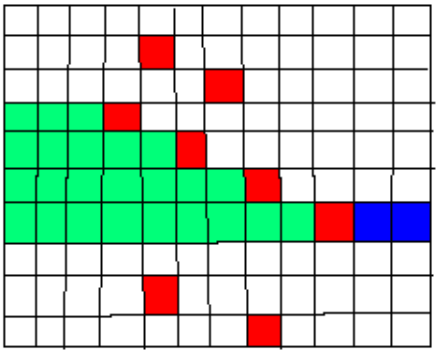
$n, m \leq 1000$

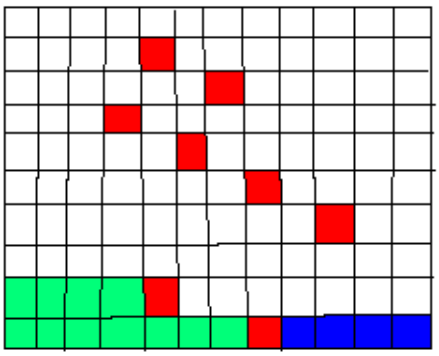
算法讨论

题目的限制很严格，我们在纸上画几个情况后可以发现，两点之间最短距离要么是其曼哈顿距离，要么比曼哈顿距离多2.

而且，当且仅当它们之间的行都恰有一个格子被挖掉，且这些格子都在它们的列坐标之间，且这些格子的列坐标递增时， 最短路径长度才会比曼哈顿距离多2，否则必然存在一条路径长度等于曼哈顿距离。（当然把上句话的“行”都换成“列”，“列”都换成“行”也成立）

我们考虑统计这些路径。 可以发现，对于某个给定的行上某点，这类“长路径”的的另一个端点的形状是必定阶梯状的一片区域（见下图）。





红色代表被挖掉的格子，蓝色代表当前考虑的行，绿色是会导致答案多2的起点格子。

维护这个阶梯状的区域即可，我们可以做到$O(NM)$

特别注意

本题细节较多，要注意避免重复计数，也要避免漏计数，还要注意一些边界情况的处理。

1.33 Codeforces 226E Noble Knight's Path

通过情况

AC

关键词

数据结构 可持久化数据结构

题目大意

给定一棵n结点的树。 最初每个结点值均为0.

要求支持完成m个操作，第i个操作发生在时间i：

- 给定a. 把一个结点a赋上值i。 （保证每个结点只会被赋值一次）
- 给定a,b,v. 查询从a到b的路径中，第k个权值小于v的结点.

$n, m \leq 10^5$

算法讨论

经典数据结构题。

此题在线处理的话，较为复杂。我们考虑离线处理。

我们先把每个结点在什么时候被赋了值弄出来。 考虑当前在时刻t的一个询问(a, b, v)。

此时，一个结点的权值不小于v，等价于在我们的树中，这个结点的权值在$[v, t]$之间。（被赋过值且在时刻t之前赋的值，才真正有效）

于是问题转化为，要求支持查询一条路径上有多少个数在区间$[l, r]$之间。

这有一个经典做法：按值建从每个点到根的可持久化线段树，维护这个点到根的路径中，有多少个点的权值在$[l, r]$之间。

接下来，预处理倍增祖先，二分答案后即可快速查询。可以做到$O(n \log^2 n)$。

特别注意

无

1.34 Codeforces 217D Bitonix' Patrol

通过情况

AC

关键词

搜索剪枝 bitmask

题目大意

有一个长度为n的序列a，你被要求删掉其中若干个数，得到的新的序列c（设其长度为L），以满足：

不存在一个长度为L的仅由-1,0和1构成的序列b，使得$sigma_{1\leq i\leq L}(ci*bi)$是一个给定的$m$的倍数

问L的最大可能值。

$n\leq 10000$, $m\leq 120$

算法讨论

我们把序列a模m的余数算出来，余数大于$\frac{m}{2}$的都替换成$m-reminder$. 这样余数的规模只有60了。

我们可以观察到，同一个余数的油箱最多只能留一个。（否则必定可以把一个系数设为1，另一个设为-1（或1），其余设为0，此时和为m的倍数，不合题意）

因此我们用一个长度为m的二进制串s维护不能拥有的油箱容量的集合，留下一个余数为k油箱会导致集合s变为$s|(s<<<k)|(s>>>k)$ 这里<<<和>>>都是循环移动。

自己手写一个东西来用位运算搞循环左右移（C++选手可以用bitset以方便实现，Java选手的bitset自带了循环移动,可以直接用）。

爆搜本来复杂度是2^{60}，看似很可怕，但用s剪枝后，可以想象绝大多数状态都剪掉了，而且这个剪枝是直接剪指数的，所以效果极好，于是就过了。

特别注意

一个代码实现时的小建议： 把位运算操作的基本功能都用宏定义起来（比如截取一个串的中间若干位等），这样可以有效防止被一堆括号和运算符绕晕。

1.35 Codeforces 67E Save the City!

通过情况

AC

关键词

计算几何

题目大意

给定一个逆时针给出的简单多边形，其中第1条边平行于x轴，且其他的边都在这条边的同一侧。

我们可以在第一条边上任意选整点。问有多少个整点，满足，这个点与任意其他多边形顶点的连线段不会与这个多边形的边正规相交。

多边形顶点数≤ 1000

算法讨论

题目可以等价于，问有多少个整点P，满足，这个多边形的顶点序列关于点P的极角序是单调的。

可以发现，为了让相邻两个顶点的关于某个点P极角序单调，点P必须在过这两点的有向直线的某一侧，也就是在某个半平面内。

另一方面，点P要求在第1条边上。 因此，多边形的每条边，实际等于对点P的一个约束，要求点P在第一条线段的某区间内。

我们能很容易的求出每个约束，以及它们的交集（因为这些约束区间都是在第一条边上的）

最后统计得到的交集线段中有多少整点即可。 时间复杂度就是排序的复杂度$O(n \log n)$

特别注意

注意实数精度误差。

1.36 Codeforces 67C Sequence of Balls

通过情况

AC

关键词

dp

题目大意

给两个串$s1$,$s2$. 你可以对$s1$执行4种操作：

- 在任意位置加入一个任意字符，代价ti
- 删除一个任意位置的字符，代价td
- 将一个任意位置的字符替换为另一个，代价tr
- 交换任意两个相邻字符，代价te，且满足$2 * te \leq ti + td$

问把$s1$变换到$s2$的最少代价。 长度都不超过4000

算法讨论

有个非常诡异的条件：$2 * te <= ti + td$。 仔细思考这个条件，发现这意味着，交换两两不重叠。（否则必然能删掉至少2个交换而替换为1个插入和1个删除）

这个性质给解题带来了极大的方便，因为我们可以dp了。

$dp[x][y]$表示$s1$匹配到了x，$s2$匹配到了y的最小代价，则转移是：

- （插入一个字符）$dp[x][y] + ti => dp[x][y + 1]$
- （删除一个字符）$dp[x][y] + td => dp[x + 1][y]$
- （替换一个字符）$dp[x][y] + tr => dp[x + 1][y + 1]$
- （本身就匹配了）如果 $s1[x + 1] == s2[y + 1]$, 则$dp[x][y] => dp[x + 1][y + 1]$

- （交换相邻字符） 设$z1$为字符$s2[y+1]$在$s1$的第$x+2$个字符后第一次出现的位置；$z2$为字符$s1[x+1]$在$s2$的第$y+2$个字符后 第一次出现的位置，则 $dp[x][y]+td*(z1-x-2)+ti*(z2-y-2)+te => dp[z1][z2]$ （删掉$i+2$到$z1-1$的所有字符，交换i和$z1$,然后把缺的字符插入回来）

因为交换两两不重叠，所以上面这么做显然是对的。

时间复杂度$O(N^2)$

特别注意

无

1.37 Codeforces 70D Professors task

通过情况

AC

关键词

数据结构 计算几何

题目大意

有一个点集，初始时点集中有三点，且保证这三点不共线。 要求支持：

- 向点集中加一个点
- 查询某点是否在点集的凸包内。

操作数$\leq 10^5$

算法讨论

经典问题，用平衡树维护全凸壳。

但直接分别维护上下凸壳的话，会比较复杂。 我们注意这题的特殊性： 一开始有3个点，而且不共线。

我们很容易想到，在这个三角形内随便选一点作为原点，其他点用极角表示法进行表示。这样我们只要以极角序为关键字维护凸包即可。

这样只需要一棵平衡树了，实现简单了很多，也少了很多细节。

特别注意

注意精度。

1.38 Codeforces 70E Information Reform

通过情况

AC

关键词

dp

题目大意

给定一棵n结点树，边权均为1. 你可以选定若干点为“消息中心”，建立一个消息中心需要k的费用。 然后，对于每个不是消息中心的结点，找到距离它最近的消息中心，设距离为s，则需要$d[s]$的费用。 问最小费用。

k值和d数组都是给定的。 d数组递增。 $n \leq 180$

算法讨论

这类问题很显然是树形dp。

我们考虑一个点最近的消息中心在哪里，无外乎2种情况： 『在其子树内』或『在其子树外』。 因此，我们得到如下状态表示：

- 用$dp[0][i][dep]$表示考虑i为根的子树，在子树内距离i为dep的某个结点处有一个消息中心，此时子树i内的所有结点的最小代价和。
- 用$dp[1][i][dep]$表示考虑i为根的子树，在子树外距离i为dep的某个结点处有一个消息中心，此时子树i内的所有结点的最小代价和。

我们定义$best[i] = min\{dp[0][i][?]\}$

那么状态转移比较显然：

- $dp[0][i][dep] = min\{dp[0][k][dep-1] + sigma\{min\{dp[1][t][dep+1], best[t]\}\}$ k是i的一个孩子，t是i的其他所有孩子。
- $dp[1][i][dep] = sigma\{min\{dp[1][t][dep+1], best[t]\}\}$ t是i的所有孩子

第一个转移是$O(N)$的，但我们可以通过一个小trick优化到均摊$O(1)$：

我们可以先计算出$sigma\{min\{dp[1][t][dep+1], best[t]\}\}$，然后再枚举$k$，利用之前计算的结果就能$O(1)$得到我们想要的每个$k$的结果了。

总时间复杂度$O(N^3)$

特别注意

注意细节。

1.39 Codeforces 156E Mrs. Hudson's Pancakes

通过情况

AC

关键词

状压DP

题目大意

题目比较复杂，我直接描述经过若干简化后的模型吧。

给定n。反复询问0～$n-1$中，在d进制下的表示能匹配某个格式串的数的乘积模p的 值（给定d和格式串，格式串只包含数字和问号，问号可以替换为任意数字）。

$n \leq 10000, p \leq 30000, 2 \leq d \leq 16$

算法讨论

因为n只有10000，可以想象，所有本质不同的合法格式串数目也不会太多。我们考虑预处理出所有d和格式串的组合下询问的答案。

首先我们枚举d。 确定了d后，格式串是一个$d+1$进制的数，如果小于d则表示真正的数字，而等于d代表问号。

因此，格式串的位数不会超过$\log_d n$，因此格式串总共有$(d+1)^{\log_d n}$种。 这个值在$d=2$时最大，约200万种。可以接受。

我们先把所有完全确定的格式串（不含问号）赋为1. 那么对于每个含问号的格式串， 我们枚举最高位问号的值，于是我们去掉了这个问号，转化成了d个之前已经预处理出的状态。

时间复杂度是 $sigma(d*(d+1)^{\log_d n})$， 化简得 $sigma_{2\leq d\leq 16}(d*n^{\log_d d+1})$ 。 经计算，这个复杂度可以接受。

特别注意

对这种题面长、任务乱的题目，耐心看懂题，就已经成功了一半.....

1.40 Codeforces 105D Entertaining Geodetics

通过情况

AC

关键词

模拟

题目大意

给定一个n*m的矩阵地图。 每个格子有一个颜色，也可能是一种特殊的颜色——透明色。

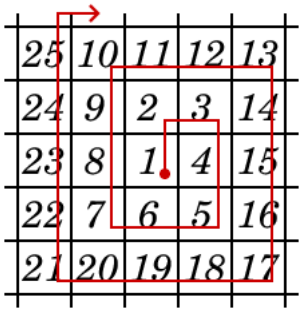
有的格子上放了一些金字塔。每个金字塔也有一种颜色，也可能是透明的。

金字塔都在格子上，每个格子上最多一个金字塔。

有一个金字塔队列。 首先你从地图上取出一个金字塔放入队列。 接下来将执行以下流程：

- 如果队列为空，结束。
- 取出队首的金字塔P，观察这个金字塔P最初放置的格子的当前的颜色，如果这个格子颜色是透明的或其颜色和金字塔P的颜色相同，则回到上一步。
- 把所有这个颜色的格子重新涂为金字塔P的颜色。这被记作一次重染色。 取出所有被重涂色的格子上的金字塔（如果有的话）

• 对每个金字塔赋上一个权值。 权值按照以下的螺旋型矩阵赋值（假设把螺旋矩阵放在地图上，'1'与P重合，那么金字塔所在位置的值就是这个金字塔的权值）



• 把金字塔按权值从小到大排序，按顺序依次压入队列，并从地图上移除。
• 返回第一步，继续执行。

你知道所有格子的颜色、所有符号的位置以及最初放入队列的金字塔的位置。计算出队列里符号彻底消失时所造成的重染色次数。

$n, m \leq 300$

算法讨论

从自然语言转化为程序语言。照着模拟吧。

一个非常有用的性质：

• 任意时刻，队列中的金字塔所在格子的颜色都一样，始终是最近一次染色的颜色。

有了上述性质，我们就省去了判定颜色这个很复杂的步骤，可以直接模拟了。

$O(NM \log N)$

特别注意

注意当染色时，要特判自身这个金字塔已经被删除了。

1.41 Codeforces 193D Two Segments

通过情况

AC

关键词

数据结构

题目大意

给定n的排列p。 问有多少对区间$[L, R]$，使得在p中标记了值为$L, L+1, L+2...R$的元素后，发现被标记的元素恰好形成了不超过2个区间。

$n \leq 10^5$

算法讨论

我们不妨称我们选中的$[L,R]$区间为“值区间”，而在p中标出的这些元素形成的区间集合为“映射区间集”

考虑枚举R，动态维护值区间为$[L,R]$时映射区间集的区间个数。$(1 \le L \le R)$

我们考虑R从$k-1$转移到k时的对映射区间集的区间个数的影响。

- 首先，增加了一个新的区间集$[k,k]$，只有一个区间。
- 对于原来就有的区间$[L,k-1]$，就相当于增加了一个新的元素k。 考虑这个元素k在p的位置：
  - 如果其左右两侧的元素都已经被标记，那么k的加入会导致总区间个数减1
  - 如果其左右两侧的元素恰有1个元素被标记，那么k的加入不会改变总区间个数。
  - 如果其左右两侧的元素都没有被标记，那么k的加入会导致总区间个数加1

我们考虑，一个元素在什么情况下，会被标记呢？ 显然，如果这个元素的值在$[L,k-1]$之间，它就被标记了！

我们考虑在固定k的情况下，L与被标记区间个数的关系。

- 当$L=k$时，显然$[k,k-1]$为空集，左右元素都不可能被标记。
- 随着L的慢慢下降，当L降低到k左右元素中较大的一个的时候，标记情况发生了一次改变： 从没有标记，变成了恰有1个标记
- L继续下降，当L降低到k左右元素中较小的一个的时候，标记情况发生了一次改变：从恰有1个标记，变成了恰有2个标记

于是我们发现，标记状况其实只有3段，每段内标记状况都是一样的。

于是，我们$[L,k]$的映射区间集的区间个数当作一个数列。 $(1 \le L \le k)$

我们需要支持：

- 对一段元素同时加一个值
- 查询一段区间里有多少个元素小于等于2.

这个咋看没什么好方法，似乎只能分块了。 $O(N^{1.5})$对$n=10^5$压力很大，不知道能不能过。

但我们仔细观察，发现我们漏掉了一个重要性质：

所有的元素恒为正整数！（这是由定义决定的，显然在有元素的情况下，区间个数不可能为负数或0）

这怎么利用呢？ 我们发现，小于等于2的元素，要么是最小值，要么是次小值！

于是，我们用线段数维护这个序列，记录最小值、次小值以及它们分别出现的次数。

这样我们就做到了$O(n \log n)$

特别注意

无

1.42 Codeforces 75E Ship's Shortest Path

通过情况

AC

关键词

计算几何

题目大意

在平面上有两点S和T和一个凸多边形。 你在凸多边形外时，只能在线段ST上走，但在凸多边形内或边缘时，可以随意走。 在凸多边形外或边缘走时，代价是每单位距离1元。 在凸多边形严格内部走时，代价是每单位距离2元。

问从S走到T的最小代价。

凸多边形顶点数不超过100.

算法讨论

我们可以证明先ST直线走到凸多边形边缘，然后沿着边缘走一段，然后走进内部，再从内部直线到T一定是不优的。

因此最优路径只有2种：

- 直接一路直线从S走到T
- 从S往T直线走，走到和凸多边形相交，然后沿着凸多边形的边缘走（顺时针或逆时针都可能），然后重新走到线段ST上，再直线走到T。

对这两种情况分别进行计算几何即可。

特别注意

细节很多，要注意。

1.43 Codeforces 76F Tourist

通过情况

AC

关键词

dp 数据结构

题目大意

某人A在X轴原点，他速度最大为V。 他知道X轴上会发生n个事件，每个事件在时间t_i，坐标x_i处发生。

如果事件发生时，此人正好在发生地点，则这个事件他就围观到了。问他最多能围观多少事件。

$n \leq 10^5$

算法讨论

考虑把事件按时间排序。设$dp[i]$表示此人围观到了第i个事件，之前围观的最多事件数目。

则 $dp[i] = max\{dp[j]\} + 1$ 其中j满足 $\frac{abs(x[i]-x[j])}{t[i]-t[j]} \leq V$

对那个条件式分离i，j得：

$x[i] - V * t[i] \leq x[j] - V * t[j] \quad (x[i] > x[j])$

$x[i] + V * t[i] \leq x[j] + V * t[j] \quad (x[i] \leq x[j])$

则考虑把$(x[i], x[i] - V * t[i])$视为一个点，则问题等价于：

- 加一个带权点
- 查询某个点左上方的点中最大的权是多少。（第二个式子同理）

离散化+树套树。 $O(N \log^2 N)$，代码较复杂。

但实际可以证明一个很强的性质（我看了题解才知道），我们可以直接无视x的限制和第一个限制， 直接用第二个限制（也就是$x[i] + V * t[i]$）做关键字，求LIS。

可以证明答案最优。 这样就不用树套树了，时间复杂度也降到了$O(N \log N)$.

特别注意

无

1.44 Codeforces 76A Gift

通过情况

AC

关键词

最小生成树 kruskal

题目大意

有一个n点m边的无向图。 每条边有两个属性(g, s)。

你被给定wg和ws. 你要求给出一对数$(g0, s0)$，使得图中删掉所有满足 $(g > g0$ 或 $s > s0)$的边后，图依然连通.

你要求最小化$g0 * wg + s0 * ws$

$n \leq 200 \quad m \leq 50000$

算法讨论

容易发现随着g的单调增加，满足条件的最小的s单调减小。

考虑对于某个固定的g值$g = g0$。 抽出所有$g \leq g0$的边。 我们要求找到一棵生成树，使得权值最大的边尽可能小。这条权值最大的边就是最小的s。

由最小生成树性质可知，我们想找的这棵树就是最小生成树。（证： 如果一棵非最小生成树得到了更优的结果ans'，则必然意味着，删掉最小生成树中一条比ans'大的边后， 依然可以找到替代边使得图连通。与最小生成树的最小性矛盾）

所以我们从小到大枚举g，相当于不断有新的边加入图中。 于是我们需要支持：

- 向图中加入一条边。
- 求这个图的最小生成树。

显然，一条边一旦离开了最小生成树，就再也不可能回到最小生成树（否则与最小生成树的最小性矛盾）

因此，只有当前最小生成树上的边，以及新加入的那条边是有用的，所以参与计算最小生成树的边始终只有$O(N)$。

于是我们暴力地每加入一条边，都做一次kruskal，复杂度是$O(MN \log N)$.

但实际上，我们利用上一次kruskal的结果，暴力插入这条边到排过序的边集数组中而依然保持数组有序，从而省去排序的log。 这样做到了时间复杂度$O(MN)$。

特别注意

无

1.45 Codeforces 77E Martian Food

通过情况

AC

关键词

反演

题目大意

有一个半径为$R1$的圆C_0，内部有一个与其内切的半径为R_2的圆C_1。 然后放一个半径为$(R_1 - R_2)$的圆D_0，使得与C_0内切，与C_1外切

然后，每次放一个圆D_k，使得其与C_0内切，与C_1和D_{k-1}外切。 问对于某个k，求Dk的半径。

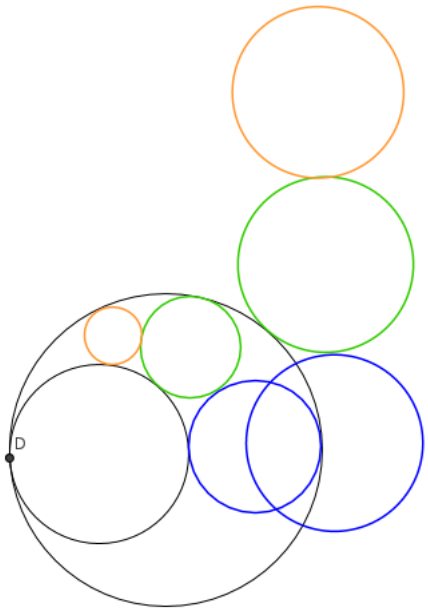
$k \leq 10000$. 多组询问。

算法讨论

我们知道，一个圆关于圆外一点反演，得到的也是一个圆。

我们设C_0与C_1的切点为O。 把圆$D_0 \quad D_k$关于O作反演，得到的是什么呢？

发现，得到的是“一摞”半径相同，x轴坐标相同，堆起来的圆！ （见下图，同色的圆代表对应圆反演的结果）



于是我们可以轻易地$O(1)$求出圆D_k关于O反演后的结果。

再把它反演回去，我们就得到了D_k的半径。 时间复杂度$O(1)$每组询问。

特别注意

无

1.46 Codeforces 79D Password

通过情况

AC

关键词

bitmask BFS DP

题目大意

有一把密码锁，由n个开关构成，最初所有开关均为关上的。 当且仅当开关$x_1, x_2, x_3...x_k$为开，其他开关为关时，密码锁才会打开。

你可以进行m种操作，每种操作有一个属性a_i. 做第i种操作时，你可以任选连续的a_i个格子，把它们全都取反（开变关，关变开）

问最少多少步打开密码锁。

$n \leq 10000 \quad m \leq 100 \quad k \leq 10$

算法讨论

很给力的题目。首先发现对连续一段开关反色很难处理，所以可以进行一个转换，如果a_i和a_{i+1}颜色相同，那么$b_i = 1$否则$b_i = 0$。

于是区间反色就变成了：对 b_i 和 b_{i+x} 反色。

我们考虑反过来做： 从给定的终止状态，倒过来搜回起始状态。 我们发现：

- 终止状态中，b_i为1的最多只有$2k$个。
- 如果两个b_i都是0那么反色显然没意义（白白增加任务）。
- 如果两个都是1，结果是两个b_i一起消掉。
- 如果b_i和b_{i+x}一个1一个0，反色相当于b_i的1移动到了b_{i+x}。

于是发现，在这个过程中，b_i为1的数目永远不会增多。

于是我们的任务转化为：

- 有不超过$2k$个石头，每次可以将某块石头移动若干距离，两块石头碰到一起就会都消失。
- 问最少多少次操作消除所有石头.

于是我们先BFS求出每个石头移动到其他石头至少需要几次移动，$O(2knm)$.

接下来，问题转化为了一个$2k$个点的无向图最小权最大匹配问题。 因为$2k \leq 20$，可以状压DP解决。 $O(2^{2k} * 2k)$.

总时间复杂度$O(2kmn + 2^{2k} * 2k)$，可以接受。

特别注意

无

1.47 Codeforces 81E Pairs

通过情况

AC

关键词

dp

题目大意

给定一个n点n边的连通图（也就是基环+树）。 每个点要么是黑的，要么是白的。 你被要求求出它的最大匹配，在保证匹配最大的情况下还要保证两端点异色的匹配尽可能多。 要输出方案。 $n \leq 10^5$

算法讨论

首先，如果给定的是一棵树，相信大家都会做了。 直接DP即可。

基环+树上的问题一个一般解法是： 对外向树先做DP，最后在基环上DP一次把这些结果合并起来。往往需要破环为链并倍长环长。

但这个解法对这题就有点杀鸡用牛刀了。

我们考虑基环上的任意一点以及与其相邻的基环上的两条边。 如果我们暴力的直接删掉一条，会怎么样呢？ 图会变成一棵树。 我们直接在这棵树上做DP，求出一个最优匹配答案。

当然，这个答案不一定最优。 因为，最优的匹配可能用到了我们删掉的那条边。 但我们观察到，因为一个点只能出现在一个匹配里， 所以既然这条我们删掉的边被用了，必然意味着另外一条基环上的邻边没有在匹配中！ 于是我们改删另外一条边，再做一次上述算法，必然能得到最优解。

于是最终算法如下：

- 任选基环上一点。
- 选择该点的两条相邻基环边中的一条，删掉，转化成一棵树，做DP，求出最优答案作为当前答案。
- 把刚才删掉的边补回来，删掉两条相邻基环边中的另一条，变成一棵树，做DP，求出最优答案，更新答案。

由刚才的分析，上述算法必然能得到最优答案。

时间复杂度O(N)

特别注意

注意多个连通块的情况。

1.48 Codeforces 82E Corridor

通过情况

AC

关键词

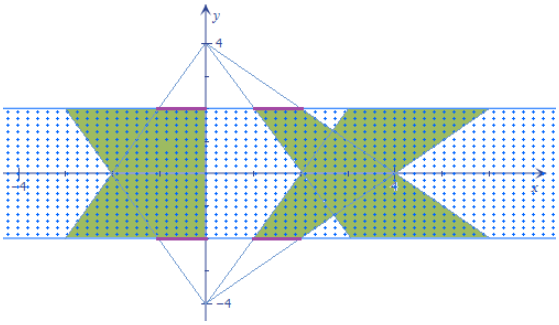
计算几何

题目大意

有一间无限长的屋子，所有y坐标绝对值不超过L的区域都属于屋子。 有两堵墙，分别是直线$y = L$和直线$y = -L$。

墙上有n扇窗户（$n \leq 500$），可以表示为线段。

现在在$(0, F)$和$(0, -F)$处各有一个点光源。（$F > L$） 问屋子里被照亮的面积。



算法讨论

显然计算几何。但直接扫描线是$O(N^3 \log N)$的，不行。

注意特殊性。发现在题目条件中，屋子里一个点，要么没照射到，要么被照射一次，要么被照射了2次。（这是显然的，因为一个点光源只有一条直线到达一个给定点）

于是，我们可以先统计单独光源照到的部分，再减去重复部分。可以发现重复部分是若干不重叠的四边形，计算它的面积即可。时间复杂度$O(N^2)$。

特别注意

注意细节，注意精度

1.49 Codeforces 83E Two Subsequences

通过情况

AC

关键词

bitmask dp

题目大意

给定长度为n的字符串序列A。A中所有串长度相同且仅由数字0和1构成。

我们定义函数：

- f(空序列) =空串
- $f(s) = s$
- $f(s1, s2)$=最小长度的有一个前缀等于$s1$并且有一个后缀等于$s2$的字符串。
- $f(a_1, a_2...a_n) = f(f(a_1, a_2...a_{n-1}), a_n) \quad n \geq 3$

你被要求把给定的长度为n的字符串序列A按顺序分为两个子序列B和C，最小化length(f(B))+length(f(C))。

算法讨论

首先定义$w[a][b]$表示a串后面拼b串会增加多少个字符。

最朴素的状态表示$dp[n][k]$表示前n个string，第二队列末端是第k个string的最小总长。转移是：

- $dp[n][k] = dp[n-1][k] + w[s[n-1]][s[n]] \quad$ 其中$k! = n-1$
- $dp[n][n-1] = min\{dp[n-1][k] + w[s[k]][s[n]]\}.$

这个是$O(N^2)$的显然不行。

这时候观察方程可以发现其实可以“优化”一维，$dp[n]$表示串n和串$n-1$不在一个队列里的最小长度，则 $dp[n] = min\{dp[k] + sum[n-1] - sum[k] + w[s[k-1]][s[n]]\}$ 其中$sum[n] = sigma_{2 \leq i \leq n}\{w[s[i-1], s[i]]\}$

但发现$w[s[k-1]][s[n]]$没有任何有用的性质，根本不能优化转移的复杂度。于是还是$O(N^2)$的。

继续观察，可以想到考虑对于某个给定的n，$dp[n][k]$这个序列能不能用数据结构维护。发现依然不可以。

我们发现矛盾在于$w[s[k]][s[i]]$是同时和$s[k]$,$s[i]$有关的，而且几乎没有任何有用性质。

突破口在于串长小于等于20，$w[]$这个函数的值也小于20.

于是我们考虑枚举w的值。然后要分离$s[k]$这个变量，解决方法是发现w确定时候（假设$w = x$）那么$s[k]$只有后X位有效的，而且状态数只有2^x种。

于是我们定义新状态$dp[n][L][STATE]$表示前n个数，第二队列的最后一个数要求恰好末L位被覆盖，而且这末L位状态是$STATE$。

这样假设每个小串串场LEN,转移就变成了

- $dp[n][L][STATE] + w[s[n-1]][s[n]] => dp[n+1][L][STATE]$
- $dp[n][L][s[n+1]$的末L位$] + LEN - L => dp[n+1][K][s[n+1]$的最后K位$]$. 其中$0 \leq L, K \leq LEN$

这样考虑n确定时候剩下两维，发现对于确定的L, 剩下的STATE这一维只是要支持全体加一个数，和查询某一个数。

这个显然什么数据结构都不需要，就是线性表。

这样从$dp[n][][]$转移到$dp[n+1][][]$就变成了$O(LEN)$的了。

总时间复杂度只有$O(N * LEN)$，完全可以接受。最后答案只要把整个表扫一遍就出来了，时间复杂度是$O(2^{LEN})$的，也完全可以接受。

特别注意

无

1.50 Codeforces 85E Guard Towers

通过情况

AC

关键词

并查集

题目大意

平面上有n个点，你被要求把每个点染成红色或蓝色。

最小化： 同色点间距离的最大值

并求出： 有多少种不同染色方案能得到这个最优值，模一个大质数。

$n \leq 5000$

算法讨论

考虑对于一个给定的限制L，判定是否可行。 显然，如果两点之间距离大于L，那么这两点必须异色。

于是我们把这些“互斥点”之间连边，判定是否二分图即可。 用并查集做到线性。

考虑从大到小枚举L，则每次L的降低都意味着加入了一条新的约束。 等于往图中不断加入新的边，直到图不是二分图。

这也显然可以用并查集做到$O(1)$每条边。

于是我们做到了$O(N^2)$。 考虑到之前需要把两点之间距离从大到小排序，就是$O(N^2 \log N)$

特别注意

无

1.51 Codeforces 86E Long sequence

通过情况

AC

关键词

矩阵乘法

题目大意

给定n（$1 \leq n \leq 50$），求两个非全零的长度为n的01序列A和C，满足： 如果我们如下定义数列f：

- $f_i = A_i$ 当$i <= n$
- $f_i = sigma_{1 \leq k \leq n}(A_{i-k} * C_k)$ 当$i > n$

则f的循环节恰为$2^n - 1$

求任意可行解。

算法讨论

因为f的下一项只和前n项有关，而且不能全零，所以总共就只有$2^n - 1$种可能，而现在循环节就是$2^n - 1$,意味着每种组合都出现了。 所以，f前n项到底是啥，其实无所谓。 我们不妨设前n项全是1，也就是A数列全1.

通过小数据打表发现，可行解非常多。 于是我们产生了一个想法，随机产生一组C值，检测它是否可行。 检测方法很简单：

- 首先我们需要检测$2^n - 1$到底是不是f的循环节。方法很简单，用矩阵乘法算一下就知道了。
- 如果前一步骤判定成功，那么我们只需确定$2^n - 1$是最大的循环节即可。因为$2^n - 1$已经是理论上最大的可能循环节了，所以如果有更小的循环节， 那么必然是$2^n - 1$的约数。
- 枚举所有$2^n - 1$的约数，进行判定。注意只需枚举不是其他约数的约数的约数即可（好绕.....）
- 如果所有的约数都判定失败，就证明我们找到了一组可行解。

上述算法在绝大多数时候都能在时限内出解。但运气实在太差得话，超时也是可能的。 所以最好还是结合打表（因为n只有50），以保证AC。

特别注意

无

1.52 Codeforces 89D Space Mines

通过情况

AC

关键词

计算几何

题目大意

在3维空间中，给定若干球和线段。

问一个球，从某点出发，以某方向射出，是否和这些物体都不相交，如果不是，找到第一个相交的物体。

物体数目≤ 10000

算法讨论

纯暴力三维计算几何题。

推了一下可以发现，就是射线和球的交点。

没啥可说的，暴力解析流，使劲写，注意别少特判。

特别注意

无

1.53 Codeforces 91D Grocer's Problem

通过情况

AC

关键词

构造 分类讨论

题目大意

给定n的排列p。 你每次可以任选不超过5个元素，随意安置它们的顺序，然后放回原位置。

问最少多少次操作才能恢复到$p_i = i$恒成立的状态。 $n \leq 10^5$

算法讨论

考虑一个轮换，如果轮换的长度不小于5，那么一定可以用一次操作把其中4个元素放回目标位置，得到一个长度少了4的轮换。 考虑经过上一步操作后，我们得到了若干长度为2、3、4的轮换。

- 首先，每个4轮换必须花一次操作单独处理。
- 接下来，我们显然应该把一个3轮换和一个2轮换合在一起处理，直到其中一种轮换用光。
- 如果3轮换还有不少于2个，我们一定应该取出2个3轮换，把一个轮换恢复到目标状态，并交换另一个轮换中2个元素，使其中一个元素到达目标，并产生一个2轮换。
- 如此反复，直到不能执行为止。
- 此时，我们要么仅剩下一个3轮换，要么仅剩下若干2轮换。
- 如果是前者，直接处理掉即可。
- 如果是后者，显然唯一的处理办法是每次处理两个2轮换，直到处理完为止。

特别注意

注意细节。

1.54 Codeforces 93D Flags

通过情况

AC

关键词

找规律 矩阵乘法

题目大意

给n个球，要求每个球染红黑白黄四色之一。满足：
- 相邻球颜色不能相同
- “白黄”不能相邻，“红黑”不能相邻
- 不能存在三个球颜色依次是“黑白红”或“红白黑”
- 对称的方案视为一种。

设n个球的方案数为$f(n)$

求$f(l)+f(l+1)+...+f(r)$ 模一大质数

$n \leq 10^9$

算法讨论

设不考虑情况5时，n个球的答案为$g(n)$ 可以证明，考虑情况5时，n个球答案为：
- $(g(n)+g(n/2))/2$　当n为奇数；
- $g(n)/2$　当n为偶数。

于是我们发现 $f(1)+f(2)+...+f(r) = (g(1)+g(2)+...+g(r))+(g(1)+g(2)+...+g((r+1)/2))$

任务转化为求$sigma_{1\leq i\leq n}(g(i))$

这个显然可以矩阵乘法。

当然也可以找出规律，规律如下：$(n \geq 2)$
- n为偶数时，答案是$19 * 3^{\frac{n}{2}-1} - 7$

- n为奇数时，答案是$11 * 3^{\frac{n}{2}} - 7$

特别注意

无

1.55 Codeforces 97C Winning Strategy

通过情况

AC

关键词

结论 dp

题目大意

给定长度为n的单调递增序列p，并定义$p_0 = 0$。你被要求给出无限序列a,满足$0 \le a_i \le n$且 $sigma_{a_{i<k}}(2 * a_i) + a_k \le (k-1) * n$恒成立。

要求使得$p[a_i]$的平均值的极限存在、并最大化这个极限。 $0 \le n \le 100$

算法讨论

此题我第一想法是二分答案后DP，DP方程比较显然（设当前二分答案为M）：

$dp[k][s]$表示到了第k场比赛，$(k-1)*n-2*sigma(a[i])$的值是s时，比平均值$M*(k-1)$多出多少。

转移： $dp[k][s] + p[i] - M => p[k+1][s+n-2*i] \quad 0 \le i \le n$

目标状态是$dp[?][0]$，因为可以发现这个无限数列必然是循环的。

另外还能发现大于n的s值是无意义的，因为所有a_i都小于等于n，而且最终s回到了0，所以一定可以把后面更大的a_i提前而消耗s。

这样的话复杂度是$\log C * n^2 *?$ 其中问号是求的循环节长度，可以猜测循环节不可能太长。

但这样终究不太靠谱..... 于是我看了标程，发现这题原来是结论题.....

结论是： 最优选择必然由不超过2个不同的$p[i]$构成。 于是做法显然。 $O(N^2)$

p.s 这样看的话，循环节长度必然不超过$2n$，于是我上面那个做法可能也可以在时限内通过。

特别注意

无

1.56 Codeforces 97A Domino

通过情况

AC

关键词

爆搜剪枝

题目大意

给定一个$n*m$的棋盘，其中一些格子被挖掉了。 保证这个矩阵没挖掉的部分可以由14个$2*2$的纸片拼起来。 可以发现，如果一个棋盘没挖掉的部分能由14个$2*2$纸片拼起来，那么拼法必然是唯一的。

对于$0 \leq i \leq j \leq 6$的所有数对(i,j)，都恰好存在一块$1*2$的骨牌，上面写着i和j。 于是总共有28块骨牌。 你被要求把这28块骨牌放在棋盘上以恰好覆盖所有没被挖掉的格子，并且使得这14个$2*2$纸片里每个纸片上的4个数都相同。

问方案数。 $n,m \leq 100$

算法讨论

规模很小，考虑爆搜。 但直接搜显然会TLE。

经过观察，我们发现，如果我们得到了任意一种可行方案，因为写着0～6的$2*2$纸片必然恰好每个出现了两次，我们可以任意的置换纸片的颜色，答案依然可行。

也就是说，很多解本质是一样的。 我们不妨只统计那些“纸片上数字第一次出现的位置递增”的解。 这些解每个都能置换出$7!=5040$个本质相同的解。 而且可以发现，每个解都能一一对应到我们求的解里。

于是我们爆搜出“纸片上数字第一次出现的位置递增”的解的个数，最后乘5040就是答案。

特别注意

无

1.57 Codeforces 98D Help Monks

通过情况

AC

关键词

结论 构造

题目大意

hanoi问题。但有圆盘是相同大小的。 移动时相同大小的可以任意次序叠放，但最终结束时还是必须按原来的顺序。

问最少次数及方案。

$n \leq 20$

算法讨论

《具体数学》原题。 递归构造如下：

设$move(a,b,cnt)$表示连续从a往b移动盘子cnt次

设$solve(a,b,c,L,order)$ 表示从a经过b移动第L及以上的盘子到c，$order$表示是保持相同盘子的原有次序还是逆序。 则构造方法如下：

- 找到第一个和$A[L]$不同的盘子，设为第R个。 设共有cnt个$A[i]$大小的盘子
- 如果$R > n$,则 如要求顺序，则$move(a,b,cnt-1)$， $move(a,c,1)$， $move(b,c,cnt-1)$ 否则$move(a,c,cnt)$。 返回。
- 否则，如要求逆序或cnt为1，则$solve(a,c,b,R,0)$， $move(a,c,cnt)$， $solve(b,a,c,R,0)$, 返回。
- 否则，$solve(a,b,c,R,0)$， $move(a,b,cnt)$， $solve(c,b,a,R,0)$， $move(b,c,cnt)$, $solve(a,b,c,R,1)$， 返回。

最优性的证明见《具体数学》。

特别注意

无

1.58 Codeforces 98C Help Greg the Dwarf

通过情况

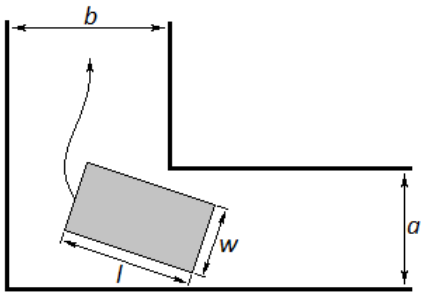
AC

关键词

三分法 分类讨论

题目大意

给定一个“L”型墓室（见下图），有一个长度L给定的的长方形，问最大的宽度W使其能拖进墓室。



算法讨论

宽度显然仅受L型的那个拐点限制。 设那个拐点坐标为(Mx, My)， 则长方形的底是直线系是 $\frac{x}{a} + \frac{y}{b} = 1 \quad a^2 + b^2 = L^2 \quad a, b \geq 0$

应用直线与点距离公式得，$dist = \frac{abs(b*Mx+a*My-ab)}{L}$

把a用$\sqrt{L^2 - b^2}$带入，求关于b的2次导数，发现可以证明始终是一个单调函数。

因此原函数是单峰的（但可能上凸也可能下凸）

于是三分答案，求函数最小值即可。

记得求了最小值之后还有若干特判以满足题目一些琐碎的要求以及不需要旋转的特殊情况。

特别注意

注意不需要旋转，直接拖进墓室的特殊情况。

1.59 Codeforces 191D Metro Scheme

通过情况

AC

关键词

构造 DP

题目大意

给定一棵n点m边的边仙人掌图（也就是每条边和每个点都至多属于一个简单环的无向图）。

你被要求用最少的简单链或者简单环来不重不漏地覆盖这棵边仙人掌图的所有边。 最小化链和简单环个数的和。

$n \leq 10^5 \quad m \leq 3 * 10^5$

算法讨论

我们先考虑树的情况。 考虑一个非根结点p，假设它有s个孩子，那么显然，最优策略是：

- 因为这个结点非根，所以其父亲到它的边必然被一条链覆盖了。我们应该把这条链延伸到它的一个孩子。
- 接下来我们显然应该把p的其他孩子两两配对，通过跨越p的路径连接起来。总共需要新生成$[\frac{s-1}{2}]$条链。
- 如果s是偶数，最后还会剩下一个孩子没匹配。只能再单独建一条链了。

于是，我们发现p对答案的贡献只与其孩子的数量s有关，也就是$[\frac{s}{2}]$.

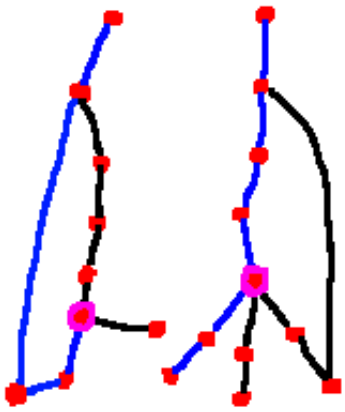
根结点类似的推一下， 可以发现对答案的贡献是$[\frac{s+1}{2}]$

接下来考虑仙人掌图的情况。我们先求它的DFS树，因为每条边至多属于一个简单环，这意味着DFS树中后向边不会相互包含或相交。

我们不妨把所有的简单环去掉。 先对剩下的森林求最优解。

因为环之间没有包含或相交关系，所以这些环两两独立互不影响。 我们考虑一个环。 我们分两种情况进行讨论：

- 环中深度最低的点不是根结点。 此时，考虑环上的其他点，如果存在任意一点的孩子数大于0（这里的孩子数不包含环上的孩子）， 那么必然存在一种方案使得在不增加答案的情况下覆盖这个环（见下图）。否则必须新建一个环来直接覆盖这个环。



- 环中深度最低的点是根结点。 此时，考虑环上的所有点，如果存在至少2个点的孩子数大于0（这里的孩子数不包含环上的孩子）， 那么必然存在一种方案使得在不增加答案的情况下覆盖这个环。否则必须新建一个环来直接覆盖这个环。构造与上图类似。

特别注意

无

1.60 Codeforces 164D Minimum Diameter

通过情况

AC

关键词

爆搜剪枝

题目大意

给定平面上的n个点。你需要去删掉恰好k个点$(k < n)$，以最小化剩下的$n - k$个点中，距离最远的两点之间的距离。

$n \leq 1000 \quad k \leq 30$

算法讨论

答案显然满足二分性质，很自然的想到应该二分答案。

设当前二分的答案是M，那么如果两点之间距离大于M，就在这两点间连边。 我们需要删掉若干点，使得所有的边的两个端点中至少有一个被删掉了。

这是一个无向图最小点覆盖问题，是NPC问题。 但k十分小，我们只能在k上做文章。

我们考虑爆搜，枚举每个点是否被删除。 如果一个点没有被删掉，那么由其出发的所有边的另一个端点都必须删掉。 但这么做还是会超时。

我们加入一个剪枝： 如果这个点的度数是1，那么显然没必要删这个点，因为删对面的那个点肯定不比删这个点来的差。 加入了这个剪枝后就过了。

实际上，我们可以证明加入这个剪枝后每次判定的时间复杂度是$O(fib(k))$的，$fib(k)$是斐波那契数列的第k项。

特别注意

无

1.61 Codeforces 150E Freezing with Style

通过情况

AC

关键词

树分治 数据结构

题目大意

给定一棵n结点边上带权树。 求一条经过的边数在$[L, R]$之间的简单路径，使得边权的中位数最大。 $n \leq 10^5$

算法讨论

这题很类似WC2010的《重建计划》，做法也差不多，都是树分治。

考虑二分答案，把大于答案的边设为1，小于的设为-1.

对树进行点分治，任务转化为求两条长度和在某区间内的路径使得其边权和大于0.

这个问题非常经典，可以用线段树维护。 $O(N \log^3 N)$.

然后使劲玩常数，使劲优化，还是过不了。最后把线段树换成zkw线段树，过了。给zkw跪。

特别注意

- 如果还是过不了，就在程序开头加一句“//orz zkw!!!!!!!!!!”就能过了2333
- 如果还是过不了说明你膜拜的不够虔诚2333
- 不管你信不信，反正我是信了。 不对，反正我是过了！ 实践已经雄辩地证明了膜拜zkw的有效性！

1.62 Codeforces 101E Candies and Stones

通过情况

AC

关键词

分治 dp

题目大意

给定一个$n*n$的矩阵的生成方式（无特殊性可抓，仅仅为了减少读入时间）。

求左上角到右下角的一条权值最大的路径，只能往右或往下走。要方案。

$n \leq 20000$. 另外： 内存只有45MB

算法讨论

这题如果不要输出方案，就是一个放在普及组都是水题的题目。

我们可以先$O(N^2)$ dp算出最优结果。然后我们考虑用分治来在可以接受的内存内求出方案。具体算法如下：

- 考虑把矩阵在中间划一刀，分成两块。
- 考虑底下的一块从右下角往中心线走，dp算出到中心线每个点的最大权。 上面的一块从左上角往中心线走，同样算出最大权。
- 我们枚举中心线上一点，如果两部分加起来正好是最优解，那么该点必然可行。
- 分治求出上面和下面分别如何走到该点。

因为得到这个分界点后，上面只需求列坐标小于等于它的部分，下面只需求列坐标大于等于它的部分， 所以总时间复杂度是$O(N^2)+O(\frac{N^2}{2})+O(\frac{N^2}{4})+..=O(N^2)$，仅仅在常数上大了一点。

于是我们在没有改变原有时间复杂度的情况下解决了本题。

特别注意

注意常数。

1.63 Codeforces 103E Buying Sets

通过情况

AC

关键词

网络流 最大权闭合图

题目大意

给定n个集合, 每个集合元素都是1和n之间的正整数。

题目保证对于任意的$k(1 \leq k \leq n)$,从集合中任意选出k个集合,都满足这k个集合的并集的势一定大于等于k.

每个集合有一个权值（可以为负）。

你被要求选出其中某些集合, 使得这些集合的并集的势等于选出的集合的数目.每个选择方案的代价是所选的集合的权值的和.

你被要求最小化总代价。 $1 \leq n \leq 300$

算法讨论

题目保证任意k个集合的并的势一定大于等于k。 这个条件怎么利用呢？

我们不妨把集合1～n作为二分图的左部，数1～n作为二分图的右部。 此时，这意味着左边的任意k个点延伸出的边一定能到达右边不少于k个点。 也就是说，这张二分图满足Halls定理。

因此，这张二分图必然存在完备匹配。

继续观察后我们可以发现，最终的答案一定是由若干匹配组成的。 也就是说，答案是这些匹配的子集。

因此，考虑如果选了集合s，那么也必须选择集合s中除了与s匹配的那个元素外的其他元素所匹配的集合。

于是，问题转化为一个最大权闭合图问题。 用网络流求解。

特别注意

无

1.64 Codeforces 105E Lift and Throw

通过情况

AC

关键词

爆搜 哈希

题目大意

给定一条标有整点$(1, 2, 3...)$的射线. 定义两个点之间的距离为其下标之差的绝对值.

有三个人A,B,C一开始在这条射线上不同的三个点, 他们希望其中某个人能够到达下标最大的点.

每个角色只能进行下面的3种操作, 且每种操作不能每人不能进行超过一次.

- 移动一定的距离
- 把另一个角色高举过头

• 将举在头上的角色扔出一段距离

具体的操作限制是这样的：

• 每个角色有一个movement range参数, 他们只能移动到没有人的位置, 并且起点和终点的距离不超过movement range.

• 如果角色A和另一个角色B距离为1, 并且角色B没有被别的角色举起, 那么A就能举起B. 同时, B会移动到A的位置, B原来所占的位置变为没有人的位置.

• 被举起的角色不能进行任何操作, 举起别人的角色不能移动.

• 每个角色还有一个throwing range参数, 即他能把举起的角色扔出的最远的距离. 注意, 一个角色只能被扔到没有别的角色占据的位置.

• 一个角色举起另一个同样举起一个角色的角色是允许的. 这种情况下会出现3个人在同一个位置的情况. 这种情况下上面的两个角色不能进行任何操作, 而最下面的角色可以同时扔出上面的两个角色.

你的任务是计算这些角色能够到达的位置的最大下标, 即最大的数字x, 使得存在一个角色能够到达x. 上文中提到的所有数值都不超过10.

算法讨论

爆搜。 搜索时，每个角色需要记录的状态如下：

- 它是否移动过、是否举起过其他角色、是否扔出过其他角色。
- 它现在所在的位置。
- 它现在是否是被人举起的状态。
- 它现在是否举着别人，举的是谁。

用hash判重即可。

特别注意

无

1.65 Codeforces 107D Crime Management

通过情况

AC

关键词

DP

题目大意

对每个小写字母c对应一个非负整数权值w_c. 保证所有非0的w_c的积不超过123.

求长度为n的小写字母串，使得这个串中，每个小写字母c的出现个数都是w_c的倍数（如果w_c是0就不允许出现这种小写字母）

问可行方案数。 $n \leq 10^{18}$

算法讨论

如果两个字符串中，各个字母c出现次数模w_c都相同，那这两个字符串显然本质相同（不管后面再拼什么串，要么都可行，要么都不可行）

因为所有非0的w_c的积不超过123，所以本质不同的状态数不超过123种。

计算出状态之间的转移，表示成矩阵。 问题转化为求这个矩阵的n次方。 快速幂即可。$O(123^3 * \log n)$

特别注意

无

1.66 Codeforces 113D Museum

通过情况

AC

关键词

高斯消元 迭代

题目大意

给定n点m边连通无向图。每个点i有一个属性值p_i。 A和B两人，最初分别在点x和点y上。每一秒钟，每个人会做出以下行动：

- 假设此人现在在点i。
- 他有p_i的几率保持原地不动。
- 他有$1 - p_i$的几率等概率的选择一条从该点连出的边，走过去，到达另一端点。

当且仅当两个人在某时刻同时位于同一个顶点时，才视为相遇。然后两人就不再移动。 （也就是说如果一个人从a到b，一个人从b到a，在边上相遇不视为相遇）

问在每个房间相遇的概率。 $n \leq 22$

算法讨论

我们考虑枚举相遇点s。 我们定义$f_{x,y}$表示A在x处，B在y处，最后在s点相遇的概率。 定义$p_{x,y}$表示人现在在x点，一秒后在y点的概率。

于是显然有方程组：

$f_{s,s} = 1$

$f_{x,x} = 0 \quad (x \neq s)$

$f_{x,y} = \sum f_{x',y'} * p_{x,x'} * p_{y,y'} \quad (x \neq y)$

暴力高斯消元是$O(N^6)$的，加上枚举的那一维，就是$O(N^7)$。 需要加上一个常数优化方可通过： 观察到『A在x点、B在y点』和『A在y点，B在x点』是等价的。 于是我们要求未知元中$x \leq y$，于是未知元的个数被削减到$\frac{n^2}{2}$个。 这样复杂度就是$O(\frac{N^7}{8})$，高斯消元常数写小一点就可以通过。

另外一个比较“讨巧”的方法是，利用答案对精度要求不是很高，直接迭代。 此方法优势是通过适当的设置参数可以在很短时间内跑出解，劣势是如果参数设置不当的话，不是TLE就是WA..

特别注意

无

1.67 Codeforces 115D Unambiguous Arithmetic Expression

通过情况

AC

关键词

DP

题目大意

我们定义UAE (unambiguous arithmetic expression) 为

- 所有的自然数是UAE,有前导零的自然数(比如0000,0010)也是UAE
- 如果X和Y是UAE, 那么"$(X)+(Y)$","$(X)-(Y)$","$(X)*(Y)$","$(X)/(Y)$"也是UAE
- 如果X是UAE, 那么"$-(X)$"和"$+(X)$"是UAE

现在给你一个只包含'0'-'9'和'+','-','*','/'的字符串，让你计算有多少种不同的UAE,满足去掉了括号符号后就变成了输入中的串，对大质数取模。

算法讨论

最朴素的做法显然是区间DP：

$dp[l][r]$表示原串$[l,r]$区间内有多少种表示法。

那么$dp[l][r] = sigma_{l \le k < r}(dp[l][k] * dp[k+1][r])$ 其中k需要是运算符，并且不能是乘号和除号后面的加减。

但这么做是$O(N^3)$的，显然不行。

我们考虑把所有数和符号提取出来。 我们先不考虑单目负号和正号的情况，只考虑双目四则运算。

这时，我们发现，一个可行解必然满足一对括号里恰好有两对匹配的括号，类似((...)(...))这样的格式。

- 我们定义两个右括号是相邻的，当且仅当它们都同时被夹在某相邻两数之间。
- 我们定义两个左括号是相互独立的，当且仅当存在一种合法方案，使得其对应右括号不相邻。

我们考虑前i个字符组成的子串的不完备的配对方案。 我们设$dp[i][s]$表示当前考虑到了第i个数，之前还有s个独立的左括号。

我们考虑在第i个字符后放括号的情况。

• 不放左括号。 此时，第$i+1$个数的后面必须放右括号以匹配最后一个独立左括号。 于是独立的左括号将少一个。$dp[i][s] => dp[i+1][s-1]$

• 放k个左括号（$k \geq 1$）。 此时，发现最后一个独立的左括号后，已经有了一对匹配的括号。因此，新放的第一个左括号一旦被配对， 最后一个独立的左括号也必须立即被配对，否则将违背题意。 因此，新加入的第一个左括号不是独立的，它对应的右括号必须与最后一个独立左括号对应的右括号相邻。 因此$dp[i][s] => dp[i+1][s+k-1] \quad k \geq 1$

综上，$dp[i][s]$可以转移到所有$dp[i+1][k]$其中$k \geq s-1$。 这显然可以用部分和优化到转移均摊$O(1)$.

接下来，我们考虑单目运算符（正负号）。 用与上面相同的方法可以证明，正负号的转移是$dp[i][s] => dp[i+1][k] \quad k \geq s \geq 1$ 和$dp[i+1][0] = 0$

于是问题在$O(N^2)$内解决了。

特别注意

无

1.68 Codeforces 120I Luck is in Numbers

通过情况

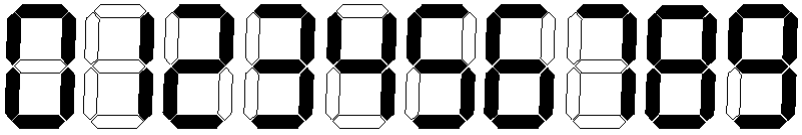
AC

关键词

构造 统计

题目大意

以下是用7个LED灯显示数字的方法。 定义数对(x,y)的幸运度$f(x,y)$是其LED表示中同时打开的LED的数量。



定一个长度为$2n$的数字数列A。 定义其幸运度为$sigma_{1 \leq k \leq n}(f(A_k, A_{k+n}))$. 你被要求找到一个长度同样为$2n$的数字数列$B$，使得$B > A$. 且$B$的幸运度大于$A$的幸运度。 如果有多个，找到最小的一个$B$。 或说明这样的$B$不存在。

$n \leq 10^5$

算法讨论

这种题很明显是逐位确定法。 我们从低到高，先找到最低的可以保留的位：

- 假设当前考虑低位第i位。
- 找到与其对应的那一位（如果$i \leq n$就是$i+n$位，否则是$i-n$位）

• 把第i位改成8，维护新的结果（我们需要找最低的可保留位，所以改成8以让新结果尽可能大）。

• 如果新结果大于原结果，跳出。 否则$i = i + 1$. （注意，第i位此时已经被改成8了，会保留下去）

此时，我们找到了最低的可以保留的位。

接下来，我们从高到低逐位确定，当前位最小能放多少。 也用同样的方法确定即可。

$O(N)$

特别注意

无

1.69 Codeforces 123E Maze

通过情况

AC

关键词

推公式 统计

题目大意

给定一棵n个点的树。 这棵树是一个迷宫，每个点i都有给定的概率s_i成为这个迷宫的入口，有t_i概率成为出口。 某人以下面的伪代码，从入口开始走：

```
DFS(x)
  if x == exit vertex then
    finish search
  flag[x] <- TRUE
  random shuffle the vertices' order in V(x) // here all permutations have equal probability to be chosen
  for i <- 1 to length[V] do
    if flag[V[i]] = FALSE then
      count++;
      DFS(y);
  count++;
```

其中$V(x)$是与x点相邻的点的序列。$Flag$数组初始时是全部为$false$。 你的任务是计算$count$的数学期望值。

$n \leq 10^5$

算法讨论

此题是推公式题。 我们不妨设这棵树的根是1，定义$size[i]$为i为根的子树的大小，$poss[i]$为起点在i为根子树内的几率。

我们考虑所有起点在i的子树内某一点，终点在i的父亲（假设i非根）的期望总步数。

经过对题目中伪代码的分析可以发现，无论出发点是哪一点，这个期望总步数其实就是$size[i] * poss[i]$！（纸上画画就能看出来了）

利用上面的结论，我们还可以得到，起点在i子树外一点，终点在i的期望步数总和是$(n - size[i]) * (1 - poss[i])$. （证明：我们考虑把DFS树的根换成$i$，那么现在$i$的父亲变成了$i$的孩子，大小是$n - size[i]$，几率是$1 - poss[i]$）

得到了这个结论，直接dfs这棵树时统计一遍即可。

特别注意

无

1.70 Codeforces 125E MST Company

通过情况

AC

关键词

二分答案 最小生成树

题目大意

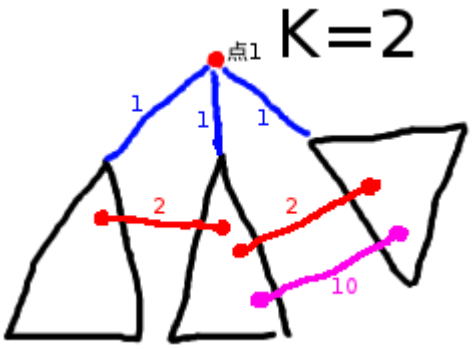
给定一个n点m边的无向带权图，求一棵点1的度数恰为K的最小生成树。

$k \leq n \leq 5000 \quad m \leq 10^5$

算法讨论

这题我最初的想法是：二分不与1相连的边的最大边权然后MST，然后看看要多少条边能把1连进去。

但很不幸，这个做法是错误的。因为最后一步里，“有多少条边才能把1连进”是离散的（这里的离散指存在达不到的整数），所以可能正好跳过了K。 而且，当跳过K的时候，夹住K的那两个答案还不一定是最优解，比如下图。 因此这个算法是错误的。



正确做法是，我们二分一个实数权值$delta$，把所有的和1相邻的边权都+$delta$，然后MST看看有多少条边。显然，随着$delta$的增大，点1的度数单调减少，满足二分性质。

这个做法因为二分权值$delta$时点1的度的变化是连续的（因为$delta$是实数），所以可以保证能得到K。

特别注意

无

1.71 Codeforces 193E Fibonacci Number

通过情况

AC

关键词

矩阵乘法 暴力

题目大意

问最小的n，使得斐波那契数的第n项模10^{13}的值是某个给定值。 保证如果存在解，那么n小于2^{63}.

算法讨论

小数据暴力找规律发现： 当$x \geq 3$时斐波那契数列模10^x的循环节就是$1.5 * 10^x$。

而且，循环节内末x位分布的很均匀（想想也是啊）。 因此，循环节内末x位是某个给定值的数很少。

于是暴力从小到大枚举x，并维护循环节内末x位的值是要求值的斐波那契项的列表。

对于每个末x位是题目要求的值的斐波那契数，暴力枚举第$x+1$位的取值，判定它的最后$x+1$位是不是要求的值即可，从而得到末$x+1$位为所求的列表。

$x = 13$时的列表中的最小值即为所求。

特别注意

无

1.72 Codeforces 145D Lucky Pair

通过情况

AC

关键词

推公式 统计 分类讨论

题目大意

定义幸运数字是那些仅由4和7构成的数字。给定长度为n的非负整数序列A. 保证A中幸运数字不超过1000个。

要求从整个序列中选出两个互不相交的子段$A[L_1, R_1]$和$A[L_2, R_2]$ $(1 \le L_1 \le R_1 < L_2 \le R_2 \le n)$， 使得不存在某个幸运数字既在$A[L_1, R_1]$出现，又在$A[L_2, R_2]$出现。

问方案数。 $n \le 10^5$

算法讨论

因为幸运数字不超过1000个，我们必须在这个上面做文章。 不妨假设序列中有s个幸运数，第i个幸运数出现在w_i处。额外定义$w_0 = 0, \quad w[s+1] = n+1$。

我们先固定$1 \le i \le s$，表示L_1位于$(w_{i-1}, w_i]$。 接着，我们枚举$i \le j \le s$，表示R_1位于$[w_j, w_{j+1})$.

我们只需快速统计出此时满足条件的区间对即可。

显然，因为区间$[L_1, R_1]$只包含了第i到第j个幸运数，这意味着不包含这些幸运数且不与$[L_1, R_1]$相交的区间$[L_2, R_2]$都是合法的。

可以想象，在去掉值在第i到第j个幸运数中出现的幸运数后，序列被分成了很多段。$[L_2, R_2]$不能跨越段，而可以在段内任取。

因此，我们假设段长分别为$c_1 \sim c_m$，我们只需维护$P = sigma_{1 \le k \le m} \frac{c_k * (c_k+1)}{2}$.

考虑随着j的增加，一些段会被破碎成两个段。 于是我们只需用一个set维护段的端点，即可快速找到新切开的位置所在段，并快速维护P值和新的段情况。

到这一步，这题已经大体完成了。 接下来还有一些琐碎的情况要考虑：

- 如果L_2和R_1都位于$[w_j, w_{j+1})$的话，可能会重复统计。因此我们要去掉这部分。
- 如果L_1和R_2都位于$(w_{i-1}, w_i]$的话，也会重复统计。 因此我们要去掉这部分。
- 如果某一段没有包含任何幸运数字，那么在上面的算法中就没统计到了，也要单独统计。

上面几种情况在纸上画一画就都能推出公式。

时间复杂度$\mathrm{O}(n^2 \log n)$

特别注意

注意细节，注意统计的不重不漏。

1.73 Codeforces 132E Bits of merry old England

通过情况

AC

关键词

最小权路径覆盖 网络流

题目大意

你被要求输出一个由n个数（$n \le 250$）组成的序列。有m个变量（$m \le 26$）给你用，变量必须是单个小写英文字母。

你被要求给出一个操作序列，输出这n个数。 你能做的操作只有2种：

- 对某个变量赋值
- 输出某个变量

变量没有初值，也就是，你必须给变量赋过值才能输出这个变量。

对于某个你给出的操作序列，定义其代价如下：

每个赋值操作的代价是『你想要赋给这个变量的值的二进制表示中1的个数』； 输出操作没有代价。

要求给出一个操作序列以输出这n个数。同时，必须最小化操作序列的代价。 要任意最优方案。

算法讨论

考虑输出了第i个数的变量又要去输出第j个数。 那么显然如果第i个数不等于第j个数，就会有一个代价。

我们考虑把n个数当成点，赋值操作当成带权边。 那么一个变量在程序中的行为可以理解为一条简单路径。

我们的任务是以最小的代价用不超过K条简单路径覆盖所有的点。

因为这张图边权的特殊性质，可以发现最优解中不会出现路径相交的情况（否则一定可以跳过交点，得到更优的解）。

因此这是一个最小权不相交路径覆盖问题。可以用费用流解决，具体算法如下：

- 左边有n个点，源点往点1连一条容量K,费用0的边，往其余点连一条容量1，费用0的边。
- 右边有n个点，每个点往汇点连一条容量1，费用0的边。
- 左边每个点$i > 1$往右边每个点$j \geq i$连容量1，权值为把原值为第i个数的变量赋值为第j个数的代价的边，表示点$i-1$的后继是j。
- 点1向所有右边的点连容量1，权值为赋值为第j个数的代价的边，表示给变量赋初值为第j个数。
- 最小费用最大流即为答案。

特别注意

无

1.74 Codeforces 138D World of Darkraft

通过情况

AC

关键词

SG 坐标转换

题目大意

给定一个$n*m$的棋盘，棋盘中每个元素都是字符'L','R','X'三者之一。 每个元素都有一个状态，“活动”或“非活动”。 最初矩阵中每个元素都是“活动”的。 A和B在这个矩阵上进行回合制博弈，双方轮流进行操作。A先手。 每次操作是这样的：

• 轮到操作的人选择一个状态是“活动”的格子。然后根据格子上的字符，将会发生以下3种事件。

• 如果格子上字符是"L"，格子会以自身为起点，向左下方和右上方对角线方向发出2条射线，射线在碰到“非活动”的格子或超出棋盘边界时才会停止。随后，射线经过的格子（包括选择的格子自身）全部变成“非活动”状态。

• 如果格子上字符是"R"，格子会以自身为起点，向右下方和左上方对角线方向发出2条射线，射线效果同上。

• 如果格子上字符是“X”，格子会以自身为起点，向左上、左下、右上、右下对角线方向发出4条射线，射线效果同上。

• 如果轮到某人操作时，所有的格子都是非活动状态的，那么这个人就输了，游戏结束。

问A是否有必胜策略。 $n, m \leq 20$

算法讨论

这是一个组合游戏博弈问题，可以应用SG定理。 我们只需找到相互独立的SG状态即可。

我们先假设原来的棋盘是在一个斜45度长方形里的。 那么发现在执行了若干操作后，棋盘必定被分割成了若干小的斜45度长方形。

于是我们得到了SG状态，应用SG定理即可。 为了方便的表示斜45度长方形，可以使用坐标转换$(x, y) => (x + y, x - y)$从而把长方形转45度使之与坐标轴平行。

时间复杂度$O(N^6)$，但常数非常小。所以可以通过。

特别注意

注意细节。

1.75 Codeforces 140F New Year Snowflake

通过情况

AC

关键词

哈希

题目大意

定义一个点集是中心对称的，指存在一个点X（不一定要属于这个点集），对于任意点集中一点a，一定有点集中某点b（b可以和a相同），使得b是a关于X的对称点。 此时称X为点集

的对称中心。

现给定平面内n个点的集合，你可以添加最多K个点（也可以不添加），使得添加后的点集是中心对称的。

问有多少个不同的点可能成为对称中心，并将他们的坐标输出。

如果可能有无穷多个对称中心，输出-1。

$n \leq 2 * 10^5 \quad 0 \leq K \leq 10$

算法讨论

任意取$K+2$个点，则因为最多只能加K个点，所以这其中必然有两个点是配对的。 枚举这两个点，得到一个对称中心。

因为$K \leq 10$，所以最多只会得到66个对称中心，用hash暴力$O(N)$判定每个对称中心是否可行。 时间复杂度$O(K^2 * N)$

特别注意

无

1.76 Codeforces 147B Smile House

通过情况

AC

关键词

倍增 最短路 dp

题目大意

给定一个n点m边有向图，求边数最少的负环。 $n \leq 300 \quad m \leq \frac{n*(n-1)}{2}$

算法讨论

首先如果没有边数最少的要求，这题应该初中生都会做：spfa一遍即可。

我们考虑需要边数最少的一个朴素做法： $dp[c][x][y]$表示走了不超过c条边从x走到了y的最小边权和。 转移显然。

但这么做是$O(N^4)$的，不可接受。

我们考虑倍增预处理以加速。 先预处理$f[c][x][y]$表示走了不超过2^c条边从x走到了y的最小权值和。 这一步是$O(n^3 \log n)$的。

接下来，我们考虑二分答案，假设当前答案为M，那么只需把M表示为二进制，然后用相同的转移利用f数组在$O(n^3 \log n)$时间内拼出$dp[M][x][y]$.

于是我们得到了一个$O(n^3 \log^2 n)$的算法

但是，因为常数原因，上面的算法依然会TLE。 怎么办呢？ 做法是逐步二分。

我们从小到大枚举c，每次尝试把2^c往答案上拼，看新的答案是否符合要求，如果依然不符合，那说明答案必然大于2^c，于是真正的拼上去，否则就不拼。

这样时间复杂度降到了$O(n^3 \log n)$

特别注意

无

1.77 Codeforces 152D Frames

通过情况

AC

关键词

离散化 暴力

题目大意

给定一个$n * m$的矩形版。 上面画了两个每边长不小于3的长方形。 你被要求识别出这两个长方形。 $n, m \leq 1000$

算法讨论

我们可以发现其实大多数格子都是没用的，完全相同的连续行、列如果大于6行（列），那么一定可以直接压成6行（列）而不会影响答案。

这样做完后，可以发现离散后的矩阵只有21×21了。 于是我们直接暴力枚举各个长方形的对角顶点，判定答案。 时间复杂度$O(21^8)$。 看上去很可怕，但实际常数极小，实际速度是CF上最快的代码之一。

特别注意

哪些行列作为关键点进行离散需要想清楚。

1.78 Codeforces 183D T-shirt

通过情况

AC

关键词

dp

题目大意

有n个人和m种衬衫。 每个人只适合一种衬衫。 你根据每个人的身材推断出了第i个人恰好适合第j种衬衫的概率是p_{ij}.

你可以任意带n件衬衫过去，你希望期望的人和衬衫的最大匹配数最大。 问最大的期望最大匹配数是多少。

$n \leq 3000 \quad m \leq 300$

算法讨论

我们不妨假设至少x人实际衬衫是k的几率是$poss[k][x]$.

利用期望的线性性质，我们可以发现，如果带x件k号衬衫过去，那么期望匹配数是$sigma_{1 \leq i \leq x} poss[k][i]$。

可以发现，在固定k时，$poss[k]$单调减少。 而假设原来有x件k号衬衫，现在多带一件k号衬衫，那么第$x+1$件衬衫对答案的贡献是$poss[k][x+1]$.

因此，我们发现，实际答案必定是整个$poss$数组中最大的n个的和，而因为$poss[k]$的单调减少性质，这最大的n个必定组成了一个可行解。

但直接计算$poss$数组是三方的，会TLE。 我们再次利用单调减少性质进行优化。发现如果$poss[k][x]$还没被选取，则必然不会选取$poss[k][x+1]$， 因此我们在选择了$poss[k][x]$后再现场计算$poss[k][x+1]$的值。

在$poss[k][0]$到$poss[k][x]$的值都知道的情况下，计算$poss[k][x+1]$是$O(N)$的。 因此最终复杂度$O(N^2 + NM)$

特别注意

无

1.79 Codeforces 217E Alien DNA

通过情况

AC

关键词

数据结构

题目大意

给定一个母串s，每次可以取出一段区间的子串，把它“错位”后再插入到这个区间的后面。 所谓“错位”指的是首先把这段序列偶数位置上的字符取出，按照原顺序接在序列后面；再把奇数位置上的字符取出，按照原顺序接在序列后面。

现在给定原串s和进行的n个操作，问最后答案的前t位。

s, t长度均不超过$3 * 10^6$，$n \leq 5000$

算法讨论

我们考虑倒过来处理，考虑最后一次操作前的字符串s'，观察哪些字符是有用的。 发现，最后一次操作必定把s'前k位的最后若干位挤出了前k位。 因此，这些被“挤出去”的位不会对答案造成影响。 于是，我们把问题转化为了求s'的前t'位。 其中$t' < t$。

如此往复，我们能推出求原串s的前t_0位. 然后再正着推过去，得到最终答案。 因为我们已经去掉了所有不对答案产生影响的位， 所以我们计算的每一位都是有用的。 因此，我们总计算次数不会超过t次。 总时间复杂度得到了保证。

过程中可以用splay进行维护，通过一些技巧可以做到$O(n \log n + T)$，当然对这题，就算暴力地写成$O(n^2 + T)$也没问题。

特别注意

注意要确定每一步都不会退化。

1.80 Codeforces 135E Weak Subsequence

通过情况

AC

关键词

推公式

题目大意

我们定义串A是串S的“子串”，如果串A在串S中出现了（也就是通常意义的子串的概念）。

我们定义长度为m的串A是长度为n的串S的“弱子序列”，当且仅当存在$1 \le i_1 \le i_2 \le .. \le i_m \le n$，使得 $S_{i_k} = A_k$对$1 \le k \le m$恒成立，且存在$1 \le k < m$使得$i_{k+1} - i_k > 1$.

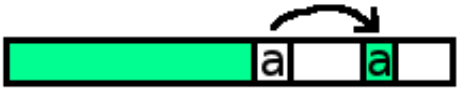
我们定义一个串是“好串”，当且仅当这个串的最长的『既是它的子串又是它的弱子序列』的串的长度恰好为给定的n。

给定n和字符集的大小K，问有多少个串是好串。对大质数取模。

$n \le 10^9, K \le 10^6$

算法讨论

经过观察，可以发现一个串的最长“子串弱子序列”必定实际上是这个串的一个前缀或后缀。（见下图，绿色是“最长子串弱子序列”的弱子序列匹配方法）



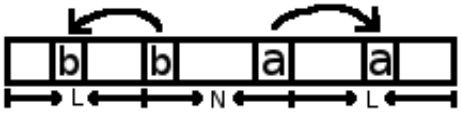
因此，由抽屉原理，可以发现，我们所求串长必须在$[n+1, n+K]$之间。 在确定了我们的串长是$n+L$后，我们的串是一个合法串，当且仅当满足以下条件：

- 『在末L位中存在一个字符与倒数第$L+1$位相同』或『在前L位中存在一个字符与第$L+1$位相同』

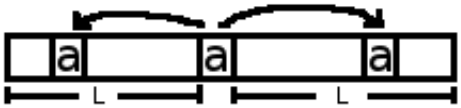
- 末L位字符两两不同。
- 前L位字符两两不同。

我们只需快速统计这类串的个数即可。 我们将分三种情况讨论：

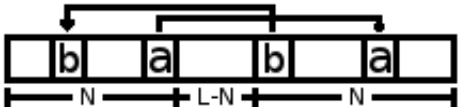
- $w-L\geq 2$，此时串形式如下：



- $w-L=1$，此时串形式如下：



- $w\leq L$，此时串形式如下：



对这三种情况我们可以分别推出公式，如下：

- $w-L\geq 2$，公式为 $(P_k^L*P_k^L)*k^2-P_k^{L+1}*P_k^{L+1})*k^{w-L-2}$
- $w-L=1$，公式为 $(P_k^L)^2-P_k^{L+1}*P_{k-1}^L$
- $w\leq L$，公式为 $P_k^{L-w}*(P_{k-L+w}^w)^2-P_k^{L-w+2}*(P_{k-L+w-2}^{w-1})^2$

特别注意

无

1.81 Codeforces 163D Large Refrigerator

通过情况

AC

关键词

爆搜剪枝

题目大意

以质因数分解形式给出V，要求作出一个三边长是整数、表面积最小、体积等于V的长方体盒子。 求三条边长。

500组询问，5秒时限，$V\leq 10^{18}$。

算法讨论

爆搜。 我们不妨假设三边长a,b,c满足$a\leq b\leq c$. 我们暴力枚举a，并保证$a\leq V^{\frac{1}{3}}$。 此时，运用平均值不等式，可知此时理论最少表面积为$4*\sqrt{V}*a+2*a^2$. 用这个条件剪枝。 接着，我们继续爆搜b的值，更新答案。 就过了。

特别注意

最优性剪枝必须在有一个较优解的情况下才能发挥良好效果。 搜索时应当以此为目标安排搜索顺序，先搜索“更可能搜出一个比较好的解”的分支。 亦可选择先贪心出一个较优解，以剪掉太差的枝条。

1.82 Codeforces 167E Wizards and Bets

通过情况

AC

关键词

dp 行列式

题目大意

给定n点m边有向无环图，其中没有入度的点被视为源点，没有出度的点被视为汇点。 保证源点和汇点数目相同。

现在考虑所有把源汇点两两配对，并用两两不相交的路径把它们两两连接起来的所有方案。

如果这个方案中，把源点按标号排序后，得到的对应汇点序列的逆序数对的个数是奇数，那么A给B一块钱，否则B给A一块钱。

问最后A的收益，对大质数取模。

$n \leq 600$

算法讨论

我们可以dp求出从某源点到某汇点的路径条数，从而得到一个矩阵。

首先，如果没有“路径必须两两不相交的限制”，那么按照题目的要求，就是裸的行列式求值（题目的给钱方式就是行列式的定义）。

然后，我们发现，加上这个限制后，其实还是裸的行列式求值。 为什么呢？ 因为如果两条路径相交，那把它们在交点处互换一下，必然会把这些有相交的解一一对应起来。 而且互换之后逆序对数的奇偶性改变了，所以两两抵消了。 这些解对答案的贡献为0.

特别注意

无

1.83 Codeforces 232D Fence

通过情况

AC

关键词

后缀数组 数据结构 函数式数据结构

题目大意

给定一个长度为n的序列A。 你被要求选择三个值L_1, L_2, s，使得满足以下条件：

- $[L_1, L_1 + s - 1]$和$[L_2, L_2 + s - 1]$不相交，且均在$[1, n]$内。
- 对$0 \leq k < s$，设$B_k = A_{L_1+k} + A_{L_2+k}$，则所有$B_k$都相等。

有Q组询问要回答：对于给定的L_1和s有多少L_2满足要求。

$n \leq 10^5 \quad Q \leq 10^5$

算法讨论

考虑对数列邻项作差，则原条件等价于新差分序列中两段序列对应数的和为0且不相交. 于是我们把差分序列后面接上差分序列每项取相反数后的序列，中间用特殊字符隔开。

询问等价于： 问在后半段序列中，有多少字符串与一给定字符串相同且对应区间不相交。

我们建立新数列的后缀数组和height数组，则询问等价于：

- 有多少后缀的与某个给定后缀的lcp大于等于s。（以满足匹配长度不小于s）
- 出现位置在序列的某个区间内。（以满足不与原区间相交）

利用LCP定理，满足与某个给定后缀的lcp大于等于s的后缀必然是连续一段。 我们可以二分得到这个区间。 于是问题进一步转化为：查询一段数中有多少个数介于某个区间之间。

这个问题可以用经典的按值建函数式trie树解决，也可以离线处理。

时间复杂度$O(n \log n) - O(\log n)$

特别注意

注意细节。

1.84 Codeforces 175E Power Defence

通过情况

AC

关键词

爆搜 dp

题目大意

有一个怪物，从$(-\infty, 0)$以每秒1的速度运动到$(+\infty, 0)$. 你可以在纵坐标为1或-1的整点上放置防御塔，每个点最多只能放一座塔。

有三种塔供你使用：

- 有nf个火塔供你使用，每个火塔的半径是rf，输出伤害是每秒df点伤害；

- 有ne个电塔供你使用，每个电塔的半径是re，输出伤害是每秒de点伤害；
- 有ns个冰塔供你使用，每个冰塔的半径是rs，不输出伤害，但可以减速怪物。 如果某一时刻怪物处于k个冰塔的作用范围内，则怪物的瞬时速度将变为$\frac{1}{k+1}$

你被要求放置这些塔使得输出伤害总值最大。 问最大可能伤害值。

$nf + ne + ns \leq 20$

算法讨论

首先可以想象，所有塔都应该紧挨在一起，占用的总列数不超过$[\frac{n+1}{2}]$列。

我们爆搜出所有冰塔的坐标。 确定了所有冰塔的坐标后，就可以DP了。

我们可以计算出在每个位置放置火塔和电塔的收益 （因为冰塔对其他塔的影响是相互独立的，都是线性增加伤害，所以可以很方便的算出来）。

接下来，我们设$dp[x][lf][le]$表示考虑到了横坐标x，还剩lf个火塔和le个电塔。 我们枚举坐标$x+1$放的火塔电塔数目（要满足三种塔在这一坐标的数目和不超过2），计算伤害，更新状态。

特别注意

无

1.85 Codeforces 176D Hyper String

通过情况

AC

关键词

dp

题目大意

给定$n \leq 2000$个字符串，串长总和不超过10^6. 你被给定一个长度不超过2000的串s和一个长串t，这个长串是由$m \leq 2000$个上面给出的字符串拼接而成的。 你被要求求出s和t的LCS.

算法讨论

直接dp的话，因为t的串长最长能达到$10^6 * n$，显然不行。

我们发现，这题的特殊性在于一个串很长，而另一个串非常短。 因此我们考虑用$dp[n][k]$表示s串匹配到了n，并且匹配了k个字符时，t串最少匹配到了哪里。

显然当n和k确定时，t串匹配位置应该尽量靠前。 通过预处理某个字符在某个串某位置下一次出现的位置，我们可以做到转移$O(1)$.

于是总复杂度$O(26 * 10^6 + 2000^2)$.

特别注意

1.86 Codeforces 178F Representative Sampling

通过情况

AC

关键词

dp

题目大意

给定$n \leq 2000$个字符串，每个字符串长度不超过500. 你被要求选出$k \leq n$个串，设其为$A_{1\sim k}$。 你被要求最大化：$\sum_{1\leq i<j\leq k} lcp(A_i, A_j)$

算法讨论

首先一种朴素做法： 把这些串建成trie树，转化为树形dp。 $dp[i][s]$表示有s个子串跨越了结点i，用一个背包dp确定各孩子的分配方案，取最大值。 但这么做是$O(N^3)$的，不行。

我们考虑把这些串排序，这样就可以利用LCP性质做一些文章了。 我们把相邻两项的height求出来。 那么，两个串的LCP就是它们之间的height的最小值。 于是，一个想法产生了： 计算每个height对答案的贡献。 我们找到当前区间内最小的height的位置，那么所有跨越了这个位置的字符串对都会对答案产生height的贡献。

我们定义状态$dp[s][l][r]$表示从区间$[l,r]$中选s个串。 我们找到$[l,r]$中height最小的位置，设为M。 则枚举设M左边选了a个，那么右边选了$s-a$个。

于是转移是： $dp[s][l][r] = \max_{0\leq a\leq s} dp[i][M-1][a] + dp[M][r][s-a] + a*(s-a)*height[M]$

可以证明这个方程是$O(N^2)$的，因为$[l,r]$区间实际组成了一棵N结点的树。

特别注意

无

1.87 Codeforces 178E The Beaver's Problem II

通过情况

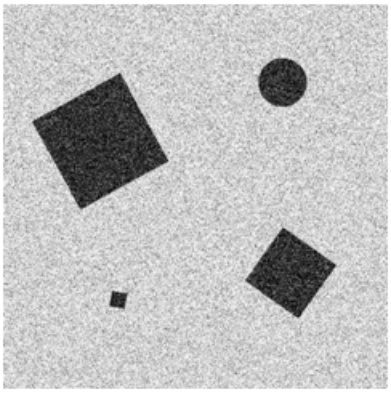
AC

关键词

乱搞 floodfill

题目大意

给定一张只包含可能被旋转过的正方形和圆的图片，图片被加入了噪音，要你识别里面有多少个正方形、多少个圆。 图片大小是$2000 * 2000$像素，保证人眼能识别出来。示例见下图。



算法讨论

如果没有噪声，那么做法比较简单：

- 我们floodfill每一个有色连通块，记录下最左上、最右下的坐标位置以及这个连通块的大小。
- 我们通过最左上、最右下的坐标对面积进行估算。 不妨设这两点距离为d. 则，如果这是一个正方形，预计面积是$\frac{d^2}{2}$，如果是圆，预计面积是$\frac{d^2*\pi}{4}$.
- 我们把预计面积与实际面积进行比较，哪个更接近实际面积就视为哪个。

如果有噪声怎么办呢？ 一个非常简单的处理方法是： 统计每个点周围的有色点个数，如果大于一个阀值就设其为黑，否则就设其为白。 这么做之后，再套用上面的算法即可。

特别注意

注意调参数。

1.88 Codeforces 180B Divisibility Rules

通过情况

AC

关键词

推结论

题目大意

我们知道，判定一个数能不能被某个数整除有时是有简单方法的。 以下是几个例子。

- 有时，只需检查待判定数最后若干位是否是它们的倍数。 我们称其为2-type。
- 有时，只需检查待判定数的各个数字和是否是它们的倍数。我们称其为3-type。

• 有时，只需检查待判定数的奇数位数字和减去偶数位数字和是否是它们的倍数。 我们称其为11-type。

• 有时，需要检查上述多条才能判定出结果，我们称其为6-type。

• 有时，某个数不满足上述所有性质，无法用上面的方法判定出来，我们称其为7-type。

给定进制数b和除数d，判定到底是属于哪个type。 $2 \le b, d \le 100$

算法讨论

我们给出各种type的判定方法。

• 如果存在一个k，使得$b^k \% d == 0$，那显然是2-type，因为k位以上不管填什么都是d的倍数。

• 如果$(b-1)\% d == 0$，那么答案是3-type. 因为原数减去其各位数字和后，必然都是$b-1$的倍数。 同时也可以证明这个条件是充分的。

• 如果$(b+1)\% d == 0$，那么答案是11-type. 证法类似， 考虑偶数位减去自身数字后一定是$b^2-1=(b+1)(b-1)$的倍数，奇数位加上自身数字后一定是$(b+1)$的倍数， 因此d必须是$b+1$的约数。 也能证明这个条件的充分性。

• 否则对d的每个极大质约数（也就是这个质因子的最高次幂）作上述判定，如果任意一个约数无法被判定，则为7-type.

• 否则为6-type.

特别注意

无

1.89 Codeforces 185D Visit of the Great

通过情况

AC

关键词

推公式 数学

题目大意

给定k, l, r,求$LCM(k^{2^l}+1, k^{2^{l+1}}+1, ..., k^{2^r}+1)\% p$

不超过10^5组询问。 $1 \le k \le 10^6 \quad 0 \le l \le r \le 10^{18} \quad 2 \le p \le 10^{18} \quad p$为素数。

算法讨论

发现k为奇数时，这些项的任意两个数之间最大公约数都是2； k为偶数时任意两个数两两互质。 于是求LCM转化成了求这些数的积。

因此，我们要求的就是$\prod_{0 \le y \le r} x^{2^y}+1$这个式子。

化减发现，这个式子中，指数从0到$2^{r+1}-1$每个数都出现了恰好一遍。

于是$\prod_{0\le y\le r} x^{2^y}+1=\sum_{0\le i<2^{r+1}} x^i=\frac{x^{2^r}-1}{x-1}$

于是，我们实际要求的式子是x^{2^r}这样的式子。 这个式子可以用快速幂+费尔马小定理快速求出。

特别注意

有很多特判和细节，比如p=2,逆元时候除数是p的倍数，还有快速幂的时候各种特判。

1.90 Codeforces 187D BRT Contract

通过情况

AC

关键词

数据结构 函数式数据结构

题目大意

有n个路口，第i个路口位于l_i处，上有一红绿灯。所有路口的红绿灯都同步：先g秒绿灯，然后r秒红灯，如此往复。 要回答Q个询问，每个询问是，在t_i秒时一辆车从坐标0出发，以每秒1的速度前进，问何时能到达最后一个路口。 $n,Q\le 10^5$

算法讨论

我们先预处理从每个路口，在红灯刚好变绿灯开始走，到终点要多长时间。 这等价于查询某个数后面的第一个介于某区间内的数在哪里。这是一个经典数据结构题，可以倒过来扫，离线+线段树。

查询时，我们只要知道第一个吃红灯的路口，往后的情况就转化成了我们预处理过的情况。

而查询第一个吃红灯的路口等价于查询第一个介于某区间内的数在哪里。这可以用线段树维护每个数第一次出现的位置以做到$O(n\log n)$。 当然也可以函数式线段树，但就要变成$O(n\log^2 n)$了。 但这两个做法都能AC.

特别注意

- C++动态申请内存很慢，能开内存池还是尽量开内存池吧。
- 注意会爆int，需要long long存。

1.91 Codeforces 176E Archaeology

通过情况

AC

关键词

dfs序 数据结构 结论

题目大意

给定一棵n点边上带权树，每个点都是黑色或白色之一。 最初所有点都是白色的。 要求支持：

- 把一个点反色
- 查询黑点的导出子树的总边权和（用最少的边把所有的黑点都连通起来的树）

$n, Q \leq 10^5$

算法讨论

我们不妨先考虑这个问题： 一棵树上有k个点，用最少的边把这k个点连通，求边权和。这个问题很不好维护。但是别怕，有神结论：

- 我们把这k个点按dfs序排序。不妨设其为$v_1, v_2, ..., v_k$。
- 则边权和= $\frac{path(v1,v2)+path(v2,v3)+...+path(vk-1,vk)+path(vk,v1)}{2}$ 其中$path(a, b)$为a, b的距离。

结论的正确性显然。

于是用set或平衡树维护当前存在的点的dfs序即可。

特别注意

无

1.92 Codeforces 196D The Next Good String

通过情况

AC

关键词

字符串hash 逐位确定法

题目大意

给定仅由小写字母组成的字符串S和一个正整数m。 要求一个长度与S相同的仅由小写字母组成的字符串$S1$，满足：

- $S1 > S$
- $S1$不包含长度大于等于m的回文子串

字符串长度$\leq 4 * 10^5$

算法讨论

首先发现『不包含长度大于等于m的回文子串』等价于『不包含长度为m或$m+1$的回文子串』， 因为更长的回文串一定包含了长度为m或$m+1$的回文串。

这种"最小的大于某个数的符合某条件的数"型问题做法几乎格式化： 逐位确定法。

我们先找到必须要改变的最低一位。 算法如下：

- 令i为S的最后一位，$s[i] = s[i] + 1$
- 判定$S[i-m+1, i]$和$S[i-m, i]$是否至少有一个是回文串。
- 如果不是，跳出。 如果是，$s[i] = s[i] + 1$。
- 如果$s[i] >' z'$，$i = i - 1$； 转回第二步。

我们得到了必须改变的最低位后，再正过来扫一遍，得到最小的答案。算法如下：

- 令$s[i] =$ "a"。
- 判定$S[i-m+1, i]$和$S[i-m, i]$是否至少有一个是回文串。
- 如果不是，$i = i + 1$，转回第一步; 否则$s[i] = s[i] + 1$，转回第二步.

得到的就是最小的答案。

特别注意

无

1.93 Codeforces 198E Gripping Story

通过情况

AC

关键词

数据结构

题目大意

在一个二维平面上，你现在的位置在(x, y)同时你手上有一块磁铁。

在这个平面上，还有n块散落的磁铁，每个磁铁都可以抽象成一个点，你的目标是吸引最多的散落的磁铁。

每一块磁铁都有五个属性，x，y，m，p，r，分别表示磁铁的横坐标，磁铁的纵坐标，磁铁的重量，磁铁的吸引力，磁铁的吸引半径。

如果一块磁铁想要把另一块磁铁吸过来的条件，则需要满足：

- 被吸引的磁铁和吸引的磁铁之间的距离小于等于吸引磁铁的吸引半径。这里距离计算的是欧几里得距离。
- 被吸引的磁铁的重量小于等于吸引磁铁的吸引力。

任何被你吸过来的磁铁都可以用来吸引新的磁铁。每块磁铁可以吸引无数多次，但是每次只能有一块磁铁在吸引，不能多块同时吸引。 同时你的位置(x, y)也是不变的。

现在你想要知道，你最多可以吸引多少散落的磁铁。

$n \leq 2.5 * 10^5$

算法讨论

如果抽象成图论问题，那么如果磁铁A能吸引到磁铁B，就从A往B连一条有向边。 问题转化为从点1出发能到达多少点。 而难度在于边数是$O(N^2)$级别的，无法暴力建立出来。

其实，这类问题做法也几乎格式化： 我们只需能支持快速返回一个新的1能到达的点即可。

考虑把距离船的距离和本身重量看作二元组(x, y)，那么我们只要支持：

- 快速返回任意一个横坐标$\leq x$,纵坐标$\leq y$的点，并把它删掉。

这个可以离散x，用线段树维护x轴、里面套有序表维护y轴就行了（因为在一个线段树区间内，如果y轴最小的点都不满足，其他点就更不满足了。 所以返回和删除的点一定是有序表的第一个元素，故不需要平衡树）

时间复杂度$O(n \log n)$

特别注意

1.94 Codeforces 200E Tractor College

通过情况

AC

关键词

暴力 分类讨论 数学

题目大意

拖拉机大学（注：拖拉机是一种扑克游戏，和“80分”类似）要给学生发奖学金。 总共M元奖学金，三种奖项。

总共有c_1人得三等奖，c_2人得二等奖，c_3人得一等奖。

学校发放奖学金要满足以下要求。

- 所有的预算都要发放完。
- $0 \leq w_1 \leq w_2 \leq w_3$. 其中$w_1, w_2, w_3$分别是三等奖、二等奖、一等奖发的奖金数目。

学校希望在满足上述条件的情况下，最小化$abs(w_1 * c_1 - w_2 * c_2) + abs(w_2 * c_2 - w_3 * c_3)$。

现在请你告诉学校，应该如何选择w_1, w_2, w_3，或者没有分配方案。

$c_1 + c_2 + c_3 \leq 300 \quad m \leq 3 * 10^5$

算法讨论

我们不妨枚举w_2的值。 这一步是$O(M)$的。

枚举了w_2的值后，我们可求出w_1的取值范围，以及w_1和w_3要满足的方程:

$$c_1 * w_1 + c_3 * w_3 = M - c_2 * w_2$$

我们对这个方程求解。 我们先把最大公约数除掉，假设得到的方程是

$$a * w_1 + b * w_3 = c$$

因为a的范围十分小，只有$O(N)$，我们可以暴力枚举w_3从而得到w_1的最小解s。 当然也可以扩展欧几理德做到$O(\log n)$。

接下来，w_1就可以表示为$b*x+s$，x的取值范围可以利用w_1的取值范围算出。

我们观察我们的目标函数，此时w_2已经被枚举过了，我们要最小化的目标函数实际是（令$V=w_2*c_2$）

$$f(x)=abs(V-(b*x+s)*c_1)+abs(V-\frac{c-(b*x+s)*a}{b}*c_3)$$

可以发现，这是一个单峰函数，可以三分答案。 时间复杂度$O(M\log N)$.

特别注意

无

1.95 Codeforces 200A Cinema

通过情况

AC

关键词

暴力

题目大意

给你一个$n*m$的01矩阵A,每个元素初始值为0，要求完成k个操作：

- 给定(x_0,y_0),你需要按下文要求找出(x,y)
- 将矩阵A的点(x,y)赋值为1,并且输出(x,y).

找(x,y)的要求如下:

- $A[x][y]==0$
- 满足条件1的情况下,(x,y)与(x_0,y_0)的曼哈顿距离尽可能小.
- 若存在多个(x,y)满足条件2，则选出x最小的.
- 若存在多个(x,y)满足条件3，则选出y最小的.

$n,m\le 2000 \quad k\le \min(n*m,10^5)$

算法讨论

我们考虑对于固定的(x_0,y_0)，如果我们再固定x，那么距离(x_0,y_0)的曼哈顿距离最小的y显然是距离y_0最近的y。

一个想法产生了： 我们要快速找到对于某个点(x,y_0)，它正上方和正下方未被占用的最近的格子是哪个。

如果我们能快速支持上述问题，那么因为总共被占用的格子只有K个，斜矩形的边长必然不超过$\sqrt{K}$， 因此我们暴力枚举$x_0-\sqrt{K}\le x\le x_0+\sqrt{K}$的所有$x$，然后取最小值就可以了。

我们不妨更清晰的描述上面的问题。 我们需要支持：

- 维护一个长度为n的数列，最初所有数都是0.

- 支持把一个数改成1
- 支持查询一个数左边最近的0是哪个。

这显然可以用并查集做到近$O(1)$.

于是问题在$O(K^{1.5})$内解决了。

特别注意

无

1.96 Codeforces 201E Thoroughly Bureaucratic Organization

通过情况

AC

关键词

结论题

题目大意

有一个长度为n的排列A，你想通过一些询问知道它是什么样的.

每次你构造一个长度为$k(0 < k \leq m)$的序列B,满足$1 \leq B_i \leq n$且B中没有相同的元素，

系统会根据序列B生成一个长度为k的序列C，C_i的值为满足$A_j == B_i$的那个j.

然后系统随机洗牌C序列后返回给你,问至少要多少次询问才能知道A究竟是什么.

$n, m \leq 10^9$， 多测，不超过1000组测试数据

算法讨论

这题我看了题解才会做..... 又是一个诡异的贪心.....

我们首先二分答案，设当前需要判定的答案是S，求出用M次询问最长能判定出的长度N。 这显然满足二分性质。

我们考虑每个数字在这S个询问中的出现情况。 因为回答被随机洗牌过了， 所以如果两个数字在S个询问中出现情况完全一样，那么我们必然不能弄清楚回答到底是给哪个数字的。

于是，对于一个n，我们如果能选出n个长度为S的01字符串，使得每个字符串两两不相同，且竖着看的话，每一列上1的个数不超过m个。

这时神结论出现了： 存在一个方案是充分必要条件所有1的个数不超过$k * m$。 充分性显然，必要性可以用调整法证明。

因此，我们根本不用管到底怎么安排方案，我们只需贪心地选择1出现次数尽量少的字符串S. 长度为S的01串中出现了k个1的方案显然有C_S^k种。 利用这个贪心即可。 因为组合数的增长非常快，只需要很少的几次计算就会导致组合数的和大于当前二分的答案了。

我也不太清楚组合数的增长到底是什么级别的，但总之实际速度非常快就是的了。

特别注意

无

1.97 Codeforces 201D Brand New Problem

通过情况

AC

关键词

状压dp

题目大意

给定由n个两两不同的单词组成的一句话s_0，以及m个由若干单词组成的长句子s_i。

若s_0的某一个排列是长句子s_i的子序列（注意不是子串），则称s_0同s_i相似，

定义两者的差异度p为满足相似条件的排列的倒置数量（即在s_0的原排列与当前排列中相对位置发生改变的单词对数）的最小值。

现在需要你求出在m个长句子中与s_0差异度最小的句子，若存在多个，则取编号最小的句子。

$1 \leq n \leq 15$，$m \leq 10$，每个长句子的单词数目不超过$5 * 10^5$，所有单词都由不超过10个小写字母组成，且总长也不超过$5 * 10^5$。

算法讨论

这m条长句子显然相互独立。 我们考虑一条长句子，设为t。我们求出t中每个单次匹配了s的哪个单词，不匹配的单词显然无意义。

于是，我们得到了一个长度为$L \leq 5 * 10^5$的列表A，每个数都介于1和n之间。 我们的任务是，求列表的一个子序列，使得这个子序列是1～n的一个排列，且逆序数最小。

因为L十分长，而n十分小，我们应该尽量设计与L无关而与n有关的状态。

我们定义状态$dp[state][k]$表示当前选中状态中，1～n中已经出现过的数的状态是$state$（是一个bitmask）， 并且当前已经出现过的数的逆序数是k个，此时最少匹配到了L串的哪个位置（显然匹配的越少越有优势，匹配的不最少的方案显然不最优）。

有了上述状态后，我们需要预处理一些东西以加速转移。

- 我们定义$next[i][c]$表示A序列在i位置后第一次出现数字c是在哪里。 这可以在$O(nL)$内预处理出。
- 我们定义$w[state][k]$表示在状态$state$后加入新数字k，新产生的逆序数对是多少。 这可以在$O(2^n * n^2)$内预处理出。

接下来，转移就显然了：

- 我们枚举一个还没有被选中的数x。
- 用$next[dp[state][k] + 1][x]$去更新$dp[state|(1 << (x - 1))][k + w[state][x]]$的答案。

dp的复杂度也是$O(2^n * n^3)$。 但实际因为k的取值很难达到$O(n^2)$，所以常数并不大。

上述过程对m个长句子各做一遍即可。 总复杂度$O(nL + m * 2^n * n^3)$，可以接受。

特别注意

无

1.98 Codeforces 204E Little Elephant and Strings

通过情况

AC

关键词

后缀数组

题目大意

给定n个小写字母字符串，询问每个字符串有多少子串（不包括空串）是所有n个字符串中至少k个字符串的子串。

$1 \leq n, k \leq 10^5$，所有字符串总长不超过10^5

算法讨论

把所有串用特殊字符隔开，算出后缀数组和height数组。接下来要利用后缀数组和height来求答案。

我最初的想法是：考虑用最短的滑窗维护恰好有k个string出现，用滑窗内的$height$最小值更新滑窗内每个元素的最大延伸长度。

但很不幸，这个做法是有反例的，原因是，对某个左界，『最短的恰好k个string出现的滑窗』和『最长的恰好k个string的滑窗』相差的那部分的最优解可能只能由这个左界更新。

我们考虑这部分元素的情况。我们不妨设左界L时，右界为R_1和R_2；左界$L+1$时，右界R_1'和R_2'。我们发现，实际要用L更新的区间仅仅是$[R_1, R_1'-1]$，因为$[R_1', R_2]$中，$L+1$的值肯定不劣于L的值，用L去更新是毫无意义的。

因此这部分区间我们暴力更新，因为总暴力更新的区间不超过$O(N)$，所以不影响复杂度。$O(n \log n)$

特别注意

无

1.99 Codeforces 207B Military Trainings

通过情况

AC

关键词

数据结构

题目大意

给定一个长度为n的序列A. 定义一个序列的“传输时间”是从第一个位置传递消息到最后一个位置所需的最短时间。 位置i能向位置j传送消息，当且仅当$1 \le j - i \le A_j$.

初始时，“当前序列”就是A序列。你被要求执行下面的流程n次：

- 输出当前序列的“传输时间”。
- 将当前序列的第一个数移动到最后一个数的后面。

算法讨论

这题有一个非常显然的贪心： 如果消息传递到了某个位置，那么这个位置应该把消息传递到尽量靠后的位置。

但很不幸，这个贪心没有利用价值。 因为这题有n组询问， 如果我们直接倍长序列，预处理每个点往后传递到哪里，那么可能会出现“传递超过了实际序列末尾”的情况。 而这个情况极难解决。 我们必须换一个思路。

我们考虑最后一个位置是从哪里接收到的消息。 发出消息给最后一个位置的地方必然位于$[n - A_n, n - 1]$.

而这倒数第二个位置，又是从哪里得到的消息呢？ 不妨设这个位置是k，那么倒数第三个发送消息的位置位于$[k - A_k, k - 1]$.

我们不妨设$S = \min\limits_{n-A_n \le k \le n-1} k - A_k$，对应的最小的$k$是$k_0$.

那么，如果消息当前的位置位于$[S, n - A_n - 1]$，我们必然应该把消息直接发送到k_0处，然后由k_0转达n。

而如果我们的位置$< S$，那么，显然不可能在2步内发送消息给n，所以它发送到的那个位置不会是n的直接消息来源。

因此，我们得到了结论： 对每个位置n，除非它是直接由起点接受到的消息，否则向它发送消息的前驱是固定的。 这个前驱就是满足$n - A_n \le k \le n - 1$且使得$k - A_k$最小的k，如果有多个，选择最小的k。

我们可以利用ST表，在$O(n \log n)$内找到每个点的前驱。

于是，我们把一个点的前驱当作这个点的父亲，我们得到了一棵树，我们尝试倒推消息的传递。 每倒推传递一次消息，就等于往树根方向走了一步。 当我们到达了一个可以由起点直接传递的点时（注意，不一定是起点！），我们就得到了最优答案。

我们的询问转化成了树上的询问。 我们对这棵树倍增祖先，查询时利用倍增数组二分即可，做到$O(n \log n)$ 的复杂度。

特别注意

注意细节。

1.100 Codeforces 207A Beaver's Calculator

通过情况

AC

关键词

贪心 构造

题目大意

给定n个序列，第i个序列长度为L_i。 不妨设第i个序列的第j项为$A_{i,j}$

现在你要在保持n个序列内部的相对顺序的情况下把这些序列合并成一个大序列B，使得满足$B_i > B_{i+1}$的i的数量尽可能少。

$n \leq 5000 \quad L_i \leq 5000$

算法讨论

我们对每个序列i，计算它有多少个k满足$A_{i,k} > A_{i,k+1}$，设其为这个序列的“层数”。

首先，因为每个序列的相对位置不能改变，所以显然最后答案的“层数”不会小于所有序列的“层数”的最大值。

我们将构造一种方案使得最后的答案等于层数最大的序列的层数。 构造如下：

- 我们每次取出所有序列中第i层的部分。
- 同一个序列的同一层内，所有数必然单调递增。
- 我们用归并排序的方法，把这些数按照排序后的顺序放入最终序列。因为每个序列内部是有序的，所以该方案必定合法。
- $i = i + 1$，返回第一步。 直到所有层都被处理完。

此时，得到的最终序列的层数等于层数最大的序列的层数，而这是理论可能的最小值，所以必定最优。

特别注意

无

1.101 Codeforces 167D Wizards and Roads

通过情况

AC

关键词

树形dp 数据结构

题目大意

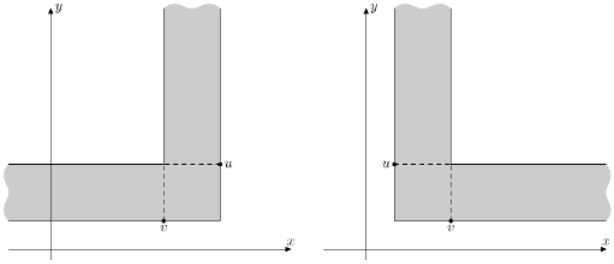
二维平面上，给定n个点，保证数据随机，且任两点横、纵坐标均不相同。 然后，如果两个点u,v满足“拐角性质”就在它们之间连一条边。 具体定义如下：

- $u_y > v_y$
- 任何严格在点u，v形成的“拐角”中的城市w，都有一个城市s并不在u，v形成的拐角中，并且s的X坐标在w和u的之间，同时$s_y > v_y$。

拐角的定义如下：$p_1(X_1,Y_1)$，$p_2(X_2,Y_2)$ $(Y_1<Y_2)$ 所形成的拐角是满足下列两个条件中的至少一个的点的集合(x,y)。

- $\min(X_1,X_2)\le X\le \max(X_1,X_2)$且$y\ge Y_1$
- $Y_1\le Y\le Y_2$且$(X-X_2)*(X_1-X_2)\ge 0$

为方便理解，图形表示如下：



现在有Q组询问，每组的意思是，把X坐标位于$[L,R]$之间的点都提取出来，然后按上述方法连边。 问得到的图的最大匹配数目。

$n,Q\le 10^5$

算法讨论

首先，经过观察发现，考虑每个点左下角和右下角的区域，容易发现，每个区域最多仅有一个点可行。且这个点是该区域中y坐标最大的点。 因此，每个点向其下方不会连超过两条边。

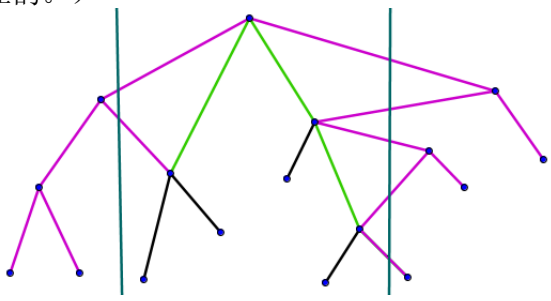
接着，考虑来自某点上方的边。 如果有两个来自该点上方的点都与该点连了边，那么因为y坐标两两不同，所以可以发现，该点和另一点必然都在第三点的某区域内。 因此，第三点在它的某区域内同时与两点连了边，与上述“每个区域最多仅有一个点可行”矛盾。 所以，任意一点上方最多只有一个点连向了它。

此时，树的形态已经被发现了：

- 这是一棵二叉树。
- 树根是y坐标最高的结点。
- 左子树由x坐标小于树根的点构成，右子树由x坐标大于树根的点构成。
- 左右子树依然满足此性质。

因此，我们只需用ST表维护区间y坐标最高的点，即可$O(n\log n)$构建出这棵树。 而且因为数据随机，这棵树的树高是$O(\log n)$级别的，很平衡。 这是一个良好的性质。

接下来，考虑怎么快速回答询问。 首先，我们发现在单独取出x坐标在$[l,r]$区间内后，树的形态改变了。 但这个改变比较简单： 只是把所有完整的子树森林按照原来的从属关系连边得到一棵树。（见下图，图中 绿色边是新添加的，黑色边是原先就存在的，红色边是不再存在的。）



而且，因为树高是$O(\log n)$的，所以新加的边的个数也是$O(\log n)$的。 我们先对原树dp出原树的某棵子树的状态。（树根被选/未被选的情况下的最大匹配数）

那么，在新树中，如果整棵子树被完整的保留了，那么答案必然不会变，直接利用即可。 否则，就现场推算转移，因为加边的个数是$O(\log n)$的，所以现场计算的次数也是$O(\log n)$的。

于是我们做到了$O(\log n)$每次询问。 总复杂度$O(n \log n)$

特别注意

无

1.102 Codeforces 209C Trails and Glades

通过情况

AC

关键词

推结论 欧拉回路

题目大意

给定n点m边无向图，可能有自环和重边。 问最少添加多少条边后，使得图存在从点1出发又回到点1的欧拉回路。

$n, m \leq 10^6$

算法讨论

利用欧拉回路存在的性质，先求出每个连通块内度数是奇数的点的个数。 我们需要加边以消除所有奇度数点。

然后我们还得把所有连通块连通起来。 可以发现，如果一个连通块包含奇数度数点，那么就不需要额外加边就能与其他连通块连通 （想象把这些连通块的奇数点收尾相连）。

但如果一个连通块里没有奇度数点，那么必须额外花费1条边才能把它与其他部分连接起来。

于是答案是：

- 如果全图连通，则答案是奇数度数点的个数/2.
- 否则答案是奇数度数点的个数/2+不含奇数度数点的连通块的个数。

特别注意

有一些细节要考虑。 注意孤立点的特殊处理。

1.103 Codeforces 212B Polycarpus is Looking for Good Substrings

通过情况

AC

关键词

二分查找 哈希

题目大意

给定长度为n的小写字母串。给定很多个集合。对于每个集合，求以下子串$s[l,r]$的数量：

- 子串中出现的字母必须出现在集合中。
- 集合中的字母必须在子串中找得到
- 不存在比这个子串更长的子串$s[l',r']$,使得$s[l',r']$满足1,2，且$l' <= l <= r <= r'$

算法讨论

如果对某个子串，询问它出现的字母集合能使得这个子串成为答案，那么我们称这个子串为“好子串”。

经过观察发现，以某个字母开头的好子串最多只有26个，而且只需记录某个位置后第一次出现的是哪个字母即可快速求出。 于是我们得到了26n个备选答案和对应的询问集合，在所有询问中查找是否存在这个询问集合，如果存在就把它的答案加一即可。 这一步可以先把所有询问集合排序后二分查找，或哈希亦可。

如果最后一步用二分查找，复杂度是$O(26 * n * \log Q)$。 用哈希就是$O(26 * n + Q)$.

特别注意

注意常数。 注意重复询问某一个相同的集合的情况。

1.104 Codeforces 212D Cutting a Fence

通过情况

AC

关键词

推结论 部分和

题目大意

给定长度为n的序列A，定义

$$f(L) = \frac{\sum_{s=1}^{n-L+1} \min_{i=s}^{s+L-1} A_i}{n-L+1}$$

求$f(1)$～$f(n)$的值。 $n \leq 10^6$

算法讨论

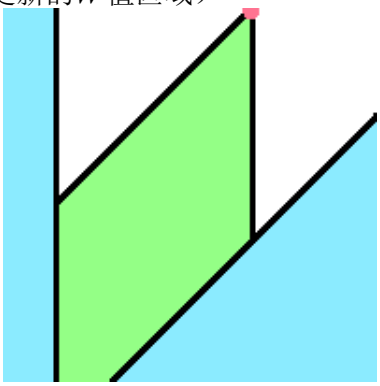
我们不妨定义$W_{i,j} = \min_{k=i}^{i+j-1} A_k$，如果数对$(i,j)$不合法（超过了边界），则$W_{i,j} = 0$。

我们画出一张$n * n$的表格表示W的值。 那么$f(L)$的值就是第L行所有W值的和。

我们考虑从小到大插入元素，计算插入元素对答案的贡献。

因为我们选择的是最小的元素，所以如果区间跨越了这个元素，并且这个区间之前并没有被求解过，那么这个区间的答案必然就是这个元素的值。

因此，我们发现插入一个元素的时候，能计算出W值在表格中对应一个45度平行四边形。（见下图，图中蓝色是已经被计算过的W值，红点是当前插入的最小值，绿色是这个值能更新的W值区域）



考虑如何快速计算出平行四边形的边界，我们实际的任务是快速找到左边、右边最近的已经插入过的点在哪里，这个可以用倒序离线+并查集处理从而做到$O(N)$。

得到了这个平行四边形的形状后，我们考虑这个平行四边形考虑对每一行的贡献总值（也就是对f的贡献），发现必然是3段等差数列合起来。

于是问题转化为：

- 维护一段序列。
- 支持给序列的一个区间增加一个等差序列。
- 求最后的结果。

显然可以离线后利用部分和$O(N)$完成。

总复杂度如果不算排序的话，就是$O(N)$了

特别注意

无

1.105 Codeforces 212C Cowboys

通过情况

AC

关键词

dp

题目大意

有n个人站成一圈，每人拿枪指着自己左边相邻的人或右边相邻的人。

如果两个人发现他们正在互相瞄准，1秒种后，他们就会一起掉转枪口。

现在给出这n个人的枪口朝向，问1秒种前有多少种可能的状态。

$n \leq 100$

算法讨论

我们考虑DP. 先枚举一秒钟前第一个人的枪口朝向，这样问题就转化成了链上的情况。

用$dp[n][dir]$表示考虑了前n个人，第n个人一秒前的枪口朝向是dir时的符合条件的状态数。

转移情况较多，但只要细心就可以推出来。 具体如下：（假设$d[i]$表示题目给出的第i个人当前枪口朝向）

- $dir == 0$ 且 $d[i+1] == 0$ ： 转移到$dp[i+1][0]$和$dp[i+1][1]$
- $dir == 0$ 且 $d[i+1] == 1$ ： 转移到$dp[i+1][1]$
- $dir == 1$ 且 $d[i] == 0$ 且 $d[i+1] == 0$ ： 转移到$dp[i+1][1]$
- $dir == 1$ 且 $d[i] == 0$ 且 $d[i+1] == 1$ ： 转移到$dp[i+1][0]$
- $dir == 1$ 且 $d[i] == 1$ ： 没有满足条件的转移

特别注意

注意细节。

1.106 Codeforces 213E Two Permutations

通过情况

AC

关键词

字符串哈希 树状数组

题目大意

给定n的排列A和m的排列B。 我们要求所有的d值，使得：

- 定义长度为n的序列$C_i = A_i + d$
- 序列C是序列B的子序列（不是子串）

问有多少个符合条件的d值。 $1 \leq n \leq m \leq 2 * 10^5$

算法讨论

定义数字i在序列A中出现的位置为S_i； 数字i在序列B中出现的位置是T_i，

可以发现，判定一个d是否可行，等价于判定$T[d+1, d+n]$和S的相对大小关系是否相同。

所谓“序列A、B相对大小关系相同”，定义为， 条件“如果序列A的第i个元素在A中是第k大，那么序列B的第i个元素在序列B中也是第k大”对所有i都成立。

这种“相对大小关系”的问题，可以用变种KMP解决，但较为复杂与难以理解。 但有一个很简单的做法： 把序列当作字符串，计算hash值。

用两个树状数组进行维护，一个维护当前区间里出现的值，一个维护当前区间里出现的值的权值。

利用这两个树状数组可以支持$O(\log n)$在头部删除字符与在尾部添加字符。

总时间复杂度$O(n \log n)$

特别注意

无

1.107 Codeforces 217C Formurosa

通过情况

AC

关键词

推结论 DP

题目大意

有一长度为n的01序列A，你不知道每个数到底是0还是1， 你只知道它们不全是0，也不全是1.

给定一个表达式，只含与、或、异或、括号、0、1和问号。

你可以任意的把表达式中所有问号用A的某一项代替（可以用同一项代替多个问号），并得知运算结果是0还是1。

你可以执行上述测试任意多次。 问是否可能判断出A数列所有元素的值。

$n \leq 10^6$ 表达式长度不超过10^6

算法讨论

我们将证明以下条件是能判定出A数列的充分条件：

- 假设表达式是$f(s)$ 其中s是二进制串，表示各问号的取值。
- 只要存在一个s使得$f(s) \neq f(\sim s)$就可保证能判定出A数列（$\sim s$是s按位取反）。

证明如下：

- 我们枚举所有的位置对(x, y)，把s中的0都替换成A_x，1都替换成A_y。
- 如果$A_x = A_y$，那么$f(s)$与$f(\sim s)$结果显然相同。
- 如果$A_x \neq A_y$，那么$f(s)$与$f(\sim s)$结果必然不同。
- 题目保证了所有的A不全为0或1，因此，我们必定能找到一对数，一个是0一个是1，于是进而可以利用上述结论判定出具体哪个是0哪个是1。

• 得到了某个位置具体是哪个数后，就可以拿这个位置与其他所有位置进行判定，依据$f(s)$与$f(\sim s)$的关系推出其他所有数。

同时，这个条件也可以证明为必要条件。

我们考虑在表达式树上进行dp以求出是否存在s使得$f(s) \neq f(\sim s)$。 状态要记录3个，分别是：

- 是否存在s使得$f(s) = f(\sim s) = 0$
- 是否存在s使得$f(s) = f(\sim s) = 1$
- 是否存在s使得$f(s) \neq f(\sim s)$

转移方程有9个（3个符号*3个状态），但都比较显然，很容易就推出了。 总复杂度$O(N)$

特别注意

无

1.108 Codeforces 229E Gifts

通过情况

AC

关键词

dp

题目大意

甲有n类物品，第i类有L_i个，第i类的第j个物品的价值是$w_{i,j}$，同类物品价值两两不同。乙可以给甲提供一个长度为n的序列A，且$A_i \leq L_i$，所有A_i的和恰好是给定的m。 然后，甲将从第i类物品中等概率选择A_i个给乙。

乙想要价值最大的m件物品，于是他只会给甲提交那些有机会获得最高价值的m件物品的请求，而如果有多种方案都机会获得最高价的m件物品，乙会等概率随机选一种。

问乙真的获得了最大价值的m件物品的几率。

$\sum L_i \leq 1000 \quad m \leq 1000$

算法讨论

我们不妨设价值第m高的物品的价格是V。 那么，所有价格严格大于V的物品一定会被选中，设第i类有d_i件物品价格严格大于V。

我们不妨设价格等于V的物品共有S件，而选完了所有价格大于V的物品后，乙还可以选择T个物品。 那么，乙会等概率的从这S个物品中选T个提交给甲。 于是，乙获得价值最高的m件物品的几率就是『所有C_S^T种选择的成功几率的和』除以C_S^T。

我们用$dp[i][j]$表示已经考虑了前$i-1$种物品，现在价格等于V的物品还可以选j件时的成功率总和。 转移如下：

- 如果第i类物品不包含价格V的物品，那么乙只会把d_i提交给甲。 $dp[i][j] = \frac{dp[i-1][j]}{C_{L_i}^{d_i}}$

• 如果第i类物品包含价格V的物品，那么乙既可能提交d_i给甲，也可能提交d_i+1给A，因此$dp[i][j]=\frac{dp[i-1][j]}{C_{L_i}^{d_i}}+\frac{dp[i-1][j-1]}{C_{L_i}^{d_i+1}}$
最后$\frac{dp[n][m]}{C_S^T}$就是答案。 时间复杂度$O(nm)$

特别注意

注意精度问题，注意用科学计数法输出结果。

2 Special Thanks

• 首先，我感谢党，感谢祖国，感谢人民.....

• 然后，感谢龙浩民同学在LaTeX方面给予我的大量帮助，让我能用LaTeX写出这篇题解。

• 最后，感谢亲爱的你，以顽强的毅力，看到了这里。 ^_^