信息学竞赛中的整数和多项式

毕克

清华大学　交叉信息研究院

2017 年 2 月 6 日

wwwwodddd 的前言

wwwwodddd 的前言

1. 春节刚过，给大家拜个晚年，祝大家春节快乐，晚年幸福。

wwwwodddd 的前言

1. 春节刚过，给大家拜个晚年，祝大家春节快乐，晚年幸福。
2. 我应该是目前讲课中最弱的，所以我选择了许多简单题。

wwwwodddd 的前言

1. 春节刚过，给大家拜个晚年，祝大家春节快乐，晚年幸福。
2. 我应该是目前讲课中最弱的，所以我选择了许多简单题。
3. 毕恭毕敬，克勤克俭。BIKE 也是今天我要讲的一个题目。

wwwwodddd 的前言

1. 春节刚过，给大家拜个晚年，祝大家春节快乐，晚年幸福。
2. 我应该是目前讲课中最弱的，所以我选择了许多简单题。
3. 毕恭毕敬，克勤克俭。BIKE 也是今天我要讲的一个题目。
4. 我对下发讲义做了较多修改，在此表示非常抱歉。

wwwwodddd 的前言

1. 春节刚过，给大家拜个晚年，祝大家春节快乐，晚年幸福。
2. 我应该是目前讲课中最弱的，所以我选择了许多简单题。
3. 毕恭毕敬，克勤克俭。BIKE 也是今天我要讲的一个题目。
4. 我对下发讲义做了较多修改，在此表示非常抱歉。
5. 

主要内容

1. 信息学竞赛中的整数。
2. 信息学竞赛中的多项式。
3. 信息学竞赛中的生成函数。
4. 信息学竞赛中的 Burnside 引理。

第一部分

- 相信大家不会一开始就需要重连。
- 又到了数学的季节。
- 第一部分是关于整数的题目，都非常简单。
- 

整数

乘法：同一数的若干次连加，我们可以使用乘法来计算。

$$\underbrace{a + a + a + \cdots + a}_{n} = a \times n$$

乘方：同一数的若干次连乘，我们可以使用乘方来计算。

$$\underbrace{b \times b \times b \times \cdots \times b}_{n} = b^n$$

快速幂

在计算 $a^n \bmod p$ 时，我们将 n 拆成若干个 2 的整数次幂。

比如说 $a^{21} = a^{16} \cdot a^4 \cdot a^1$。

其中 a^{2^k} 的计算，可以在 $O(\log n)$ 的时间内完成，

所以总时间复杂度是 $O(\log n)$ 的。

快速幂

在计算 $a^n \bmod p$ 时，我们将 n 拆成若干个 2 的整数次幂。
比如说 $a^{21} = a^{16} \cdot a^4 \cdot a^1$。
其中 a^{2^k} 的计算，可以在 $O(\log n)$ 的时间内完成，
所以总时间复杂度是 $O(\log n)$ 的。

这些就是我们计算的方法，现在你已经理解了计算背后的思想。
我们一起看一个易于理解的例子，来把我们所学到的用于实践！

俄罗斯套娃 (Matrjoschka)

输入 $n, k(0 \le n, k \le 10^9)$，我们有 $k+1$ 个集合 $S_0, \ldots, S_k$。

其中 $|S_0| = n$ 即 S_0 的大小为 n，并且有 $S_i \subseteq S_{i-1}(1 \le i \le k)$

即 S_i 是 S_{i-1} 的子集。

问 $(S_0, S_1, \ldots, S_k)$ 有多少种可能？结果模 10^9+7 输出。

俄罗斯套娃 (Matrjoschka)

输入 $n, k(0 \le n, k \le 10^9)$，我们有 $k+1$ 个集合 $S_0, \ldots, S_k$。

其中 $|S_0| = n$ 即 S_0 的大小为 n，并且有 $S_i \subseteq S_{i-1}(1 \le i \le k)$

即 S_i 是 S_{i-1} 的子集。

问 $(S_0, S_1, \ldots, S_k)$ 有多少种可能？结果模 10^9+7 输出。

比如对于 $n=2, k=2$ 我们不妨设 $S_0 = \{A, B\}$。

那么一共有 9 种可能

$(S_0, \{\}, \{\}), (S_0, \{A\}, \{\}), (S_0, \{B\}, \{\})$

$(S_0, \{A\}, \{A\}), (S_0, \{B\}, \{B\}), (S_0, \{A, B\}, \{\})$

$(S_0, \{A, B\}, \{A\}), (S_0, \{A, B\}, \{B\}), (S_0, \{A, B\}, \{A, B\})$

讨论

- 希望大家不仅通过猜测得到答案，
- 并且给出一个易于理解的解释。

解答

答案是 $(k+1)^n$

对于每个元素均有 $k+1$ 种可能，即在 $S_0,\ldots,S_i(0\le i\le k)$ 中。所有元素均不相关，所以根据乘法原理可知，答案是 $(k+1)^n$，最后用快速幂计算即可。

另一种解释，我们考虑 $k+1$ 进制下所有的 n 位数，允许有前导零。如果第 i 位是 j 那么说明第 i 个元素在 $S_0,\ldots,S_j$ 中出现了。显然这两者是一一对应的关系，即答案是 $(k+1)^n$。

朋友 (Freund)

B 君有 9 个朋友，名字分别是 A, B, C, D, E, F, G, H, I。

B 君会 n 种语言，这 9 个朋友会的语言都是这 n 种语言的子集。

我们用对应的字母表示这个人所会的语言的集合。

比如用 A 表示 A 会的语言，B 表示 B 会的语言……

$A, B, C, D, E, F, G, H, I$ 这些集合有一些关系如下

$A \subseteq B, B \subseteq C, D \subseteq E, E \subseteq F, G \subseteq H, H \subseteq I$

$A \subseteq E, E \subseteq C, D \subseteq G, G \subseteq F, G \subseteq B$

输入一个整数 $n (0 \leq n \leq 200)$，问这 9 个集合有多少种可能性。

（取模和不取模分别考虑）

样例

样例输入一：

2

样例输出一：

1024

样例输入二：

4

样例输出二：

1048576

讨论

讨论

- 有了上一题的经验，这一题就变得异常简单。

讨论

- 有了上一题的经验，这一题就变得异常简单。
- 一个经验丰富的选手可以直接看出答案是 32^n。

讨论

- 有了上一题的经验，这一题就变得异常简单。
- 一个经验丰富的选手可以直接看出答案是 32^n。
- 如果需要高精度，在一些平台上有更妙的做法。

解答

解答

- 首先我们注意到每种语言是无关的。

解答

- 首先我们注意到每种语言是无关的。
- 根据乘法原理，答案一定是 x^n 的形式。

解答

- 首先我们注意到每种语言是无关的。
- 根据乘法原理，答案一定是 x^n 的形式。
- 根据样例可知 $x = 32$，答案是 32^n。
  - 也可以自己写一个暴力试一试。
  - 为什么是 32？出题人（我）通过不断的调整得到的。

解答

- 首先我们注意到每种语言是无关的。
- 根据乘法原理，答案一定是 x^n 的形式。
- 根据样例可知 $x = 32$，答案是 32^n。
  - 也可以自己写一个暴力试一试。
  - 为什么是 32？出题人（我）通过不断的调整得到的。
- 如果需要高精度，在一些平台上有更妙的做法。
  - 这个题本身是出在 ACM 中的，所以可以使用 Java。
  - 在 Linux 等环境下，使用 `printf("%0.f", pow(32, n));` 也是可以正确输出结果的。
  - 只有 2 的整数次幂，在 double 范围内可以输出，最大 2^{1023}。

解答

- 首先我们注意到每种语言是无关的。
- 根据乘法原理，答案一定是 x^n 的形式。
- 根据样例可知 $x = 32$，答案是 32^n。
  - 也可以自己写一个暴力试一试。
  - 为什么是 32？出题人（我）通过不断的调整得到的。
- 如果需要高精度，在一些平台上有更妙的做法。
  - 这个题本身是出在 ACM 中的，所以可以使用 Java。
  - 在 Linux 等环境下，使用 `printf("%0.f", pow(32, n));` 也是可以正确输出结果的。
  - 只有 2 的整数次幂，在 double 范围内可以输出，最大 2^{1023}。
- 根据我的观察，很多人会猜测答案是 n^{10}。

子集 (Subsets)

输入 n 和 k。$(1 \leq n, k \leq 10^9)$

设 S 集合的大小为 n，考虑 S 的 $\frac{k(k+1)}{2}$ 个子集 $S_{i,j}$，将这些矩阵排成一个下三角矩阵的样子。

其中第一行为 $S_{1,1}$，第二行为 $S_{2,1}, S_{2,2}$，一直到第 k 行为 $S_{k,1}, S_{k,2}, \ldots, S_{k,k}$。

这些集合还满足对于在一行中左右相邻的两个集合，右侧是左侧的子集，即 $S_{i,j} \subseteq S_{i,j-1}$。

这些集合还满足对于在一列中上下相邻的两个集合，下方是上方的子集，即 $S_{i,j} \subseteq S_{i-1,j}$。

问对于 S 的这些子集，有多少可能的情况，结果模 $10^9 + 7$ 输出。

样例

样例输入：

2 2

样例输出：

16

对于 $k = 2$ 的情况，相当于 Matrjoschka 题中 $k = 3$ 的情况，所以答案是 $(3 + 1)^2 = 16$。

讨论

讨论

- 和上一题一样，答案是 x^n 的形式，其中 x 只和 k 有关。

解答

- 和上一题一样，答案是 x^n 的形式，其中 x 只和 k 有关。

解答

- 和上一题一样，答案是 x^n 的形式，其中 x 只和 k 有关。
- 通过猜测可知 $x = 2^k$
  - 相当于从左下角画一条分割线到对角线。
  - 这条分割线每步向上或向右，一共 k 步。
  - 总计方案数 2^k。

解答

- 和上一题一样，答案是 x^n 的形式，其中 x 只和 k 有关。
- 通过猜测可知 $x = 2^k$
  - 相当于从左下角画一条分割线到对角线。
  - 这条分割线每步向上或向右，一共 k 步。
  - 总计方案数 2^k。
- 最终答案 2^{kn}

斐波那契 (Fibonacci)

输入一个 n，求第 n 项 Fibonacci 数 f_n。

其中 f_i 的定义满足 $f_0 = 0, f_1 = 1, f_i = f_{i-1} + f_{i-2}$。

结果对 $p = 10^9 + 9$ 取模。

讨论 & 一种做法

- 这个就不需要讨论了，大家一定会矩阵乘法的做法。
- 大家考虑一下不用矩阵乘法的做法。
- 如果递推的阶数是 k，求第 n 项。
  - 矩阵乘法 $O(k^3 \log n)$
  - 暴力倍增多项式取模 $O(k^2 \log n)$。我在之后会介绍。
  - FFT 优化多项式取模 $O(k \log k \log n)$。
  - 在第一天的营员交流中，一位营员认为大家都会 $O(k \log k \log n)$ 的做法。

另一种做法

老司机们非常熟练，一定知道如下递归的做法。

注意到

$f_{2n-1} = f_n^2 + f_{n-1}^2$

$f_{2n} = (2f_{n-1} + f_n)f_n$

$f_{n+1} = f_n + f_{n-1}$

即我们可以利用 f_n, f_{n-1} 算出 f_{2n}, f_{2n-1} 或者 f_{n+1}, f_n。

也就是说如果知道 n 的答案，可以计算出 $2n$ 或者 $n+1$ 的答案，

初始情况我们知道 $n = 1$ 的答案 $f_1 = 1, f_0 = 0$，对于其他的数

我们可以分奇偶来递归处理。

第三种做法

考虑 $x^2 = 5 \pmod p$

我们求出一个解（通过暴力或者 Baby Step Giant Step 结合原根）

比如 $x = 383008016$

这时我们可以将 x 带入 Fibonacci 数的公式，并结合乘法逆元和快速幂，即可在不使用矩阵乘法的情况下计算出 Fibonacci 数。

更多的推广？

- 对于以 1 和 9 结尾的质数 p，我们可以对 5 开根号。即存在 $x^2 = 5 \pmod{p}$。
  - 这在 Codechef FN 中会用到。
- 对于一些质数 p，存在三次单位根 x。即 $x^3 = 1 \pmod{p}$
  - 这在之后的题目 Stein 会用到。
- 对于一些质数 p，存在四次单位根 x。即 $x^4 = 1 \pmod{p}$
- 对于一些质数 p，存在 $k(1 \le k \le 22)$ 次单位根 x。即 $x^k = 1 \pmod{p}$。
  - 这在之后的题目 Codechef BIKE 中会用到。
- 对于一些质数 p，存在 2^k 次单位根 ω。即 $\omega^{2^k} = 1 \pmod{p}$
  - 这在 FFT 数论变换中应用广泛。

亿兆京垓 (Radixphi)

设 p 是 $x^2 = x + 1$ 的大于 1 的根 $p = \frac{1+\sqrt{5}}{2}$。

输入一个正整数 n，你要求出一个有限的整数集合 S，满足

$$\sum_{x \in S} p^x = n$$

集合 S 中不能有两个数一样，不能有两个数相差 1。

可以证明这样的集合 S 存在且唯一。

输出这个集合。

$1 \leq n \leq 10^9$

样例

样例输入一：

3

样例输出一：

2 -2

样例输入二：

9

样例输出二：

4 1 -2 -4

讨论

讨论

- 贪心？

讨论

- 贪心？
- 倍增（或者其他什么），然后调整？

讨论

- 贪心？
- 倍增（或者其他什么），然后调整？
- 一种更妙的做法？

解答

- 直接从大到小贪心。
  - 完全正确！
  - 精度可能会有问题。
- 某种调整算法？
  - 假设 S 是多重集合，即可以包含重复元素。
  - 我们可以在 S 中删去 i 并加入 $i-1$ 和 $i-2$。
  - 或在 S 中加入 i 并删去 $i-1$ 和 $i-2$。
  - 会不会超时呢？

解答

某天我想到了一个很妙的做法：

```cpp
#include <cstdio>
const int N1 = 45, N2 = 90;
int a[123], c;
long long f[N2] = {1, 2};
long long n;
int main() {
    scanf("%lld", &n);
    for (int i = 2; i < N2; i++) {
        f[i] = f[i - 1] + f[i - 2];
    }
    n *= f[N1];
    for (int i = N2 - 1; i >= 0; i--) {
        if (n >= f[i]) {
            a[c++] = i;
            n -= f[i];
        }
    }
    for (int i = c - 1; i >= 0; i--) {
        printf("%d\n", a[i] - N1);
    }
    return 0;
}
```

生成变异的 Fibonacci 数 f_i，即从 $f_0 = 1, f_1 = 2$ 开始。

选取两个数 N_1, N_2 保证 S 集合中最小的数大于 $-N_1$，S 集合中最大的数小于 $N_2 - N_1$。

考虑 nf_{N_1} 的解，找到解

$$\sum_{x \in S} f_x = nf_{N_1}$$

将 S 中所有数字都减去 N_1 即是答案。

解答

证明大概是这样的：

我们注意到不仅

$$\sum_{x\in S} p^x = n$$

而且

$$\sum_{x\in S} q^x = n$$

其中 $q = 1 - p = -1/p = \frac{1-\sqrt{5}}{2}$。

p 和 q 是 $x^2 = x + 1$ 的两个根。

联系 Fibonacci 数 $f_n = \frac{p^n - q^n}{\sqrt{5}}$ 我们有

$$\sum_{x \in S} p^{x+N_1} = np^{N_1}$$

$$\sum_{x \in S} q^{x+N_1} = nq^{N_1}$$

两边作减法并同时除以 $\sqrt{5}$，即有

$$\sum_{x \in S} (p^{x+N_1} - q^{x+N_1})/\sqrt{5} = n(p^{N_1} - q^{N_1})/\sqrt{5}$$

$$\sum_{x \in S} f_{x+N_1} = nf_{N_1}$$

这时再进行贪心就不再有精度问题了。

我们还需要证明第一个等式，即在

$$\sum_{x \in S} p^x = n$$

的情况下有

$$\sum_{x \in S} q^x = n$$

这个结论显然对于一些实数 n 不成立的。

回想我们的调整做法，我们这么调整的基础是

$$x^i = x^{i-1} + x^{i-2}$$

但是这里 x 是什么并没有关系，所以这两个式子会同时成立。

Irrational base(PE558)

设 r 是 $x^3 = x^2 + 1$ 的实根。

我们注意到每一个正整数均可以写成若干个 r 的次幂和。

如果我们要求只能使用有限个次幂，并且任意两个次幂之间的差至少是 3，这样的表示方式是唯一的。

设 $w(n)$ 表示 n 在这种表示下，有多少项。

设 $S(m) = \sum_{j=1}^{m} w(j^2)$。

求 $S(5000000)$

比如

$$3 = r^{-10} + r^{-5} + r^{-1} + r^{2}$$

$$10 = r^{-10} + r^{-7} + r^{6}$$

所以我们有 $w(3) = 4, w(10) = 3$。

已知 $S(10) = 61, S(1000) = 19403$

但这些对于思考题目并没有帮助。

讨论

讨论

- 贪心?

讨论

- 贪心？
- 倍增（或者其他什么），然后调整？

讨论

- 贪心？
- 倍增（或者其他什么），然后调整？
- 一种更妙的做法？

解答

- 直接从大到小贪心。
  - 完全正确！
  - 精度可能会有问题。
- 某种调整算法？
  - 假设 S 是多重集合，即可以包含重复元素。
  - 我们可以在 S 中删去 i 并加入 $i-1$ 和 $i-3$。
  - 或在 S 中加入 i 并删去 $i-1$ 和 $i-3$。
  - 会不会超时呢？

解答

558.py

```python
f = [1, 2, 3]
for i in range(300):
    f.append(f[-1] + f[-3])

def F(n):
    n *= f[200];
    a = []
    for i in range(300)[::-1]:
        if n >= f[i]:
            n -= f[i]
            a.append(i)
    return len(a)

n = 5000000
s = 0
for i in xrange(1, n + 1):
    s += F(i * i)
    if i % 10000 == 0:
        print i

print s
```

第二部分

- 重连时间到！
- 为什么你会这么熟练啊！你究竟写过多少次快速幂！
- 这一部分主要是关于多项式的一些内容。
- 学习矩阵乘法，多项式运算，FFT，FWT 等内容，可以更好的帮助大家本部分内容。
- 

多项式

我想特别强调（循环）卷积的概念

- 基本形式，已知 A, B 他们的卷积 C 满足

$$C_i = \sum_j A_j B_{i-j}$$

  其中减法是在对数组长度取模意义下的减法。
- 暴力卷积是 $O(n^2)$，无法自己和自己卷积 t 次。
- 因此我们选一些点求值，运算，最后插值回多项式。
- 求值和插值暴力进行的话是 $O(n^2)$，根据卷积的不同情况，我们有不同的优化方法
- 对于一维或二维，使用 FFT 进行优化。
- 对于 k 维，每一维长度均为 2 或 3，使用 FWT 进行优化。

幂求和

B 君想到了一个有趣的题，输入 n 和 k，求 1 到 n 的 k 次幂和，即

$$\sum_{i=1}^{n} i^k$$

结果模 10^9+7 输出。

其中 n 满足 $1 \le n \le 10^9$，k 满足 $1 \le k \le 2000$。

杜老师觉得这个题太蠢了，于是将 k 修改为 $1 \le k \le 200000$。

解答

这个问题已经被杜瑜皓的《多项式及求和》详尽的讨论过了。
我只在此只想总结两种我认为比较简洁的做法。

- 差分和组合数，时间复杂度 $O(k^2)$。
- 拉格朗日插值法 $O(k\log k)$ 或 $O(k)$。

差分和组合数

考虑数列 $1^k, 2^k, \ldots, i^k$ 相邻两项做差之后，得到的数列的每项应该是一个 $k-1$ 次关于 i 的多项式。
再次相邻两项做差之后，得到的数列的每项应该是一个 $k-2$ 次关于 i 的多项式。
如此进行 k 次，得到的数列的每项应该是一个 0 次关于 i 的多项式，即常数数列。
再次相邻两项做差之后，一定会得到一个全是 0 的数列。
假设经过 i 次差分数列之后的数列第一项为 r_i 那么答案就是

$$\sum_{i=0}^{k} r_i \binom{n}{i+1}$$

举例来说，考虑数列

$1, 8, 27, 64, 125, 216, \ldots$

求差分可得

$7, 19, 37, 61, 91, \ldots$

$12, 18, 24, 30, \ldots$

$6, 6, 6, \ldots$

$0, 0, \ldots$

每个数列的第一项为 $1, 7, 12, 6$。所以最终的答案即为

$$\binom{n}{1} + 7\binom{n}{2} + 12\binom{n}{3} + 6\binom{n}{4} = \frac{n^2(n+1)^2}{4}$$

拉格朗日插值法

首先根据一些常识我们可以知道答案是一个 $d = k + 1$ 次的多项式，我们需要先通过暴力求出这个多项式在 $0, 1, \ldots, d$ 这 $d + 1$ 个点的值，再计算这个多项式在 n 这个点的值。

$$f(n) = \sum_{i=0}^{d} (-1)^{d-i} f(i) \frac{n(n-1)\cdots(n-d)}{(d-i)!i!(n-i)}$$

我们注意到 $f(x) = x^k$ 是一个积性函数，也就是说计算 1^k 到 k^k 只需要 $O(k)$ 的时间，即 $O(k/\log k)$ 个质数 x 的 $f(x)$，计算每个质数需要 $O(\log x)$ 的时间，其他的数通过积性函数计算。

然后计算上面的表达式只需要结合预处理即可以做到 $O(k)$。

差分法

差分法虽然不是最优的做法，但是他可以非常简单地处理任意多项式的求和，只要有前若干项的值即可。

一个使用的例子是 WC 2015 未来程序的第 8 个测试点。

A lagged Fibonacci sequence(PE258)

定义数列 g_n，如下：

$$g_i = \begin{cases} 1 & 0 \le i \le 1999 \\ g_{i-2000} + g_{i-1999} & i \ge 2000 \end{cases}$$

求 $n = 10^{18}$ 时的 g_n 结果对 20092010 取模。

讨论

- 直接暴力，时间复杂度 $O(n)$，你需要未来程序。
- 联想到一般的 Fibonacci 循环节长度是 $O(p)$ 的，我们可以试图找循环节，但是很遗憾会失败。
- 直接矩阵乘法，时间复杂度 $O(k^3 \log n)$，也不太行啊。
- 更喵的做法？

一个高端的做法

$O(k^2 \log n)$，可以在时限内出解。

简而言之，就是在模 20092010 的前提下，计算 $x^{10^{18}} \bmod (x^{2000} - x - 1)$ 的结果，并且将 $x = 1$ 带入求值，就是这题的答案。

其中多项式取模的部分可以有很多种实现方法，比如暴力倍增，或者是用 Sage 之类的语言。

解答

我在这个题目的讨论中看到了

About 5 lines of Sage:

Python

```
A = Integers(20092010)
R = PolynomialRing(A, 'x'); x = R.gen()
S = R.quotient(x^2000 - x - 1, 'a'); a = S.gen()
v = a^(10^18)
v.lift().subs(x=1)
```

That's about 0.12s of computing time on my 2GHz athlon64 machine.

大意就是在模 20092010 的前提下，计算 $x^{10^{18}} \bmod (x^{2000} - x - 1)$ 的结果，并且将 $x = 1$ 带入求值输出。

证明

首先我们陈述 Cayley–Hamilton 定理

考虑转移矩阵 M 的特征多项式 $p(x)$，我们有 $p(M)=0$。

在本题中，即有 $M^{2000}-M-I=0$。这个形式非常喵，他表示

$$M^{10^{18}}=f(M)(M^{2000}-M-I)+g(M)$$

其中 $f(M)$ 是一个关于 M 的多项式，相当于商。

$g(M)$ 是一个关于 M 小于 2000 次的多项式，相当于余数。

第一部分一定是 0，我们可以根据第二部分 $g(M)$ 和初始值计算出最终的答案。

特征多项式

我们设 A 为 $n \times n$ 的方阵，I 是 $n \times n$ 的单位矩阵，那么 A 的特征多项式就是

$$p(\lambda) = \det(\lambda I - A)$$

其中 det 表示行列式求值。

在本题中，特征多项式为 $x^2000 - x - 1$。

生成树甲 (KirchhoffA)

输入 n 个点 m 个无向边，可能有重边，求生成树个数。

$1 \le n \le 300, 1 \le m \le 10^5$

讨论

这个就不需要讨论了，大家一定会。

我们来复 (yu) 习 (xi) 一下基尔霍夫定理。

基尔霍夫定理

构造一个矩阵 A。

如果 $i = j$，那么 A_{ij} 为点 $i(j)$ 的度数。

如果 $i \neq j$，那么 A_{ij} 为 i 到 j 的边数的相反数。

最终得到的矩阵，删掉任意一行，任意一列之后的矩阵行列式求值，可得到原图中生成树的个数。

举个例子

考虑大小为 n 的完全图，我们来证明他的生成树个数为 n^{n-2}。

生成树乙 (KirchhoffB)

输入 n 个点 m 个无向边，可能有重边，无向边有红蓝两种颜色，求恰好包含 k 条红边的生成树个数。

$1 \le n \le 50, 1 \le m \le 10^5$

讨论

这个就不需要讨论了，一些人一定会。

我们来复 (yu) 习 (xi) 一下基尔霍夫定理的一个简单推广。

简单推广

我们设红边是 x，蓝边是 1，按照上题算矩阵并且行列式求值。
这样得到的值一定是一个多项式。
其中 x^k 次方的系数，表示恰好有 k 条红边的生成树个数。
但是多项式加减乘除都很不方便，怎么办呢？
我们先选 n 个位置求值，最后用这 n 个位置的值插出原多项式即可。

举个例子

考虑大小为 n 的完全图，每两个点之前有一条红边和一条蓝边。
我们可以很容易的求出对应的行列式的值为 $n^{n-2}(1+x)^{n-1}$。
也就是说含有 k 条边的生成树个数 $n^{n-2}\binom{n-1}{k}$。

生成树丙 (KirchhoffC)

输入 n 个点 m 个无向边，可能有重边，无向边有红黄蓝绿四种颜色，求红色不少于黄色，蓝色不少于绿色的生成树个数。

$1 \le n \le 20, 1 \le m \le 10^5$

讨论

这个就不需要讨论了，一些人一定会。

我们来复 (yu) 习 (xi) 一下基尔霍夫定理的一个中等难度的推广。

中等推广

我们设红边是 x，黄边是 x^{-1}，蓝边是 y，绿边是 y^{-1} 按照上题算矩阵并且行列式求值，算出所有 $x^i y^j (i,j \geq 0)$ 的系数和即可。当然这里有一点点小困难，就是次数变成负数如何处理，我们可以将红黄蓝绿分别设为 x^2y, y, xy^2, x，这样就不会遇到问题了。

Colorful World(PKU ACM)

给定 $Ap = q$ 中的 A 和 p，其中 A 是 $n \times n$ 的循环矩阵，p 和 q 是 $n \times 1$ 的矩阵。
所有数均是实数，求解 p。$(n \le 1024)$

循环矩阵即存在长度为 n 的数组 a，满足 $A_{ij} = a_{(i-j) \bmod p}$。

讨论

- 暴力的消元并无法通过。
- 循环矩阵有什么特殊性质？

解法

考虑三个多项式 $A(x)=\sum_{i=0}^{n-1}a_ix^i$ 和 $Q(x)=\sum_{i=0}^{n-1}p_ix^i$ 和 $Q(x)=\sum_{i=0}^{n-1}q_ix^i$。

注意到如果 $x^n=1$，我们有 $A(x)P(x)=Q(x)$。

而满足 $x^n=1$ 恰好是 n 个单位根，我们可以在 n 个单位根计算 $P(\omega_i)=Q(\omega_i)/A(\omega_i)$ 最后用暴力 DFT 算出 p_i 的值。

时间复杂度 $O(n^2)$。

其他

如果 n 是 2 的整数次幂，这个算法可以优化到 $O(n \log n)$。

单车 BIKE

输入 $n(n \le 22)$ 个点，$m(m \le 8000)$ 个边，每个边有两个长度 $f = f_i, r = r_i$。

问对于每个点 k，有多少条路径由 t 条边组成，从 k 开始，并且以 k 结束；

并且路径上所有边 f 的和 $\bmod n$ 为 i；

并且路径上所有边 r 的和 $\bmod (n-1)$ 为 j。

方案数 $\bmod 1163962801$ 输出。

解答

首先我们将一条边看成 $x^{r_i}y^{f_i}$，这样矩阵乘法之后。

$x^a y^b$ 的系数即为 f 长度和为 a，r 长度和为 b 的方案数。

现在我们需要让这个多项式运算，像循环卷积一样来进行运算。

也就是说不要有超过或等于 x^n 或 y^{n-1} 的项。

我们注意到 1163962801 是一个很特别的数字，这是一个质数，

这模 2 到 22 的任意一个数字都余 1。

也就是说 $x^n = 1 (2 \le n \le 22)$ 在对这个数取模的情况下有 n 个单位根。

我们可以直接带入 $x^n = 1$ 的 n 个根，计算出 $f(x_i)$ 和 $g(x_i)$，最后利用 $f(x)g(x)$ 在这 n 个根 x_i 上的值插值得到多项式。
这个过程类似于傅里叶变换。

解答

最终解法

- 枚举 $n(n-1)$ 个插值。
  - 计算出在此插值下的邻接矩阵。
  - 计算快速幂，统计结果。
- 利用统计结果，插值计算答案。

代码厨师 (DMCS)

用 $1\times1\times1\times2$ 的立方体，填满 $2\times2\times2\times n$ 的空间。
其中立方体是可以旋转的，也就是说他们可以被旋转成
$2\times1\times1\times1$ 或者 $1\times1\times2\times1$。
输入一个 $n(1\le n\le10^9)$，求方案数模 10^9+7 的结果。

解答

状态压缩 DP，然后一个大小为 2^{16} 的矩阵乘法是显而易见的。
然后我们优化这个矩阵乘法，去掉其中的 0 状态，和对称状态。
这样大概剩下几千个，我们可以预处理前几千项，然后计算线性递推。

具体来说，接口处我们用 1 表示切过一个 $1 \times 1 \times 1 \times 1 \times 2$ 的立方体，用 0 表示没有切过。这样在切口处，一定有偶数个 1。这可以剪枝掉一大部分。

类似的思路，我们可以对接口黑白染色，像国际象棋棋盘一样，这样在黑色区域内的 1 的个数必须和在白色区域内的 1 的个数相同。

这又可以剪枝掉一大部分。

还有一个情况需要考虑，就是在所有状态中，很多状态可以通过对称和旋转得到，对于这些状态我们只需要记录一次而不需要每次都记录。

这又可以剪枝掉一大部分。

如果我们的做法一样这里我们会得到一个 571 阶的矩阵。

我们需要考虑两个问题：

1. 已知初始值和转移矩阵，求递推公式？
   - 以此可以使用 $k^2 \log n$ 的优化。
2. 已知初始值和递推公式，求递推公式至少有几阶？
   - 举例，比如初始值是 $1, 1, 2, (3, 5, 8, 11, \ldots)$，递推公式是 $f_i = 2f_{i-2} + f_{i-3}$。
   - 递推公式可以只有 2 阶，也就是 $f_i = f_{i-1} + f_{i-2}$。

已知初始值和转移矩阵，求递推公式

假设初始值是 f，转移矩阵是 A。

那么对于 $f, Af, A^2, \dots, A^k f$ 这 $k+1$ 个向量来说

存在 $a_i(1 \le i \le k)$ 使得

$$A^k f = \sum_{1 \le i \le k} a_i A^{k-i} f$$

只考虑列向量的第一项

$$f_k = \sum_{1 \le i \le k} a_i f_{k-i}$$

即为递推数列。

已知初始值和递推公式，求递推公式至少有几阶

考虑如下矩阵

$$\begin{bmatrix} f_0 & f_1 & \cdots & f_k \\ f_1 & f_2 & \cdots & f_{k+1} \\ \vdots & \vdots & \ddots & \vdots \\ f_k & f_{k+1} & \cdots & f_{2k} \end{bmatrix}$$

这个矩阵的行列式一定是 0，他的秩就是递推数列的阶数。

通过计算他的秩，可以同时得到最简的递推公式表达。

解答

通过这个求秩的优化之后，可以得到一个 71 阶的线性递推。
并使用任意方法（矩阵乘法，或优化后的 $O(k^2 \log n)$ 的做法）
通过这个题目。

解答

在出这个题目之初我没有注意到最后一个求秩的优化，所以我误认为我化简出的一个 500 多阶的矩阵是最优解，出了一个愚蠢的数据范围。

超立方体 (CUBE)

一个 $n = 2^k (k \leq 20)$ 个点的无向图，标号 0 到 $2^n - 1$，如果 i 和 j 的二进制表示只差一个 1，那么他们之间有一条边。每个点有一个权值，每过一个周期，每个点的权值等于所有和他相邻的点的权值和。

输出 $t(t \leq 10^{18})$ 个周期后的每个点的权值和，模一个数 K 输出，这个数可能是质数也可能是合数。

讨论

- 这个题目已经出了快三年了。
- 已经从难题变成了模板题了。
- 大家有没有什么奇思妙想？

解答

最基本的想法是 $O(n^3 \log t)$ 的矩阵乘法。

然后非常容易想到，这个图是完全对称的，从一个点看，其他的点可以被分成 $k+1$ 类别。

这样就是一个 $O(k^3 \log t)$ 的动态规划，一个 $O(n^2)$ 的暴力即可。

然而这么想的话……我……就做不出了。

解答

注意到，我们可以将这个题目看成一个 k 维，每一维长度是 2 的循环卷积。

然后我们可以仿照二维 FFT 进行插值。

最后的最后，一点点小 Trick。取模的数不是质数怎么办？

我们注意到只有最后一步，需要除以 n 其他地方均不需要处罚。所以在全部计算过程中，我们对 np 取模即可，这样最后一定可以整除。

第三部分

- 嘀嘀，第三部分。
- 为什么会变成这样……第一次学会了整数。学会了多项式。两件快乐事情重合在一起。而这两份快乐，又给我带来更多的快乐。得到的，本该是像梦境一般幸福的时间……但是，为什么，会变成这样呢？
- 这部分讲解了一个我学 OI 的时候觉得没有用的东西，生成函数。
- 

普通型生成函数

序列 $\{a_i\}$ 的生成函数就是

$$A(x) = \sum_{i=0}^{\infty} a_i x^i$$

已知 $\{a_i\}, \{b_i\}$ 的生成函数分别是 $A(x), B(x)$。
那么 $\{a_i \pm b_i\}$ 的生成函数是 $A(x) \pm B(x)$。
值得注意的是数列 $\{a_i\}, \{b_i\}$ 的卷积的生成函数，恰好是 $C(x) = A(x)B(x)$。

$$c_i = \sum_j a_j b_{i-j}$$

关键公式

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + \cdots = \sum_{i \geq 0} x^i$$

以此推出

$$\frac{1}{1-x^2} = 1 + x^2 + x^4 + x^6 + \cdots = \sum_{i \geq 0} x^{2i}$$

推广的二项式定理

$$(1+x)^n = \binom{n}{0}x^0 + \binom{n}{1}x^1 + \binom{n}{2}x^2 + \cdots = \sum_{i \geq 0} \binom{n}{i} x^i$$

例子

Fibonacci 数的生成函数

$$F(x) = xF(x) + x^2F(x) + f_0 + (f_1 - f_0)x$$

带入 $f_0 = 0, f_1 = 1$ 我们有

$$F(x) = \frac{x}{1 - x - x^2}$$

如果想继续计算通项，我们需要解方程得到

$$\frac{x}{1 - x - x^2} = \frac{a}{1 - \alpha x} + \frac{b}{1 - \beta x}$$

例子

Catalan 数的生成函数

$$C(x) = C(x)^2 + 1$$

解二次方程得

$$C(x) = \frac{1 - \sqrt{1-4x}}{2x}$$

如果想继续计算通项，我们需要用推广的二项式定理展开 $\sqrt{1-4x}$。

指数型生成函数

序列 $\{a_i\}$ 的生成函数就是

$$A(x) = \sum_{i=0}^{\infty} \frac{a_i x^i}{i!}$$

已知 $\{a_i\}, \{b_i\}$ 的生成函数分别是 $A(x), B(x)$。
那么 $\{a_i \pm b_i\}$ 的生成函数是 $A(x) \pm B(x)$。
值得注意的是数列 $\{a_i\}, \{b_i\}$ 的卷积的生成函数，恰好是 $C(x) = A(x)B(x)$。

$$c_i = \sum_j \binom{i}{j} a_j b_{i-j}$$

关键公式

$$e^x = 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \cdots = \sum_{i \geq 0} \frac{x^i}{i!}$$

以此推出

$$e^{-x} = 1 - x + \frac{1}{2}x^2 - \frac{1}{6}x^3 + \cdots = \sum_{i \geq 0} \frac{(-x)^i}{i!}$$

$$\frac{e^x + e^{-x}}{2} = 1 + \frac{1}{2}x^2 + \frac{1}{24}x^4 + \cdots = \sum_{i \geq 0} \frac{x^{2i}}{(2i)!}$$

这些就是我们计算的方法，现在你已经理解了计算背后的思想。我们一起看一个易于理解的例子，来把我们所学到的用于实践！

平方根 (Quadratwurzel)

B 君发现一个有趣的数列 $\{a_i\}$，数列的下标从 0 开始。
满足对于任意 n，均有

$$\sum_{0\leq i\leq n} a_i a_{n-i} = 1$$

B 君发现如果令 $a_0 = 1$ 的话，那么这个数列的解是唯一的。
特别的，虽然 a_n 是实数，但是 $a_n 4^n$ 一定是整数。

讨论

讨论

- 一个经验丰富的选手可以直接看出答案是 $\binom{2n}{n}$。

解答

$$\sum_{0\le i\le n} a_i a_{n-i} = 1$$

表示自己和自己的卷积是一个全是 1 的数列。

考虑数列 $\{a_i\}$ 的生成函数 $A(x)$，满足

$$A(x)^2 = \sum x^i = \frac{1}{1-x}$$

$$A(x) = \frac{1}{\sqrt{1-x}}$$

可得

$$a_i = \frac{\binom{2n}{n}}{4^n}$$

解答

剩下的部分就显而易见了。

对于输入的 n，我们需要输出 $4^n a_n = \binom{2n}{n}$。

也就是说我们需要再 $O(1)$ 的时间内，

计算一个组合数对质数取模的结果。

相信这个对大家来说很简单，我就不再赘述了。

Platonic Dice(PE389)

扔一个 4 面的骰子，计结果为 T。

扔 T 个 6 面的骰子，计所有面的和结果为 C。

扔 C 个 8 面的骰子，计所有面的和结果为 O。

扔 O 个 12 面的骰子，计所有面的和结果为 D。

扔 D 个 20 面的骰子，计所有面的和结果为 I。

求 I 的方差。

讨论

- （这么简单的题没必要讲吧，直接暴力出来最终结果的分布，不是想计算什么就计算什么吗？）
- 我最近听说了一种很喵的做法，虽然对很多人来说很显然。

解答

计算 $E = (4+1)(6+1)(8+1)(12+1)(20+1)/32 = 2687.34375$

方差即是 $(E^2 - E)/3 = 2406376.3623$。

答案与骰子顺序无关！

简单题 (Einfach)

一个数组长度为 n，我们要在每个位置中填一个 1 到 6 的数字。
但是其中 5 和 6 在整个数组中，必须出现偶数次。
问有多少种合法的方案？结果对 $p = 10^9 + 9$ 取模。
$1 \le n \le 10^9$

讨论

这个就不需要讨论了，大家一定会。

为了方便后文讲解，这里我们介绍一种使用生成函数的做法。

解法

设 $A(x) = \frac{1}{0!} + \frac{x}{1!} + \frac{x^2}{2!} + \cdots = e^x$

设 $B(x) = \frac{1}{0!} + \frac{x^2}{2!} + \cdots = \frac{e^x+e^{-x}}{2}$

我们需要求 $A(x)^4 B(x)^2$ 中 x^n 的系数，直接展开即可。

解法

$$C(x) = A(x)^4 B(x)^2 = \frac{e^{6x}}{4} + \frac{e^{4x}}{2} + \frac{e^{2x}}{4}$$

展开之后，可得 $c_i = \frac{6^n+2\times4^n+2^n}{4}$。

中等题 (Medium)

一个数组长度为 n，我们要在每个位置中填入 1 到 $a+b+c$ 的一个数字。

但是其中 $a+1$ 到 $a+b$ 的 b 个数字在整个数组中，必须出现偶数次。

但是其中 $a+b+1$ 到 $a+b+c$ 的 c 个数字在整个数组中，必须出现奇数次。

问有多少种合法的方案？结果对 $p=10^9+9$ 取模。

$1 \leq n \leq 10^9, 1 \leq a+b+c \leq 100$

讨论

仿照上一题的做法

解法

设 $A(x) = \frac{1}{0!} + \frac{x}{1!} + \frac{x^2}{2!} + \cdots = e^x$

设 $B(x) = \frac{1}{0!} + \frac{x^2}{2!} + \cdots = \frac{e^x + e^{-x}}{2}$

设 $C(x) = \frac{x}{1!} + \frac{x^3}{3!} + \cdots = \frac{e^x - e^{-x}}{2}$

我们需要求 $A(x)^a B(x)^b C(x)^c$ 中 x^n 的系数，直接展开即可。

而 $A(x)^a B(x)^b C(x)^c$ 的结果中，一定由若干项 e^{ux} 连乘得到，其中 u 为整数。

困难题 (Schwierig)

一个数组长度为 n，我们要在每个位置中填入 1 到 $a+b+c+d$ 的一个数字。

但是其中 $a+1$ 到 $a+b$ 的 b 个数字在整个数组中，必须出现 3 的倍数次。

但是其中 $a+b+1$ 到 $a+b+c$ 的 c 个数字在整个数组中，必须出现 3 的倍数 +1 次。

但是其中 $a+b+c+1$ 到 $a+b+c+d$ 的 d 个数字在整个数组中，必须出现 3 的倍数 +2 次。

问有多少种合法的方案？结果对 $p=10^9+9$ 取模。

$1 \leq n \leq 10^9, 1 \leq a+b+c+d \leq 100$

讨论

根据之前的提示和上一题的做法

我们引入三次单位根 ω，他满足 $\omega^3 = 1 \bmod p$。

其中 $\omega = 115381398$ 是一个解。

解法

设 $A(x)=\frac{1}{0!}+\frac{x}{1!}+\frac{x^2}{2!}+\cdots=e^x$

设 $B(x)=\frac{1}{0!}+\frac{x^3}{3!}+\cdots=\frac{e^x+e^{\omega x}+e^{\omega^2 x}}{3}$

设 $C(x)=\frac{x}{1!}+\frac{x^4}{4!}+\cdots=\frac{e^x+\omega^2 e^{\omega x}+\omega e^{\omega^2 x}}{3}$

设 $D(x)=\frac{x^2}{2!}+\frac{x^5}{5!}+\cdots=\frac{e^x+\omega e^{\omega x}+\omega^2 e^{\omega^2 x}}{3}$

我们需要求 $A(x)^a B(x)^b C(x)^c D(x)^d$ 中 x^n 的系数，找到 p 的一个三次单位根 ω，直接带入并且展开即可。

而 $A(x)^a B(x)^b C(x)^c D(x)^d$ 的结果中，一定由若干项 $e^{(u+v\omega)x}$ 连乘得到，其中 u,v 为整数。

第四部分

- 生成函数什么的已经无所谓了。因为已经不再有题，值得去写了。多项式已经不需要了。因为已经不再有数，值得去算了。
- 这一部分主要是关于本质不同的计数。
- 希望大家了解 Burnside 引理和 Polya 定理
- 

伯恩赛德引理

$$|X/G| = \frac{1}{|G|}\sum_{g\in G}|X^g|$$

其中 G 是一些由 X 的排列构成的群（也是一个集合）

$X^g = \{x \in X | g.x = x\}$ 也就排列 g 的不动点。

波利亚计数定理

$$|Y^X/G| = \frac{1}{|G|}\sum_{g\in G} m^{c(g)}$$

其中 X 表示位置集合，Y 表示颜色集合，G 是一些由 X 的排列构成的群。

其中 Y^X 表示所有 $X \to Y$ 的函数的集合。

简单来说 $|Y^X/G|$ 表示在置换 G 下有多少种本质不同的函数。

其中 $c(g)$ 表示排列 g 的循环指标 (the number of cycles)。

这些就是我们计算的方法，现在你已经理解了计算背后的思想。我们一起看一个易于理解的例子，来把我们所学到的用于实践！

路径 (PATH)

在一个 $n \times n$ 的网格中，从左下角到右上角，每次向上或者向右走一格，一共需要向右走 n 次，向上走 n 次。输入 $n(1 \leq n \leq 10^6)$，问本质不同的方案有多少种。结果对 $p = 10^9 + 9$ 取模。

如果两个方案可以通过对称和旋转变成相同的，我们认为他们本质相同。

讨论

- 作为一个为了让大家熟悉 Burnside 而设置的题目，我们就直接看结果吧。

解答

将向左看成 0，向上看成 1，一条路径可以用一个包含 n 个 0，n 个 1 的零一序列来进行描述。

我们有四种变换，分别是

- 不变。
- reverse。
- 零一互换。
- 零一互换并且 reverse。

我们考虑在这四种变换下，有哪些方案不会变，即不动点个数。

- 不变。所有方案都不会发生变化，答案 $\binom{2n}{n}$。
- reverse。这意味着第 i 个和倒数第 i 个绑定在一起，如果 n 是偶数，答案为 $\binom{n}{n/2}$，否则为 0。
- 零一互换。方案不可能不发生变化，答案为 0。
- 零一互换并且 reverse。这意味着第 i 个和倒数第 i 个必须一个是 1 一个是 0，也就是说前 i 个填好之后，后 i 个会自动形成一个合法的，答案为 2^n。

最终的答案就是这 4 个答案的平均值。

环染色 (Ring)

一个由 $n(2 \leq n \leq 10^9)$ 个珠子组成的环，我们要将所有珠子染成 $c(1 \leq c \leq 10^9)$ 种颜色之一。问存在多少种本质不同的方案，结果对 $10^9 + 7$ 取模。

当两种方案通过旋转后变得相同时，我们认为他们本质相同。

但是不能翻转。

讨论

这个就不需要讨论了，大家一定会。
答案是 $\frac{\sum_{d|n}\varphi(n/d)c^d}{n}$
我们可以对这个题目做一些修改。

环染色 (Ring)

一个由 $n(2 \le n \le 10^9)$ 个珠子组成的环，我们要将所有珠子染成 黑白两种颜色。但是黑色不能相邻，问有多少种本质不同的方案，结果对 10^9+7 取模。
当两种方案通过旋转后变得相同时，我们认为他们本质相同。
但是不能翻转。

讨论

这个就不需要讨论了，大家一定会。
答案是 $\frac{\sum_{d|n}\varphi(n/d)f_d}{n}$
其中 f_d 是不考虑本质相同，染一个长度为 d 的环的合法方案数。

$$f_0 = 1, f_1 = 3, f_i = f_{i-1} + f_{i-2}$$

我们还可以对这个题目做一些修改。

面染色 (Face)

一个由 $n \times m$ 的矩形，我们要用他非常正常地密铺整个平面，我们要将所有格子染成 c 种颜色。问有多少种本质不同的方案，结果对 10^9+7 取模。

当两种方案看起来一样时，即矩形通过循环平移时一样，我们认为他们本质相同。

$1 \le n \le 10^9, 1 \le m \le 10^9, 1 \le c \le 10^9$

讨论

这个就不需要讨论了，大家一定会。

答案是 $\frac{\sum_{a|n}\sum_{b|m}\varphi(n/a)\varphi(m/b)c^{nm/\operatorname{lcm}(a,b)}}{nm}$

我们还可以对这个题目做一些修改，将他变成一个华丽的题目。

面染色 (Face)

一个由 $n \times m$ 的矩形，我们要用他非常正常地密铺整个平面，我们要将所有格子染成 c 种颜色。输入 n, m, c 和一个长度为 c 的排列 p。问有多少种本质不同的方案，结果对 $10^9 + 7$ 取模。当两种方案看起来一样时，即矩形通过循环平移时一样，或者将每种颜色 i 同时替换为 p_i 时相同，我们认为他们本质相同。

$1 \leq n, m \leq 10^9, 1 \leq c \leq 16$

讨论

这个就需要讨论了。

- 输入的排列 p 究竟影响了什么？

解答

- 输入的排列 p 究竟影响了什么？
  - p 不是有用的，有用的是 p 中每个轮换的长度。
- 新题目究竟改了什么？
  - 我们不能简简单单的颜色数 c 的循环指标次方了。
  - 定义新函数 $C(x)$

```
def C(x):
    result = 0
    for i in period_lengths:
        if x % i == 0:
            result += i
    return result;
```

所以最终答案是

$$\frac{\sum_{a|n}\sum_{b|m}\varphi(n/a)\varphi(m/b)(C(\operatorname{lcm}(a,b)))^{nm/\operatorname{lcm}(a,b)}}{nm}$$

我们还可以做进一步的修改，将他变成一个更华丽丽的题目。

羊毛 (Wolle)

一个由 $n \times m$ 的矩形，我们要用他非常正常地密铺整个平面，我们要将所有格子染成 c 种颜色。但是 B 君是一个色盲，B 君只能判断两种颜色是否相同，而无法判断出每种颜色具体是什么。输入 n, m, c。问有多少种本质不同的方案，结果对 $10^9 + 7$ 取模。当两种方案看起来一样时，即矩形通过循环平移时一样，或者将颜色重新标号，我们认为他们本质相同。

$1 \leq n, m \leq 10^9, 1 \leq c \leq 16$

讨论

- 一些简单的情况？
- 两种颜色？
- 三种颜色？

解答

我们得到一个枚举所有颜色的排列，
利用上一题的结论，最后对答案求平均的做法。
但是 c 的范围是 16。

解答

我们只需要枚举排列中所有轮换的长度，也就是 c 的一个拆分。
然后计算这个拆分对应的排列有多少个。
而 16 的拆分数只有 231 完全可以接受。

Isomorphism(sgu282)

对一个 n 个点的无向完全图所有的边进行染 m 种颜色，问有多少种本质不同的染色方法。
如果两个方案通过对点的重新标号可以变为相同的，那么我们认为他们本质相同。
$1 \leq n \leq 53, 1 \leq m \leq 1000$

解法：

一共有 $n!$ 个置换（每一个置换是一个排列），不方便直接做。

我们观察置换的特征，发现置换中循环节的个数只和这个排列的循环节的长度有关系。

于是一个新的想法出现了，枚举 n 的所有拆分。然后对于每个拆分（也是一种置换）计算循环节的个数。

现在有两种边，第一种是在同一个环里，对于长度为 l 的环，有 l 个循环节。
第二种是不在同一个环里，对于长度为 a 和 b 的两个环，有 $\gcd(a, b)$。
第二个要考虑长度相同的环。

最后一题 (Letzte)

B 君作为一个色盲，对一个 n 个点的无向完全图所有的边进行染 m 种颜色，问有多少种本质不同的染色方法。

如果两个方案通过对点的重新标号，或者对颜色的重新标号，可以变为相同的，那么我们认为他们本质相同。

$1 \leq n \leq 16, 1 \leq m \leq 16$

讨论

解答

- 有了之前的经验，这个题目显得也就不那么困难了。
- 做法便是 sgu282 和 Wolle 嵌套在一起（所以数据范围只有 16）
- 需要特别注意的是，之前的 C 函数中用到的变量 x 在同一次计算中也不同。
- 我们需要统计每一种轨道长度和这种长度的数量，并作出相应的计算。
- 最后求平均即可。

题目出处

- Matrjoschka 我自己出的（显然别人出过了）
- Freund 我自己出的
- Subsets 我自己出的
- Fibonacci 参考 Codechef FN
- Radixphi 我自己出的，参考下一题（显然别人出过了）
- PE558 来自 Project Euler 558

题目出处

- Powersum 我自己出的（显然别人出过了）
- DMCS 我自己出的，参考上一题（也许别人出过了）
  Codechef DMCS
- PE258 来自 Project Euler 258
- Kirchhoff 我自己出的（也许别人出过了）
- Circular 这是 14 年 PKU ACM 的 D 题 Colorful World。在 Openjudge 上可以提交。
- BIKE 参考上一题，Codechef BIKE
- CUBE 我自己出的（也许别人出过了）THOI 2014
- Stein 我自己出的，参考上一题。清华集训 2016

题目出处

- Quadratwurzel 参考 GTM238 习题 2.2
- Einfach 某人出在了玲珑杯上，我自己加强的。
- PE389 来自 Project Euler 389，做法来自于这道题的讨论。

题目出处

- PATH 参考 GTM238 习题 6.5。
- Ring 经典题目大合集。
- Color 某一年的 ACM 题，但是我找不到出处了。
- Wolle 我自己出的。
- sgu282 来自 sgu282，经典论文题。
- Letzte 上面几个题的大杂烩。

感谢

- 我需要感谢许多人与我讨论这些题目并指点我做法。
- 其中特别感谢我的同届同学
  罗雨屏，贾志鹏，胡渊鸣，陈立杰，王若松。
- 还有我的高中同学 李煜东，刘炎明。
- 最后一行，献给 G 君。

参考资料

参考资料

- 其实我这六七年（是的，高一开始）也没看什么算法书。

参考资料

- 其实我这六七年（是的，高一开始）也没看什么算法书。
- 大概提三本书：

参考资料

- 其实我这六七年（是的，高一开始）也没看什么算法书。
- 大概提三本书：
  - 《Introduction to Algorithms》也就是《算法导论》，讲解了算法的概念和许多常用的算法，比如网络流，FFT。

参考资料

- 其实我这六七年（是的，高一开始）也没看什么算法书。
- 大概提三本书：
  - 《Introduction to Algorithms》也就是《算法导论》，讲解了算法的概念和许多常用的算法，比如网络流，FFT。
  - 《Concrete Mathematics》也就是《具体数学》，讲解了常见的组合数学问题。

参考资料

- 其实我这六七年（是的，高一开始）也没看什么算法书。
- 大概提三本书：
  - 《Introduction to Algorithms》也就是《算法导论》，讲解了算法的概念和许多常用的算法，比如网络流，FFT。
  - 《Concrete Mathematics》也就是《具体数学》，讲解了常见的组合数学问题。
  - 《A Course in Enumeration》也就是上文中的 GTM238，讲解了一些高深的组合计数问题。

参考资料

- 其实我这六七年（是的，高一开始）也没看什么算法书。
- 大概提三本书：
  - 《Introduction to Algorithms》也就是《算法导论》，讲解了算法的概念和许多常用的算法，比如网络流，FFT。
  - 《Concrete Mathematics》也就是《具体数学》，讲解了常见的组合数学问题。
  - 《A Course in Enumeration》也就是上文中的 GTM238，讲解了一些高深的组合计数问题。
- 如果要说还有什么书，那就是《算法艺术与信息学竞赛》这个对我刷题选择有很大影响。

参考资料

- 其实我这六七年（是的，高一开始）也没看什么算法书。
- 大概提三本书：
  - 《Introduction to Algorithms》也就是《算法导论》，讲解了算法的概念和许多常用的算法，比如网络流，FFT。
  - 《Concrete Mathematics》也就是《具体数学》，讲解了常见的组合数学问题。
  - 《A Course in Enumeration》也就是上文中的 GTM238，讲解了一些高深的组合计数问题。
- 如果要说还有什么书，那就是《算法艺术与信息学竞赛》这个对我刷题选择有很大影响。
- 《算法竞赛入门经典》也是很大的。但这是次要的，主要还是那三本外文书的作用。

很惭愧，就做了一点微小的工作，谢谢大家。

很惭愧，就做了一点微小的工作，谢谢大家。

如果对本 PDF 有疑问，欢迎交流

很惭愧，就做了一点微小的工作，谢谢大家。

如果对本 PDF 有疑问，欢迎交流

Email：wwwwodddd@qq.com

很惭愧，就做了一点微小的工作，谢谢大家。

如果对本 PDF 有疑问，欢迎交流

Email：wwwwodddd@qq.com

QQ：猜不到就不要加了。