2020秋季周测2-数论/组合 题解

题解里啰嗦话比较多……不过换做是我看题解，可能也希望看到啰嗦点的题解。

A - Ayaya~ https://nanti.jisuanke.com/t/42395

题意：给定一个正整数2<=N<=1e9, 寻找最小的k值，使得：

对{1, 2, ..., n}的任意模为k的子集T，存在T中两个相异元素，其中一者为另一者的因数

分析：乍一看逻辑很绕，“对任意子集"难以处理，考虑否命题：

存在一个模为k的子集，使得T中任意两个不同元素都没有倍数关系

于是立刻能够构造出一个反例：

当n=2m时，{m+1, m+2, ..., 2m} = A, |A|= m = (n+1) / 2

当n=2m+1时，{m+1,m+2,...,2m+1} = B, |B| = m+1 = (n+1) / 2

于是必有k > (n + 1) / 2

一般在赛场上，到这时候你就该大胆猜想k = (n + 1) / 2 + 1然后先交一发了

（事实上，这就是正解）

如果你关心证明的话，现在就来证一波。

思考为什么答案与2有关，我们可能会感觉到，“2”是最划算的整除关系，

划算到我们不需要在整除关系中考虑除2以外的其他素因子。

对于任意x，提出所有因子2，得到x=(2^k)*y, y为奇数。

对于任意奇数y，我们可以得到一条“数链”：y，2y, 4y, 8y, ...同一链上任意两数都有整除关系

而这样的数链一共只有(n + 1) / 2条（因为y是奇数），由抽屉原理，结论得证。

难度：Easy ~ Hard，取决于胆有多大

证明确实不太好想，如果你觉得很好想的话，当我没说

B - Bad Apple!! http://acm.hdu.edu.cn/showproblem.php?pid=5950

$$\text{递推公式：} A_i = A_{i-1} + A_{i-2} + i^4$$

$$\text{新状态怎样得到：} A_{i+1} = A_i + A_{i-1} + (i+1)^4 = A_i + A_{i-1} + \sum_{k=0}^{4} C(4,k) * i^k$$

$$(i+1)^4 = \sum_{k=0}^{4} C(4,k) * i^k$$

$$\text{于是，状态} St_i = (A_i, A_{i-1}, i^4, i^3, i^2, i, 1)$$

转移矩阵详见标程

关于模数：2147493647，2^31左右，平方并不会爆long long，请安心取模

难度：Easy，只要你理解了矩阵快速幂（我当年就没理解，所以当年没做出还真是抱歉了）

C - Clownpiece http://acm.hdu.edu.cn/showproblem.php?pid=6908

模数为2^64, 类型一律声明为unsigned long long

unsigned long long的输出：使用 cout 即可（T不是很大）

n巨大

我们想快速求

$$\sum_{i=1}^{n}\sum_{d:d|i} d^k$$

还是那种变换枚举思路：我们交换和式

$$Ans = \sum_{d=1}^{n} d^k \sum_{d|i,i\le n} 1 = \sum_{d=1}^{n} d^k \lfloor n/d \rfloor$$

然后数论分块即可，注意到前缀和就是k次幂和，k<=3,直接套用公式即可

无法求逆元，使用分类讨论来防溢出，具体细节见标程

难度：Normal，和式变换、数论分块、取模处理，方法比较常规，但都有一定难度

D - Doremy

题意很简单，不再叙述

由题意立即得到一些性质：

```
1. i中1出现的位置集合一定包含于n中1出现的位置集合
```

2. i中某位为1，则该位在j中为0
3. 在i的最高位左方的j位全为0 （原谅我的语文水平）

123与原条件显然等价

举个例子：

n = 1 1 0 1 1 1 0 0 0 0 1 0 1 0 0

i = 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0

j = 0 0 0 0 0 1 1 1 1 0 0 1 0 1 1

确定了i的最高位之后，就不用再考虑i和j的大小关系了。

于是从i的最高位入手，对于最高位在从右开始数第m+1位的情况，

如果能知道此刻n在t右面有多少个1（记为k）

则更新答案

$$Ans = Ans + \sum_{i=0}^{k} C(k,i) * 2^{m-i} = Ans + 2^{m-k} * (1+2)^k$$

难度：Normal，略带思维

E - (ﾟ∀ﾟ)o彡゜えーりん！えーりん！ Uva - 11426

这是lrj蓝书上的一道例题，但的确经典好题，就挂了上来

题目的式子很别扭，换一下

$$G = \sum_{i=2}^{N} \sum_{j=1}^{i-1} gcd(i,j)$$

我们似乎可以先处理内层式子

$$\sum_{i=1}^{n-1} gcd(n,i)$$

固然不可能一个个算gcd,转换思维，先枚举gcd的值d，这里必有d | n,

接下来思考在[1, n - 1]内满足gcd(n, i) = d的 i 有多少个，这类似于欧拉函数的定义

方便起见，我们把区间扩成[1, n]，这样我们得到的答案就只是在原答案上加了一个gcd(n, n) = n而已

首先，若d = 1，答案是φ(n),

对于其他情况，一个很显然的性质是

$$\gcd(n,i) = d \text{等价于} gcd(n/d, i/d) = 1$$

于是

$$\sum_{i=1}^{n} [gcd(n,i) = d] = phi(n/d)$$

$[gcd(n,i) = d]$表示计数，即它为真时，值为1，否则为0

于是

$$f(n) = \sum_{i=1}^{n} gcd(n,i) = \sum_{d|n} d * phi(\frac{n}{d})$$

狄利克雷卷积!

而我们要求f(n)的前缀和，事实上可以先$O(nlgn)$地求出每一项，这里你要先线性筛欧拉函数

当然，因为x和phi(x)都是积性函数，所以你也可以直接对f(n)线性筛，推式子也不是很难

记得减掉1+2+...+n

难度：Hard

F - Flandre Codeforces - 451E

抽象出数学模型：求带约束不定方程

$$x_1 + x_2 + \ldots + x_n = s，0 \le x_i \le f_i, 1 \le i \le n$$

的解的个数， n <= 20

经典容斥题，考虑反面，集合A_k:

$$x_1 + x_2 + \ldots + x_n = s，x_k > f_k(\text{只约束}x_k)$$

的解的个数，显然若干个A_k的交都可以用隔板法求出

细节：0! = 1, C(0, 0) = 1，而当$i < j$时，$C(i, j) = 0$

难度：Normal（我想把数论和组合间隔开）

G - GHS（？）

无算法，就楞推

发现后面那一项是逗我们玩的

$$Ans = \sum_{a=2}^{n}\{(-a^2)(\frac{a^k - 1}{a - 1}) + a * k * (n + 1)\}, k = \lfloor log_a n \rfloor$$

n巨大

但我们发现，当$a > \sqrt{n}$时，k = 2, 用常用公式可以直接计算耶

对于剩下的部分，我们甚至可以遍历a，暴力计算出k，毕竟k最大也只有$log_2 n$

标程里时让a从大往小遍历去动态维护k的，希望没写错（过了倒是过了）

难度：？，一道不是很优雅的题，细节略繁琐，所以放在了后面

H - Hakurei

在一个m*n的棋盘上行走，可以任选一个格子作为起始点，每次可以走到上下左右相邻的格子上，且相同的格子不能经历两遍，还要满足一个要求：就是对于 已经走过的区域 上的任意两个格子，这两个格子之间的所有最短路径都一定完全位于 已经走过的区域 中。

最后这个人要将棋盘上所有格子全走一遍。

注意：
1、"路径"上顺次相邻的两个格子一定在棋盘上有公共边，即（1，1）-（1，2）-（2，2）是一条路径 而（1，1）-（2，2）不是路径。
2、两个格子间的最短路可能有多条。

计算所有可能的行走方案

乍一看非常恶心，没有任何入手方式，连起始点都是任意的，这人瞎走能走通吗?

还真是，他瞎走真走不通，看第一个样例，其实他已经死了。

我们非常希望找到此人行走的规律，直接找还真的可以找到，当时跟队友就这么做的，不过之后想到一个更直接的方法：

把思维逆转过来，既然它要把棋盘上所有格子都走一遍，我们就可以**倒着想**

他最后一步肯定是走到了某个角落，从这个时候，让时间倒流

他必须从角落开始，确定一个方向，然后沿着这个方向一直走到头，到头后必然顺势再走一步，拐到剩下的矩形的角落

然后就递归了。。但这题数据规模不允许直接写递归

（这样的话你会发现此人行走的规律就是在卷着一个矩形走，不过这已经不重要了，倒着想更方便）

n行m列，先考虑n>=2,m>=2的情况

我们从任意一个角落（共4个）出发，每次可以选择走行或者走列，发现当我们已经走了n-1次行或者已经走了m-1次列时，剩下的行走方式就唯一确定了（单行或单列）

到这里，你大概已经悟了，答案就是4 * C(n-1 + m-1, n - 1)

如果你不保准，可以对小数据打表验证（小数据直接按上面说的递归来写）

不过还是简单地证一下

人为规定：他至多走n-1行和m-1列，也就是说，如果他已经走了n-1行，对于剩下的那一单行，我们都理解为他在走列，这样下来，他恰好走了n-1行和m-1列，且他的行走方式由行列的组合方式唯一确定

这样C(n-1 + m-1, n - 1)的公式就很明显了

注意对于n或m为1的情况，前面的系数不再是4，要特判

难度：Lunatic，如果你能独立地想到的话，很厉害

I 题 - Ibuki - 2020杭电多校第一场

用到的方法：Fibonacci公式+二次剩余（见ppt最后一页）

题目要我们从0C到NC求和，但是N巨大，你肯定不能老实巴交这么做

入手点：K很小，我们是否可以把式子转化为对 0<=t<=K求和？

开始愉快地推式子：

首先记x为模5的二次剩余，$invx$为x的逆元，$e1 = (1 + x) * inv(2), e2 = (1 - x) * inv(2)$

e1,e2是一个分式，一定要对二取逆元啊！！！！

然后

$$Ans = \sum_{i=0}^{N} (F_{iC})^K = \sum_{i=0}^{N} (e_1^{iC} - e_2^{iC})^K * (invx)^K$$

把K次幂用二项式展开，然后交换和式，你会发现内层可以先用等比数列求和处理

方便起见, 令

$$w_1 = e_1^C, w_2 = e_2^C$$

则：

$$Ans = (invx)^K \sum_{i=0}^{N} \sum_{t=0}^{K} C(K,t) * w_1^{it} * w_2^{i(K-t)} * (-1)^{K-t}$$

$$= \sum_{t=0}^{K} C(K,t) * (-1)^{K-t} \sum_{i=0}^{N} w_1^{it} * w_2^{i(K-t)}$$

对于内层和式，可以考虑等比数列求和：

首项：1，公比$q_t = w_1^t * w_2^{K-t}$，注意当公比模p为1时，要特判（因为无法取逆元）

到了这里，你可以枚举$0 \le t \le K$，每次用快速幂计算q_t, 然后再快速幂算等比数列求和

这样做会T，快速幂算得太多了

如何优化?

我们发现$q_{t+1} = q_t * w_1 * inv(w_2)$

于是我们只要先用一次快速幂搞到q_0,就可以$O(K)$地得到所有公比

不仅如此，我们观察等比数列求和得式子：

$\sum_{i=0}^{N} q_t^i = (q_t^{N+1} - 1) * inv(q_t - 1)$

我们可以用类似地方法递推得到所有q_t^{N+1}，也仅仅需要再多算一次快速幂

但是，要求每个$q_t - 1$的逆元，我们还是要算K次快速幂

我的确是这么做的，毕竟$log(p)$差不多是$log(N)$的1/2

这样写完之后跑了1700ms，还算卡过，但应该不是正解。

事件复杂度$O(KlgP)$,P是模数

//这是2020年杭电多校第一场的第五题，他题解说是$O(KlgN)$就能过，我也不知道是怎么过的，大概这就是卡常大师。

// 这题会给两份标程，一份是我的，一份是杭电多校原来的标程

难度：Lunatic（没想到那么多人都来做这题）（40多分钟能A掉实在tql）