A.双剑合并

题意

n把A剑m把B剑，每一把剑都有一个值x，让A和B各自中的某一把剑的值异或，使得异或的值最大，输出此值。n和m不超过1e6,x不超过1e9

分析

1.根据这题范围肯定需要O(n)级别算法，基本上就是固定A，然后B中的每一个值必须要在常数时间内找到最大对应的匹配的A的值。从这个限制描述，隐隐约约能够感觉到要用一个类似于字典树的东西。同时，基于一种贪心策略，尽可能的要让我合并后的值的最高位为1。所以，我们可以以最高位为根建一个深度为31的01二叉树，首先让A中的值一个一个插进去，并在相应节点标记为可走。那么B的某一个值只需要从根节点开始走，尽可能的走和自己不同的位的路(比如当前B的位是0,那么如果有1我就走1否则再走0)，这样能够保证走过的一定是最优的。

2.这题没啥可错的，做这题之前读了一片论文，所以很有帮助：浅谈信息学竞赛中的0和1by武森

思考

1.这题写的比较撮，每一个都特判了一下，后面看到汝佳的伸展树感觉写法挺好的。ch[2],0表示左儿子，1表示右儿子，值为-1表示不存在。这样的话就不用讨论当前是0还是1了，用一个变量o表示x当前的01位，ch[o]如果不存在就!o就好

2.以后遇到两个数值变成最大，那么优先考虑高位。同时异或的性质比较好，可以预处理后很快求出i项和j项的异或和，同时还有一个特点sum[i]^sum[j]=0 的话表明他们两个相等同时还表明第i项到第j项异或和是0

B.单词替换

题意

给出一个只包含小写字母的s字符串，和单词A,B。把s中的所有A替换成B，长度都小于5e6。
PS:这题不需要考虑后效性

分析

1.这题就是最裸的KMP，当s中匹配到A的时候把匹配到A的首位置标记一下，输出的时候直接跳过lenA的长度同时输出B就好

思考

1.用KMP的时候一定要统一模板，以后就决定用f[0] = 0, f[1] = 0,这样的方式了，不需要纠结，也就不会有分歧

C 01的时间

题意

给你一个数x，让你找到x的倍数，同时这个倍数只能包含01数字(十进制)，输出最小的符合条件的

分析

1.这题本意是想让我们用搜索的，但写过数位dp后对这题第一反应就是可以dp啊。 dp[i][j][o]表示当前i长度余j并且第i位取o可不可达，同理dp[i][j][1]。

首先预处理出当第i位是1时对x取模的余数状态转移的话 *dp[i][j][0] |= dp[i-1][j][0] || dp[i-1][j][1]; dp[i][j][1] |= dp[i-1][(j-base[j]+x)%x][0] || dp[i-1][(j-base[j]+x)%x][1];*

2.全部切记初始化0,dp[0][0][0] = 1;

思考

1.这里刚开始我把dp所存的值弄混淆了，存了当前的最小值是多少，同时还开了一个辅助flag作

为标记能不能可达。但是，后面想一想，完全没必要。因为当状态转移好之后，第一个首位是1并且取余是0的状态就是最小的(这个可以反向想一下)。找到最小的状态之后，剩下的就是往回找，看前一个是由谁转移来的。

2.前面的转移式是最基本的转移，但实际转移中为了回溯更方便可以dp记录上一位的取值情况，不可到达初始为-1. 当有两种可选的状态的时候选较小的。

3.我曾经错误以为：上述过程有一个性质就是，对于所找到的第一个符合条件首位是1的数字，他的之前所有状态都只有一个前述状态。为什么？假如他可以由两个状态转移y,z转移过来，那么(y-z)%x==0,那么就会出现一个比当前更小的符合条件的。但是后面想一想（y-z）可能不符合只含01这个条件。（但是如果没有这个限制的话，这就是一个可以利用的性质）

E.RunningPhoton's Nightmare

题意

一个人站在S这个起始位置，想要走到一个迷宫出口E，但是现在这个人身上有一个炸弹在k时间内就会爆炸，同时到达一些R位置能够让炸弹计时器清零，问能否出迷宫，能并输出最短时间,其中迷宫大小不超过600*600，1<=k<=600，最多有100个清零器

分析

1.在S起点距离k的位置内的所有的都是可达并且最优时间等于最短距离。然后再由这些已经确定是最短的并且炸弹时间已被清零的点来更新周围距离k的点（如果炸弹时间不清零，那么这个点所能够到达的最远距离只能够在S的范围之内...）,然后这就好比是dijstra算法，只是多了一个限制（只能由R点更新周围点）。

2.刚开始拿到这一题想的是从起点开始暴搜（受到华为那个寻路比赛影响），状态包含当前已经走过的时间和炸弹剩余的时间以及当前位置，但是这样的话什么样的状态才是最优的就不好确定了，如果k小的话倒是能够对每个位置的每一个炸弹时间k记录一个最优时间。但是k比较大...

思考

1.上述算法最终只有S，E，R这三个点起作用了，那么现在就只看这三个点。如果这三类点距离小于k那么表示可以联通，也就是在原图的基础上面重新建了一个图。那就无脑跑100多次bfs求最短路，如果三类点距离d小于k则他们建一个d距离的边否则不能直接相连。

2.为了代码美观好写，用bfs求最短路的时候传入距离数组指针。

F 表达式

题意

最快多少次才能够最快的得到x^k，可以从已经得到的数中乘除，最初只有x，比如 x^{31},他最快可以通过通过6次操作

分析

1.为了能够知道当前搜索的解x是最优解，那么我能够做的就只有把比x小的解全部枚举一遍。所以就用到迭代加深搜索。

2.刚拿到这题，有很多种贪心策略，可惜后面发现都不对。然后试着用最短路的姿势来写，不过问题是你怎么才能确认这是最短路。这个搜索的空间无穷大，所以一直搜下去永远不能够确定。但这个就告诉我们不能把搜索空间弄的太深因为这样容易在某一个解空间里面无限搜下去，同时我们还要能确保当前搜到某一个解一定是最优解。由此，迭代加深搜索就凸现出来了

思考

1.对这类搜索复杂度一直很迷。而且有一个细节问题一直不能理解，为什么搜的时候一定要从当前最新的值a和前面的值进行运算得到新的值，而不能是以前的值b和c得出d，然后d和a再得出我想要的值呢？

2.搜的时候可以保留记录。以后对这类迷糊不确定上界的搜索首先就想到迭代加深吧！！！

G 神舟的宝藏

题意

让你找这样一个密码：

- 密码是一个C进制的数；
- 密码由给定的M个数字构成；
- 密码是一个给定N的十进制整数的正整数倍；
- 如果存在多个满足上面所有条件的数字，值最小数字就是密码；
- 系统输入上限为500位数字，如果求出的密码长度大于500位，则也不能打开房门。

其中C，M不超过16，N不超过5000

分析

1. 这题相当于01的时间的**进阶版**。写起来就很简单了：我们预处理的时候要以C进制为基础预处理第i为是M个数字中的某一个是对N取模的余数是多少。dp[i][j]表示长度为i，对N余数是j这样状态第i位是哪一个数字
2. 这题和之前的有两个不一样是

- 我不确定0有没有，如果有0的话我要特殊判断当前是0.状态转移的时候从最小数值开始转移**(当余数j是0的时候不包含0，因为这个时候如果能用其他数值更新那么后面就不需要再添加数字了，如果用0更新的话就会出现前导0还需要继续添加位数这样就不是最小的)**，就能够保证后面得到相同状态的时候值最小。
- 状态使用dp[i][j]而不像C题那样使用dp[i][j][k]多出来的K表示当前位取第K个数字，这样的添加了状态数和转移数，使得转移复杂度变得高很多，内存也不划算

思考

1. 像上面所说的那样从最小数字更新，这样一旦更新了，我就不准二次更新了，不需要比较值了，这样dp就可以“安心”记录上一个状态的值了
2. 这题容易处理错0这个数字，如果更新的时候把0放在最先，那么dp[i][0]永远都只会被0更新，而这是非法的（就好比是00000000...）。如果不把0放到最前面，那么处理余数非0的状态时候明可以用0来更新，而被用非0数字更新这样会导致所求的不是最小的数值。

H. DNA序列

题意

- 就是让你找这样一个字符串，让所给的串是目标串的子串，那么目标串最短应该是多少（最多8个串，每个串长度最大为5）

分析

1. 这个题我只用了迭代加深搜索，但是没剪枝过了，数据没卡我（AAAA GGGG CCCCC TTTTT就T了）。后面想了一下状态压缩记忆化搜索是可以在有限时间搜索出来的。用8个六进制的数字表示每一个串的当前位置。然后这个状态只有四种转移方向ATGC，然后从最终状态往前记忆化搜索就好了。
2. 搜索一直没想出来怎么剪枝...,所以就使用迭代加深思想，然后暴力的枚举8个队列串首字符，比如枚举当前字符为A，那么谁队首是A的就POP出来，如果没有队首是A那就没必要搜索了。

思考

1. 没啥坑点，就是不好好剪枝你会T在上面那个样例
2. 在用状态压缩转移的时候，要用记忆化从最后搜，因为最初状态往后转移的话，初状态不好枚举。

I. 小冰和小娜

题意

- 这题就是一个车有方向的放在某一个位置，现在转向和前进都会消耗一点时间，问你从起点到达终点的最短时间

分析

1. 这题就是我E题说到的，带附加属性的最短路，这个附加属性范围大小是4（4个方向）。那么直接在原先最短路的基础上面增加一维方向就好，然后就从初状态跑最短路就好。
2. 当时写的太挫了，wa了几发，太粗心了

思考

1. 没啥坑点
2. 在状态转移的时候使用dx[i],dy[i],dd[i].这样三种操作弄成一个循环了简洁不出错

J. TooEasy Or TooDifficult

题意

- 给你定义两类值，P[i]表示以第i个字符为中心的最长回文串长度，MZ和max{P[i]}有关，FJD为连续若干个P[i]异或和的最大值。问你MZ和FJD谁大

分析

1. 解决MZ就是最裸的manacher算法，FJD则利用到一个性质sum[j]^sum[i] = p[i]…P[j];这样转换之后就是求最大的sum[j]sum[i],这个问题在问题A中谈论过了，就是建立01二叉树，然后把sum[i]全部插入，再全部走一遍
2. 当时生成插入01树的时候写挫了，一直死循环，debug半天

思考

1. 没啥坑点
2. 常规题

K. 奶牛合影

题意

- 每个牛有一瑕疵值，我想再平移之后，得到一种排序他字典序最小，也就是最裸的最小表示法

分析

1. 直接用模板
2. 只要模板没有写错，几乎不会错

思考

1. 没坑点
2. 这题其实还可以用后缀数组写的，不过效率不划算

L. 奶牛序列

题意

- 约翰刚帮奶牛们拍完照，拿着合影的他，看着奶牛队列，又莫名想到了一个字符串问题：我们将n头奶牛的队列看成一个长为n的字符串S，让Ti表示从第i的字符开始的后缀。求：

- 其中，len(a)表示字符串a的长度，lcp(a,b)表示字符串a和字符串b的最长公共前缀，输入字符串长度不超过5e5

分析

1.

- 首先前面的len(Ti)和len(Tj)可以提取出来一步算出来，剩下主要就是求 lcp(Ti,Tj)的和，在后缀数组中求两个后缀的最长前缀长度，就是求各自对应h[i]数组间的最小值。由于这里i和j把所有的一对Ti和Tj组合都取遍，所以这里可以转换成求所有h[i]和h[j]数组中间的最小值的和的两倍

- 那么问题转换成求所有h[i]和h[j]数组中间的最小值的和，，如何快速的求出所有和呢，因为枚举会T。这里用到nlogn求最长公共子序列的思想。每一次我只求所有h[j]和h[i]之间的最小值的和(其中j<i).

- 然后不断的放入h[i],h[i]放入到第一个大于等于他的位置并把当前位置的值更新为 h[i]（因为所有大于h[i]的h[j]的值在和h[i]取最小之后多出来的那部分无效了，即使是在求i之后h[k]的所有lcp和也一样，由于存在h[i],那么h[j]在h[i]的左边并大于h[i]的部分无效，所以就直接用h[i]代替了h[j]）.同在在那个位置记录下最新的位置信息pos[k] = i。同时用一个数组sum[i]记录到位置i的lcp的和。

- 如何计算sum（lcp(Tj,Ti)）j<i？，就拿h[i]插入后举例子，他的前一个h[i-1]一定小于h[i],那么那个位置的sum[i-1]就不需要更新了，直接拿来用。pos[i-1]和pos[i]之间的值就全部都是h[i]了，然后再更新sum[i].

2. 好了经过上面很长的分析之后，我忘记sum(lcp)*2了

思考

1. 这题想到就不难了，当然实现的时候还是要想半天的

2. 求最长公共子序列这种nlogn的思想很实用。

M. 奶牛硬盘

题意

- 硬盘的制造商认为"一千"是1000，但是操作系统认为"一千"是1024.单位分别有“B”, “KB”, “MB”, “GB”, “TB”, “PB”, “EB”, “ZB”, “YB”

分析

1. 很简单的一道题，只和单位有关系和前面数值没关系，先求出保存率，然后1-保存率，base[0]= 1, base[1] = （1000）/1024;...

思考

1. 我在想oMB的丢失百分比是多少？？？我反正没考虑直接过了

N. 奶牛情书

题意

- 一个奶牛知道一些单词，现在有一个长度n的文本，问你这个文本至少包含一个奶牛会的单词的方案数。

分析

1.

- 正向算至少包含一个太麻烦了，所以就逆向算，算出总的(快速幂取模)，再算出一个都不包含的情况。而怎么才能算出一个都不包含的情况呢。这个就涉及了自动机套dp。

- 方案数用到dp这个都能够想到，可是字符串如何用状态表示呢。这个就要用到前缀了，把每一个前缀看做一个状态，那么添加一个字符之后所形成的前缀成为了另外一种状态（这里需要注意的是空串也是一种特殊的前缀）。

- 但是怎么很快得到所有的前缀并能得到他们的转移方向呢，很简单，那就是ac自动机。他把每一个字符串的前缀都体现出来了，ac自动机的节点数就是这里的状态数，ac自动机另外也可以叫做**有限状态机**，状态就这样来了

2.

- 在禁止一些字符串时，少考虑了把包含这个字符串的其他前缀串都禁止了。比如aba和ccabac，我们现在要把这两个串禁止用，但是单禁止这两个串还不行，因为我们的一个前

缀状态是ccaba,ccaba就包含了禁止串，所以我们要这样禁止一个串：如果他的失配后指向的串被禁止了那么当前的串也要禁止。 if(cnt[fail[u]]) cnt[u] = 1;

思考

1. 只要懂了原理就很简单了，没有坑点
2. 这个类型的ac自动机加dp帮我更好理解自动机。之前对于ac自动机的根节点和fail数组并不是很理解，现在想就是root节点就是空节点。fail指针就相当于在这个节点添加一个字符匹配失配后我还能够匹配哪里。