數學

sam571128

2022-07-06

競賽中會遇到的數學?

- 在打程式競賽時，我們時常會需要使用到一些數學的技巧來幫助我們解決題目
- 而這些數學的範圍，有許多與高中的數學課所教的不太相同
- 在競賽中會遇到的數學主要有以下幾個 (大部分都是屬於離散數學的範圍)
  1. 數論
  2. 排組
  3. 矩陣
  4. 生成函數、線性代數、還有更多 ...

競賽中會遇到的數學?

- 在打程式競賽時，我們時常會需要使用到一些數學的技巧來幫助我們解決題目
- 而這些數學的範圍，有許多與高中的數學課所教的不太相同
- 在競賽中會遇到的數學主要有以下幾個 (大部分都是屬於離散數學的範圍)
  1. 數論
  2. 排組
  3. 矩陣
  4. 生成函數、線性代數、還有更多 ...
- 本堂課會介紹剛剛講的前三個，如果時間還夠的話，我們可能會補充第四個

大綱

- 快速冪
- 數論
  1. 質因數分解
  2. 最大公因數
  3. 同餘
  4. 費馬小定理
  5. 歐拉函數
  6. 拓展歐基里得算法
  7. 中國剩餘定理
- 排列組合
- 矩陣

快速冪

當我們要計算一個數字 a 的 b 次方時，最直覺的想法是，我們可以直接計算

$$a^b = \underbrace{a \times a \times \cdots \times a}_{\text{b times}}$$

也就是在 $O(b)$ 的時間完成次方的動作。

快速冪

不過事實上，我們有更快的方式可以處理這個問題，將 b 表示為二進位的形式 $b_k b_{k-1} \dots b_1 b_0$

$$a^b = \underbrace{b_0 a^0 \cdot b_1 a^1 \cdot \dots}_{\log_2 b \text{ times}}$$

我們就可以在 $O(\log_2 b)$ 的時間完成次方的計算了

快速冪

參考程式碼

```
int fastpow(int n, int p){
    int res = 1;
    while(p){
        if(p&1) res = res * n % MOD;
        n = n * n % MOD;
        p /= 2;
    }
    return res;
}
```

練習題

CSES - Exponentiation

給你兩個數字 a, b，請輸出 $a^b \bmod 10^9 + 7$ 的答案。

數論

什麼是數論?

- 數論是數學的其中一個分支，主要研究整數的性質
- 整數、質數、模運算等等的都是數論的範圍

質因數分解

- 首先，我們先從大家最熟悉的東西開始談，也就是質因數分解
- 他的方式有很多，而我們先從最簡單的方式開始談談

質因數分解

- 首先，既然我們想要質因數分解一個數字 n，則我們可以從 2 開始一路跑到 n
- 如果找到 n 的因數時，就一直將他除掉，最後得到的那些數字就會是 n 的因數了。
- 時間複雜度:$O(n)$

```
vector<int> num;
for(int i = 2;i <= x;i++){
    while(x%i==0){
        num.push_back(i);
        x /= i;
    }
}
```

質因數分解

- 不過，我們真的需要跑到 n 那麼多個數字嗎?

質因數分解

- 不過，我們真的需要跑到 n 那麼多個數字嗎?
- 其實不用，我們會發現對於 n，如果他不是質數，他的質因數的大小不會超過 $\sqrt{n}$

質因數分解

- 不過，我們真的需要跑到 n 那麼多個數字嗎?
- 其實不用，我們會發現對於 n，如果他不是質數，他的質因數的大小不會超過 $\sqrt{n}$
- 因此，其實我們只要從 2 枚舉到 $\sqrt{n}$ 就可以將這個數字質因數完了
- 時間複雜度: $O(\sqrt{n})$

```
vector<int> num;
for(int i = 2;i*i <= x;i++){
    while(x%i==0){
        num.push_back(i);
        x /= i;
    }
}
if(x != 1) num.push_back(x);
```

質因數分解

- 這個方式聽起來已經很快了吧!
- 不過如果 n 很小，我們有更快的方式可以處理這件事!

質因數分解

- 這個方式聽起來已經很快了吧!
- 不過如果 n 很小，我們有更快的方式可以處理這件事!
- 而這個方式就是預處理! 首先，我們要先知道怎麼找 $1 \sim n$ 之間的所有質數。

判斷質數

- 我們要如何判斷 $1 \sim n$ 哪些數字會是質數呢?

判斷質數

- 我們要如何判斷 $1 \sim n$ 哪些數字會是質數呢?
- 假設我們已經知道 x 是質數了，那我們可以知道 $2x, 3x, \ldots$ 就都不會是質數了

判斷質數

- 我們要如何判斷 $1 \sim n$ 哪些數字會是質數呢?
- 假設我們已經知道 x 是質數了，那我們可以知道 $2x, 3x, \ldots$ 就都不會是質數了
- 我們可以預先將所有數字都先設質數，然後用上面的方法跑過每個數字和他們的倍數

判斷質數

- 我們要如何判斷 $1 \sim n$ 哪些數字會是質數呢?
- 假設我們已經知道 x 是質數了，那我們可以知道 $2x, 3x, \ldots$ 就都不會是質數了
- 我們可以預先將所有數字都先設質數，然後用上面的方法跑過每個數字和他們的倍數
- 這個方法叫做「埃氏篩法 (Sieve of Eratosthenes)」

判斷質數

```cpp
const int MAXN = 1e6+5;
bool prime[MAXN];
vector<int> primes;

void init(){
    fill(prime+2,primes+N,1);
    for(int i = 2;i < N;i++){
        if(prime[i]==1){
            primes.push_back(i);
        }
        for(int p : primes){
            if(p*i ≥ MAXN) break;
            prime[p*i]=p;
            if(p==lpf[i]) break;
        }
    }
}
```

判斷質數

- **不過這個方式的時間複雜度是多少呢?**
- $O(\frac{n}{2}+\frac{n}{3}+\frac{n}{5}+\cdots)\approx O(n\log\log n)$

判斷質數

- 不過這個方式的時間複雜度是多少呢?
- $O(\frac{n}{2}+\frac{n}{3}+\frac{n}{5}+\cdots)\approx O(n\log\log n)$
- 有興趣的人可以去查查看為什麼他會是這個複雜度，可以使用泰勒級數等方式證明

判斷質數

題外話: 剛剛提到了埃氏篩法的神奇複雜度估算，我們在枚舉因數時會使用到以下程式碼

```
for (int i = 2; i <= n; i++) {
    for (int j = i; j <= n; j += i) {
        //Do Something
    }
}
```

判斷質數

題外話: 剛剛提到了埃氏篩法的神奇複雜度估算，我們在枚舉因數時會使用到以下程式碼

```
for (int i = 2; i <= n; i++) {
    for (int j = i; j <= n; j += i) {
        //Do Something
    }
}
```

複雜度會是 $O(\frac{n}{1}+\frac{n}{2}+\dots)$，用積分 $\int_1^n \frac{1}{x}dx = \ln(n)$ 的方式來估計會是 $O(n\log n)$

有人會稱這個東西為 Harmonic Series (調和級數) 的複雜度，總之就是一個小技巧

判斷質數

- 雖然剛剛那個方式已經非常快速了，大部分的問題也都可以直接解決
- 不過事實上還有更快的作法可以幫助我們做到這件事

判斷質數

- 雖然剛剛那個方式已經非常快速了，大部分的問題也都可以直接解決
- 不過事實上還有更快的作法可以幫助我們做到這件事
- 我們來看看下面這份 code

判斷質數

```cpp
const int MAXN = 1e6+5;
bool prime[MAXN];
vector<int> primes;

void init(){
    fill(prime+2,prime+MAXN,true);

    for(int i = 2;i < MAXN;i++){
        if(prime[i]){
            primes.push_back(i);
        }
        for(auto p : primes){
            if(p*i >= MAXN) break;
            prime[p*i] = true;
            if(i % p == 0) break;
        }
    }
}
```

判斷質數

- 這份 code 的時間複雜度是 $O(n)$ ，也就是線性的

判斷質數

- 這份 code 的時間複雜度是 $O(n)$ ，也就是線性的
- 我們通常稱這個做法為「線性篩」

判斷質數

- 這份 code 的時間複雜度是 $O(n)$ ，也就是線性的
- 我們通常稱這個做法為「線性篩」
- 有了上面這兩個篩法，我們就可以來做質因數分解了!

回到質因數分解

- 首先，要進行更快的質因數分解，我們要對每個人找他們的 LPF

Least Prime Factor (最小質因數)

對於一個數字 x，我們定義他的 `lpf[x]` 為他最小的質因數

範例: `lpf[2]` $= 2$、`lpf[15]` $= 3$

回到質因數分解

- 首先，要進行更快的質因數分解，我們要對每個人找他們的 LPF
- 我們可以使用剛剛的兩個篩法進行預處理
- 就可以在 $O(\log n)$ 的時間對 n 進行質因數分解了!

Least Prime Factor (最小質因數)

對於一個數字 x，我們定義他的 `lpf[x]` 為他最小的質因數

範例: `lpf[2]` $= 2$、`lpf[15]` $= 3$

回到質因數分解

附上有 lpf 之後的質因數分解 code

```
vector<int> pf; //紀錄 x 的質因數

while(x != 1){
    while(x % lpf[x] == 0){
        pf.push_back(lpf[x]);
        x /= lpf[x];
    }
}
```

例題

Ten Point Round #22 pI - 狐狸買氣球 (Balloon)

現在你有一個長度為 n 的序列 $a_1, a_2, \ldots, a_n$，定義一個數字 x 的美麗程度為將 x 寫成 $x = p_1^{b_1} p_2^{b_2} \ldots p_k^{b_k}$ 後，$\sum_{i=1}^{k} (p_i \times b_i)$ 的數值。現在有 q 筆詢問，每筆詢問請輸出區間 $[l, r]$ 所有數字相乘後的美麗程度為何?

測資範圍:

- $1 \leq n, q \leq 10^6$
- $1 \leq a_i \leq 2 \times 10^6$

- 這題一樣留點時間給大家想想

例題

Ten Point Round #22 pI - 狐狸買氣球 (Balloon)

現在你有一個長度為 n 的序列 $a_1, a_2, \ldots, a_n$，定義一個數字 x 的美麗程度為將 x 寫成 $x = p_1^{b_1} p_2^{b_2} \ldots p_k^{b_k}$ 後，$\sum_{i=1}^{k} (p_i \times b_i)$ 的數值。現在有 q 筆詢問，每筆詢問請輸出區間 $[l, r]$ 所有數字相乘後的美麗程度為何?

測資範圍:

- $1 \leq n, q \leq 10^6$
- $1 \leq a_i \leq 2 \times 10^6$

- 應該會發現，不同的數字其實貢獻可以分別討論
- 然後我們可以依序對每個 a_i 進行質因數分解
- 這樣的複雜度會是 $O(n \log C)$ (C 是 a_i 的值域)
- 接著使用前綴和就可以 $O(1)$ 詢問了

練習題

CSES - Counting Divisors

給你一個數字 x，請輸出 x 有幾個因數。

提示: 若 $n = p_1^{\alpha_1} p_2^{\alpha_2} \dots p_k^{\alpha_k}$，則 n 有 $\prod_{i=1}^{k}(\alpha_i + 1)$ 個因數

Zerojudge a010. 因數分解

給你一個數字 x，請對 x 質因數分解。

CSES - Common Divisors

你有 n 個數字，你希望能找到在這個陣列中尋找兩個數字時，他們的最大公因數最大可能是多少?

提示: 調和級數 $O(n \log n)$

最大公因數 (Greatest Common Divisor)

- 接著，我們來談談最大公因數

最大公因數 (Greatest Common Divisor)

- 接著，我們來談談最大公因數
- 相信很多人都知道 C++ 其實有內建 `__gcd(x,y)` 這個函數可以找 x, y 的最大公因數
- 而 C++17 之後，甚至可以直接使用 `gcd(x,y)` 與 `lcm(x,y)`

最大公因數 (Greatest Common Divisor)

- 接著，我們來談談最大公因數
- 相信很多人都知道 C++ 其實有內建 `__gcd(x,y)` 這個函數可以找 x, y 的最大公因數
- 而 C++17 之後，甚至可以直接使用 `gcd(x,y)` 與 `lcm(x,y)`
- 不過在此，我們還是來談談計算最大公因數的方式

最大公因數 (Greatest Common Divisor)

- 要計算最大公因數，我們最常使用的方式就是所謂的「輾轉相除法 (Euclidean Algorithm)」
- 或者說他是「歐幾里得演算法」，這個方法相信大家在國小或國中時就已經學過了

最大公因數 (Greatest Common Divisor)

輾轉相除法 (歐幾里得演算法)

當我們有兩個數字時，我們可以利用 $\gcd(a,b)=\gcd(b,a \bmod b)$ 的想法，持續向下遞迴，直到我們得到 $\gcd(x,0)$ 時，x 即為 $\gcd(a,b)$

- **此算法的時間複雜度最差為 $O(\log \max(a,b))$，在 a,b 為費氏數時會有 worst case**
- **不過在一般的情況下，他基本上會跑得非常快速。**

最大公因數 (Greatest Common Divisor)

輾轉相除法 (歐幾里得演算法)

當我們有兩個數字時，我們可以利用 $\gcd(a,b) = \gcd(b, a \bmod b)$ 的想法，持續向下遞迴，直到我們得到 $\gcd(x,0)$ 時，x 即為 $\gcd(a,b)$

證明

設 $m = \gcd(a,b)$，並將 b 用除法原理寫成 $aq + (a \mod b)$，由於 $m \mid \gcd(a,b) \Rightarrow m \mid a, m \mid b$，因此，$m \mid b - aq = (a \mod b)$。

最大公因數 (Greatest Common Divisor)

輾轉相除法 (歐幾里得演算法)

當我們有兩個數字時，我們可以利用 $\gcd(a,b) = \gcd(b, a \bmod b)$ 的想法，持續向下遞迴，直到我們得到 $\gcd(x,0)$ 時，x 即為 $\gcd(a,b)$

```
int gcd(int a, int b){
    if(b==0) return a;
    return gcd(b,a%b);
}
```

貝祖定理 (Bézout's identity))

貝祖定理 (Bézout's identity)

對於任何整數 a, b, c**，若** $ax + by = c$ **存在整數解** $\iff$ $gcd(a, b) \mid c$

貝祖定理 (Bézout's identity)

證明

- ($\Rightarrow$) 假設 $ax + by = c$ 存在整數解 $x = x_0, y = y_0$，則 $ax_0 + by_0 = c$，由於 $gcd(a, b) \mid a,\ gcd(a, b) \mid b$，原式可寫為 $gcd(a, b)(\frac{a}{gcd(a, b)}x_0 + \frac{b}{gcd(a, b)}y_0) = c$，故 $gcd(a, b) \mid c$
- ($\Leftarrow$) 假設 $a = 0$，則 $gcd(a, b) = b$，式子變為 $by = c$，若 x, y 存在整數解，則 $b \mid c$。假設 a, b 皆不為 0，設集合 $S = \{ax + by | x, y \in \mathbb{Z}^2\}$，而 $S \cap \mathbb{N}$ 並非空集合 (必存在 $|a|, |b|$)。根據良序定理，S 內必存在一個最小的正整數 $d = ax_0 + by_0$。考慮 S 中另一個數字 $p = ax_1 + by_1$。根據除法原理，$p = qd + r$，其中 $q \in \mathbb{Z},\ 0 \leq r < d$。因此 $r = p - qd = (ax_1 + by_1) - q(ax_0 + by_0) = a(x_1 - qx_0) + b(y_1 - qy_0)$，得 $r \in S$。又因為前面的假設，r 必為 0。任何 S 當中的數字皆為 d 的倍數。而 $a, b \in S$，因此 $d \mid a,\ d \mid b \Rightarrow d \mid gcd(a, b)$。而 $ax + by = d$ 有整數解，因此 $ax + by = gcd(a, b)$ 必有整數解。又因為 $gcd(a, b) \mid c$，因此 $ax + by = c$ 必有整數解

拓展歐幾里得演算法 (Extended GCD Algorithm)

有了剛剛的貝祖定理之後，如果 $ax + by = c$ 有整數解，我們要怎麼找到一組 x, y 呢?

拓展歐幾里得演算法 (Extended GCD Algorithm)

如果 $ax + by = c$ 有整數解，則 $bx' + (a \bmod b)y' = c$ 必有整數解 (gcd 不變)，我們將 $a \bmod b$ 轉換為 $a - b\left\lfloor \frac{a}{b} \right\rfloor$。原式可以轉換為

$$bx' + (a - b\left\lfloor \frac{a}{b} \right\rfloor)y' = c$$

$$bx' + ay' - b\left\lfloor \frac{a}{b} \right\rfloor y' = c$$

$$ay' + b(x' - \left\lfloor \frac{a}{b} \right\rfloor y') = c$$

我們可以使用遞迴的方式一路由下往上計算答案

拓展歐幾里得演算法 (Extended GCD Algorithm)

以下為範例程式碼

```
pair<int,int> extgcd(int a, int b){
    if(b == 0) return {1,0};
    else if(a == 0) return {0,1};
    auto [x,y] = extgcd(b,a%b);
    return {y,x-a/b*y};
}
```

練習題

NHDK Ten Point Round #21 - 調色盤

你有 n 個數字 $a_1, \ldots, a_n$，接著有 q 筆詢問，每次問你是否存在一組整數解 $\{b_1, \ldots, b_n\}$ 可以使得 $a_1 b_1 + a_2 b_2 + \cdots + a_n b_n + 1115 = T$。

Atcoder Beginner Contest 186 pE - Throne

有 n 個椅子排成環狀，這 n 個椅子其中有一個是王位，目前 Colten 坐在離王位順時鐘 s 格的椅子上，他每一次移動會順時鐘移動到 k 格後的椅子上，請問他至少要幾次移動才能坐到王位上，或者永遠坐不到。

同餘 (Modular Congruence)

在我們講什麼是同餘之前，我們要先介紹「模運算 (Modular arithmetic)」，通常在題目會遇到要輸出答案 $\bmod 10^9+7$ 或 998244353 的題目時，會用到這個運算

模運算

1. 基本上就是對整數除法取餘數，也就是大家熟悉程式中的的 %
2. $r = x \bmod m$，我們可以將 x 寫成 $x = km + r,\ 0 \le r < m$
3. $m \mid x \iff x \bmod m = 0$，表示存在一個 n 可以使得 $x = mn$

同餘 (Modular Congruence)

而當我們在同一個模數底下做運算時，我們會用到同餘這個概念

同餘 (Modular Congruence)

1. $a \equiv b \pmod{m}$ **表示** $a \bmod m = b \bmod m$ **或** $(a-b) \bmod m = 0$
2. **如果** $a \equiv b \pmod{m}$**，則**
   - $a + c \equiv b + c \pmod{m}$
   - $a - c \equiv b - c \pmod{m}$
   - $ac \equiv bc \pmod{m}$
   - **不滿足** $a \div c \equiv b \div c \pmod{m}$
3. **如果** $a \equiv b \pmod{m}$, $p \equiv q \pmod{m}$
   - $a + p \equiv b + q \pmod{m}$
   - $ap \equiv bq \pmod{m}$

而我們來證明一下剛剛的那幾個性質吧

加法的證明 $(a + c \equiv b + c \pmod{m})$

當 $a \equiv b \pmod{m}$ **時，假設** $a = k_1 m + r,\ b = k_2 m + r,\ k_1, k_2, r \in \mathbb{Z}$

則 $a + c = k_1 m + r + c,\ b + c = k_2 m + r + c \Rightarrow (a + c) - (b + c) = (k_1 - k_2)m \Rightarrow$
$(a + c) - (b + c) \bmod m = 0$ **，因此** $a + c \equiv b + c \pmod{m}$

同餘 (Modular Congruence)

減法的證明 ($a - c \equiv b - c \pmod{m}$)

當 $a \equiv b \pmod{m}$ **時，假設** $a = k_1 m + r,\ b = k_2 m + r,\ k_1, k_2, r \in \mathbb{Z}$

則 $a - c = k_1 m + r - c,\ b - c = k_2 m + r - c \Rightarrow (a + c) - (b + c) = (k_1 - k_2)m \Rightarrow$
$(a - c) - (b - c) \bmod m = 0$ **，因此** $a - c \equiv b - c \pmod{m}$

同餘 (Modular Congruence)

乘法的證明 ($ac \equiv bc \pmod{m}$)

當 $a \equiv b \pmod{m}$ 時，假設 $a = k_1 m + r,\ b = k_2 m + r,\ k_1, k_2, r \in \mathbb{Z}$

則 $ac = (k_1 m + r)c,\ bc = (k_2 m + r)c \Rightarrow ac - bc = (k_1 - k_2)cm \Rightarrow ac - bc \bmod m = 0$，因此 $a - c \equiv b - c \pmod{m}$

同餘 (Modular Congruence)

- 既然我們剛剛講到了加減乘，那要進行除法怎麼辦呢?
- 模運算並不滿足直接進行除法運算的性質，那我們該怎麼處理呢?
- 事實上，在有模數的運算系統下，有另外一個東西可以幫助我們做除法的操作

模反元素 (Modular Inverse)

對於一個元素 x，在 $\bmod m$ 的系統下，x^{-1} 為滿足 $xx^{-1} \equiv 1 \pmod{m}$ 的數字，我們稱 x^{-1} 為 x 的「模反元素」或「模逆元」。而並不是對於所有 x 皆存在一個模反元素，若模反元素存在則 x 必與 m 互質。

模反元素 (Modular Inverse)

對於一個元素 x，在 $\bmod m$ 的系統下，x^{-1} 為滿足 $xx^{-1} \equiv 1 \pmod{m}$ 的數字，我們稱 x^{-1} 為 x 的「模反元素」或「模逆元」。而並不是對於所有 x 皆存在一個模反元素，若模反元素存在則 x 必與 m 互質。

- 因此，當我們在模運算系統下要計算除法時，會使用 ab^{-1} 表示 a 除以 b 的操作

模反元素 (Modular Inverse)

對於一個元素 x，在 $\bmod m$ 的系統下，x^{-1} 為滿足 $xx^{-1} \equiv 1 \pmod{m}$ 的數字，我們稱 x^{-1} 為 x 的「模反元素」或「模逆元」。而並不是對於所有 x 皆存在一個模反元素，若模反元素存在則 x 必與 m 互質。

- 因此，當我們在模運算系統下要計算除法時，會使用 ab^{-1} 表示 a 除以 b 的操作
- 不過，我們要怎麼計算這個數字呢?

同餘 (Modular Congruence)

模反元素 (Modular Inverse)

對於一個元素 x，在 $\bmod m$ 的系統下，x^{-1} 為滿足 $xx^{-1} \equiv 1 \pmod{m}$ 的數字，我們稱 x^{-1} 為 x 的「模反元素」或「模逆元」。而並不是對於所有 x 皆存在一個模反元素，若模反元素存在則 x 必與 m 互質。

- 因此，當我們在模運算系統下要計算除法時，會使用 ab^{-1} 表示 a 除以 b 的操作
- 不過，我們要怎麼計算這個數字呢?
- 主流的方式有三種，我們來一一介紹

模反元素找法

- 費馬小定理 (只有在 m 是質數時可以使用)
- 建表法 (只有在 m 是質數時可以使用)
- 拓展歐幾里得定理 (又稱 extgcd，當 $gcd(a, m) = 1$ 時可以使用)

模反元素找法一 - 費馬小定理 (Fermat's Little Theorem)

費馬小定理 (Fermat's Little Theorem)

當 p 是質數時，對於任一個整數 a，滿足 $a^{p-1} \equiv 1 \pmod{p}$

模反元素找法一 - 費馬小定理 (Fermat's Little Theorem)

費馬小定理 (Fermat's Little Theorem)

當 p 是質數時，對於任一個整數 a，滿足 $a^{p-1} \equiv 1 \pmod{p}$

證明

1. 當 $p \mid a$ 時，必滿足 $a^p \equiv a \pmod{p}$。
2. 對於 $a, 2a, \ldots, (p-1)a$，這些數字除以 p 的餘數必一一對應到 $1, 2, \ldots, p-1$，否則任意兩個相同餘數的數字可以相減使得 $p \mid a$。因此
   $(p-1)!a^{p-1} \equiv a \times 2a \times \cdots \times (p-1)a \equiv (p-1)! \pmod{p}$。得 $a^{p-1} \equiv 1 \pmod{p}$

模反元素找法一 - 費馬小定理 (Fermat's Little Theorem)

費馬小定理 (Fermat's Little Theorem)

當 p 是質數時，對於任一個整數 a，滿足 $a^{p-1} \equiv 1 \pmod{p}$

要使用這個定理計算模反元素的話，我們可以將原式兩邊同乘以 a^{-1}，得到

$$a^{p-2} \equiv a^{-1} \pmod{p}$$

因此，我們可以使用快速冪來得到 a^{-1}。
時間複雜度: $O(\log p)$

模反元素找法二 - 建表法

假設我們想要計算 i 在 $\mod p$ (p 必須是質數) 下的模反元素，推導過程如下

$$
\begin{aligned}
p - i \times \left\lfloor \frac{p}{i} \right\rfloor &= p \bmod i \\
p - i \times \left\lfloor \frac{p}{i} \right\rfloor &\equiv p \bmod i \pmod p \\
p \times i - i \times \left\lfloor \frac{p}{i} \right\rfloor &\equiv p \bmod i \pmod p \\
i(p - \left\lfloor \frac{p}{i} \right\rfloor)(p \bmod i)^{-1} &\equiv 1 \pmod p \\
(p - \left\lfloor \frac{p}{i} \right\rfloor)(p \bmod i)^{-1} &\equiv i^{-1} \pmod p
\end{aligned}
$$

這個方式可以讓我們用遞迴的方式往下做，雖然個人不確定複雜度上界究竟是多少，但經過測試，幾乎都能在 $O(\log p)$ 的時間計算完，而我們可以使用一個迴圈依序計算每個數字的模反元素，複雜度 $O(p)$。

模反元素找法二 - 建表法

參考程式碼

```
void build_inv(int p){
    inv[1] = 1;
    for(int i = 2;i < p;i++){
        inv[i] = inv[p%i] * (p-p/i) % p;
    }
}
```

模反元素找法三 - 拓展歐幾里得定理 (Extended GCD)

當我們要找 a 在 $\mod m$ 下的模反元素，若滿足 $gcd(a, m) = 1$，則我們可以列出

$$aa^{-1} + bm = 1$$

使用拓展歐幾里得找 a^{-1} 即可

歐拉函數 (Euler Phi Function)

對於正整數 n，定義 $\phi(n)$ 為小於等於 n 且與 n 互質的正整數數量。

1. **對於一個質數 p，$\phi(p) = p - 1$**
2. **若 a, b 互質，則 $\phi(ab) = \phi(a)\phi(b)$ (積性函數)**
3. $a^{\phi(m)} \equiv 1 \pmod{m}$ **(歐拉定理)**
4. **假設 $n = p_1^{\alpha_1} p_2^{\alpha_2} \dots p_k^{\alpha_k}$，則 $\phi(n) = n \prod_{i=1}^{k} (1 - \frac{1}{p_i})$**

中國剩餘定理 (Chinese Remainder Theorem)

在孫子算經中，有一題「物不知數」，題目如下:

「有物不知其數，三三數之剩二，五五數之剩三，七七數之剩二。問物幾何？」

中國剩餘定理 (Chinese Remainder Theorem)

寫成數學式會長這樣

$$\begin{cases} x \equiv 2 \pmod 3 \\ x \equiv 3 \pmod 5 \\ x \equiv 2 \pmod 7 \end{cases}$$

中國剩餘定理 (Chinese Remainder Theorem)

寫成數學式會長這樣

$$\begin{cases} x \equiv 2 \pmod 3 \\ x \equiv 3 \pmod 5 \\ x \equiv 2 \pmod 7 \end{cases}$$

這個問題其實大家在國中課本可能也有看過類似的題目，像韓信點兵等等的問題。
而如何找到這個問題的解，就是我們要來解決的。

中國剩餘定理 (Chinese Remainder Theorem)

題外話，上面那題的解，在孫子算經中同樣也有描述

「三人同行七十希，五樹梅花廿一支，七子團圓正半月，除百零五便得知」

也就是說，將除以三的餘數乘以 70，除以五的餘數乘以 21，除以七的餘數乘以 15，接著除以 105 的餘數就會是答案了!

$$2 \times 70 + 3 \times 21 + 2 \times 15 \equiv 233 \equiv 23 \pmod{105}$$

因此，中國剩餘定理就是一個可以幫助我們解決一元一次同餘聯立方程式的方法

$$\begin{cases} x \equiv a_1 & (\text{mod } m_1) \\ x \equiv a_2 & (\text{mod } m_2) \\ x \equiv a_3 & (\text{mod } m_3) \\ & \vdots \\ x \equiv a_n & (\text{mod } m_n) \end{cases}$$

中國剩餘定理 (Chinese Remainder Theorem)

這個問題有兩種解法:

1. 使用通解的形式找到答案
2. 使用拓展歐幾里得算法計算答案 (比賽中常用的方式)

第一種作法的話，我們會快速帶過，在競程上比較常使用第二種方式

中國剩餘定理 (Chinese Remainder Theorem)

構造通解的方式

假設 $m_1, m_2, \ldots, m_n$ 互質

1. 設 $M = m_1 m_2 \ldots m_n = \prod_{i=1}^{n} m_i$，而 $M_i = M/m_i$
2. 設 $t_i \equiv M_i^{-1} \pmod{m_i}$，意即 t_i 是 M_i 的模反元素
3. 則通解會是 $x \equiv a_1 t_1 M_1 + a_2 t_2 M_2 + \cdots + a_n t_n M_n \equiv \sum_{i=1}^{n} a_i t_i M_i \pmod{M}$

使用拓展歐幾里得的方法

先假設我們只想要找兩個式子的解 (m_1, m_2 可以不互質)

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \end{cases}$$

則我們可以知道對於兩個整數 k_1, k_2

$$\begin{cases} x = k_1 m_1 + a_1 \\ x = k_2 m_2 + a_2 \end{cases}$$

合併兩個式子，我們會得到

$$k_1 m_1 + a_1 = k_2 m_2 + a_2$$

移項一下會得到

$$m_1 k_1 - m_2 k_2 = a_2 - a_1$$

使用拓展歐幾里得的方法

接著我們可以使用 extgcd，找到一組 k_1, k_2 的解 (k_1', k_2')

再將 k_1 代入 $x = a_1 + m_1 k_1$ 的式子當中，兩式就合併成

$$x \equiv a_1 + m_1 k_1 \pmod{lcm(m_1, m_2)}$$

例題

CSES - Exponentiation II

給你三個數字 a, b, c，請輸出 $a^{b^c} \bmod 10^9 + 7$ 的值

- 看到這題，我們可以直接使用上面的兩種方法去簡化這題
- 可以使用歐拉函數或費馬小定理去做化簡

例題

CSES - Exponentiation II

給你三個數字 a, b, c，請輸出 $a^{b^c} \bmod 10^9 + 7$ 的值

- **看到這題，我們可以直接使用上面的兩種方法去簡化這題**
- **可以使用歐拉函數或費馬小定理去做化簡**
- $a^{b^c} \equiv a^{b^c \bmod 10^9+6} \pmod{10^9+7}$

練習題

Codeforces 907F - Power Tower

你有 n 個數字 $a_1, a_2, \ldots, a_n$，和一個模數 m，接著有 q 筆詢問，每次詢問區間 $[l, r]$ 的

$$a_l^{a_{l+1}^{\cdots^{a_r}}} \bmod m$$

Kattis - Chinese Remainder Theorem

實作中國剩餘定理，模數不一定互質

組合計數

加法原理 (Addition Rule)

若一個集合 S 可以被分成 k 個互斥的集合 $S_1, S_2, \ldots, S_k$，則

$$|S| = |S_1| + |S_2| + \cdots + |S_k|$$

加法原理 (Addition Rule)

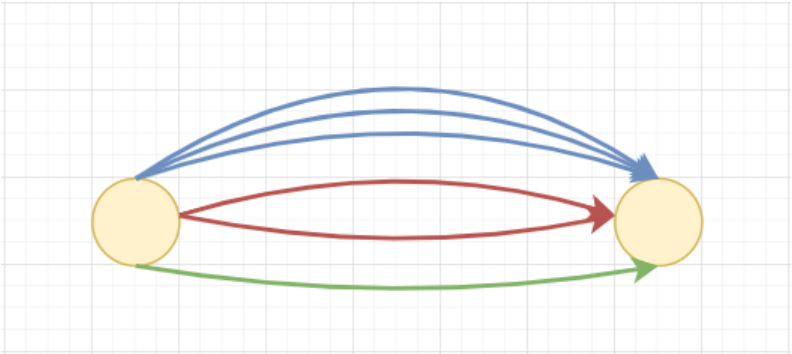
若一個集合 S 可以被分成 k 個互斥的集合 $S_1, S_2, \ldots, S_k$，則

$$|S| = |S_1| + |S_2| + \cdots + |S_k|$$

簡單來說，一個大問題，我們可以想成許多個互不相干的問題，將這些問題的答案相加就會是總共的方案數了。

加法原理 (Addition Rule)

假設從狀態 1 到狀態 2 有 3 種 A 方案，2 種 B 方案，1 種 C 方案，則一共有 6 種方案



乘法原理 (Multiplication Rule)

乘法原理 (Multiplication Rule)

若一個方案 S 可以被拆成 k 個步驟，而第 i 個步驟 S_i 共有 $|S_i|$ 種方案數，則

$$|S| = |S_1| \cdot |S_2| \cdot \cdots \cdot |S_k|$$

乘法原理 (Multiplication Rule)

乘法原理 (Multiplication Rule)

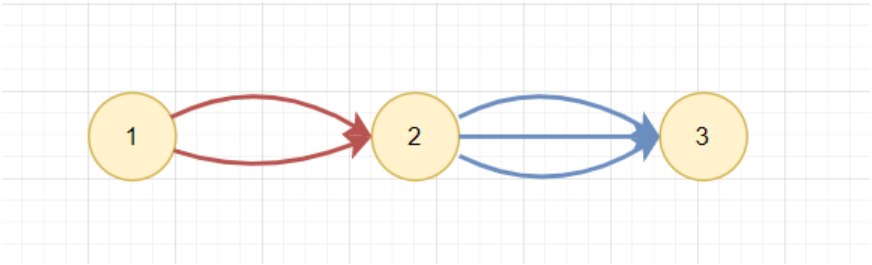
若一個方案 S 可以被拆成 k 個步驟，而第 i 個步驟 S_i 共有 $|S_i|$ 種方案數，則

$$|S| = |S_1| \cdot |S_2| \cdot \cdots \cdot |S_k|$$

簡單來說，一個大問題，如果他有很多個不同的步驟，則這些方案數相乘就會是總共的方案數了

乘法原理 (Multiplication Rule)

假設從狀態 1 走到 2 有 2 種方案，從 2 走到 3 有 3 種方案，則一共有 6 種方案從 1 到 3



排列 (Permutation)

對於相異的元素，計算他們經由不同順序可以組成的排列數量。

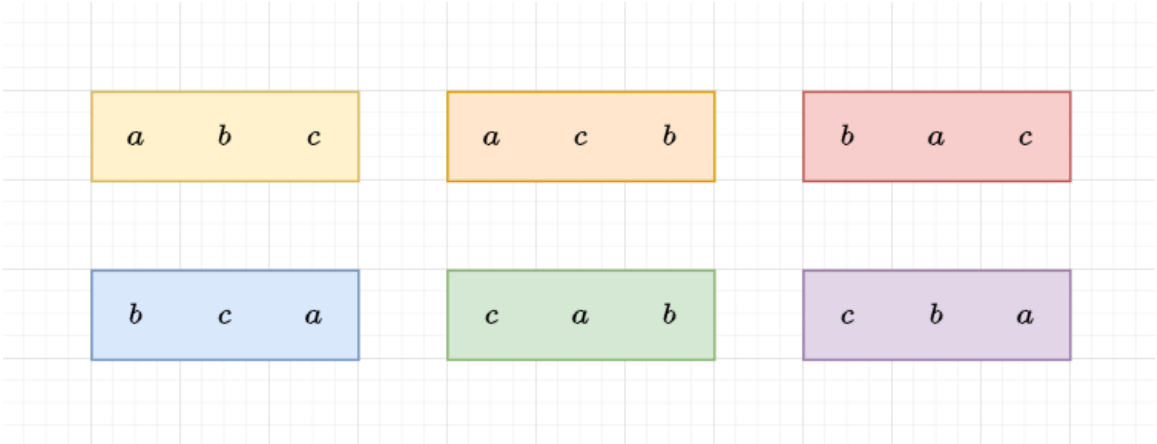
排列 (Permutation)

有 n 個相異元素時，總共有 $n!$ 種不同的排列。

- $n! = 1 \times 2 \times \cdots \times n$
- $0! = 1$

排列 (Permutation)

以下為 $\{a, b, c\}$ 的 $3! = 6$ 種排列方式



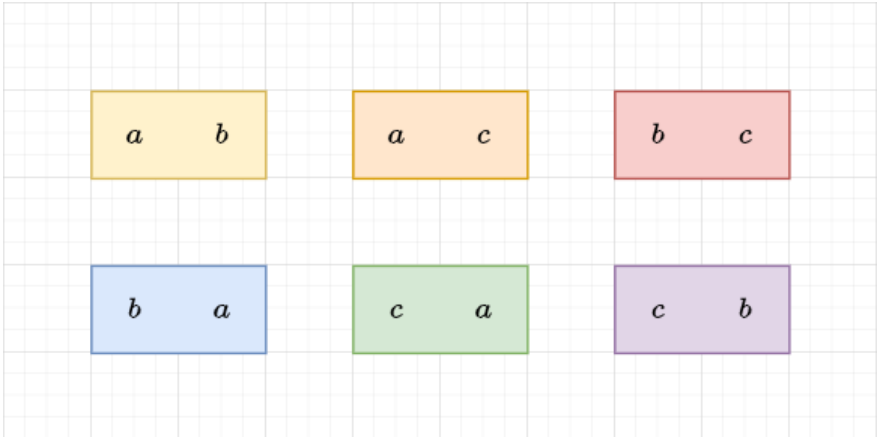
排列 (Permutation)

從 n 個相異元素選出 r 個做排列時，總共有 P_r^n 種不同的排列。

- $P_r^n = \dfrac{n!}{(n-r)!}$

排列 (Permutation)

以下從 $\{a, b, c\}$ 中取出 2 個元素做排列時的 $P_2^3 = 6$ 種排列方式



排列 (Permutation)

不盡相異物排列

有 n 個元素，而每個元素的出現次數共有 m_i 次，則排列的次數一共有

$$\frac{n!}{m_1!m_2!\dots m_k!}$$

例題

CSES - Creating Strings II

給你一個字串 s，問你有幾個字串可以由 s 經過重組後得到，由於數量可能很大，請輸出答案 $\bmod 10^9+7$。

例題

CSES - Creating Strings II

給你一個字串 s，問你有幾個字串可以由 s 經過重組後得到，由於數量可能很大，請輸出答案 $\bmod 10^9+7$。

基本上就是套剛剛的公式而已，不過由於剛剛有除法以及取模操作，記得要使用模反元素，不能直接用除的!

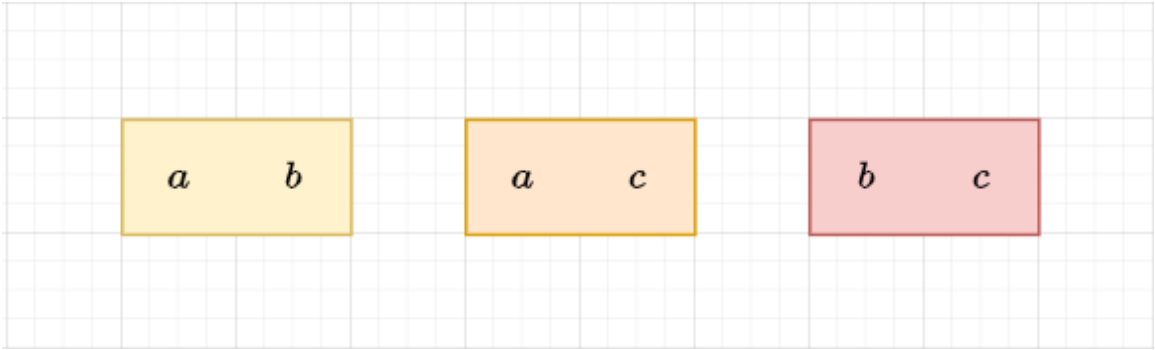
組合 (Combination)

從 n 個相異元素當中選出 r 個元素的一共有 C_r^n 種

- $C_r^n = \dfrac{n!}{(n-r)!r!}$
- $C_0^n = 1$
- $C_r^n = C_{n-r}^n$
- C_r^n 又可以被寫成 $\binom{n}{r}$
- $C_r^n = C_r^{n-1} + C_{r-1}^{n-1}$ (帕斯卡三角形)

組合 (Combination)

從 $\{a, b, c\}$ 當中任選 2 個元素的方法，一共有 $C_2^3 = 3$ 種



組合 (Combination)

重複組合 (Star and Bars)

從 n 個元素中取出 r 個，而這 r 個元素可以重複出現，一共有 H_r^n 種選法

- 等同於 $a_1 + a_2 + \cdots + a_n = r$ 的非負整數解數量
- 英文稱為 Star and Bars，可以想成是 r 個 1 和 $n-1$ 個 $+$ 在做重複排列
- H_r^n 又可以被寫成 $\left(\!\binom{n}{r}\!\right)$
- $H_r^n = C_{n-1}^{n+r-1}$

排容原理 (Inclusion-Exclusion Principle)

若我們有 n 個集合 $A_1, A_2, \ldots, A_n$，則

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{1\le i\le n} |A_i| - \sum_{1\le i<j\le n} |A_i \cap A_j| + \cdots + (-1)^{n+1}|A_1 \cap A_2 \cap \cdots \cap A_n|$$

- $n = 2$ **時，有** $|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|$
- $n = 3$ **時，有**
  $|A_1 \cup A_2 \cup A_3| = |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |A_2 \cap A_3| - |A_3 \cap A_1| + |A_1 \cap A_2 \cap A_3|$

二項式定理 (Binomial Theorem)

$$(1+x)^n = \binom{n}{0}x^0 + \binom{n}{1}x^1 + \cdots + \binom{n}{n}x^n$$

例題

AtCoder Beginner Contest 178C - Ubiquity

請找到有多少個長度為 N 的序列 $A_1, A_2, \ldots, A_N$ 滿足

- 對於所有 i，$0 \leq A_i \leq 9$
- 至少存在一個 i 使得 $A_i = 0$
- 至少存在一個 i 使得 $A_i = 9$

例題

AtCoder Beginner Contest 178C - Ubiquity

請找到有多少個長度為 N 的序列 $A_1, A_2, \ldots, A_N$ 滿足

- 對於所有 i，$0 \leq A_i \leq 9$
- 至少存在一個 i 使得 $A_i = 0$
- 至少存在一個 i 使得 $A_i = 9$

- 排容原理!
- 全部的數量 - (沒有 0) - (沒有 9) + (沒有 0 也沒有 9)
- 答案就是 $10^n - 9^n - 9^n + 8^n$

例題

CSES - Christmas Party

在一個聖誕節派對上，有 n 個人在玩交換禮物，每個人都會送出一個禮物，也會收到一個禮物。請問有幾種送法可以讓這 n 個人都收到不是自己送出去的禮物。

- 這個問題其實是經典的「錯排」問題

例題

CSES - Christmas Party

在一個聖誕節派對上，有 n 個人在玩交換禮物，每個人都會送出一個禮物，也會收到一個禮物。請問有幾種送法可以讓這 n 個人都收到不是自己送出去的禮物。

- 這個問題其實是經典的「錯排」問題
- 我們可以用排容原理來思考看看

例題

CSES - Christmas Party

在一個聖誕節派對上，有 n 個人在玩交換禮物，每個人都會送出一個禮物，也會收到一個禮物。請問有幾種送法可以讓這 n 個人都收到不是自己送出去的禮物。

- 這個問題其實是經典的「錯排」問題
- 我們可以用排容原理來思考看看
- 答案其實就是全部 - (至少 1 個人會收到自己的禮物) + (至少 2 個人會收到自己的禮物) - (至少 3 個人會收到自己的禮物) + $\ldots$
- $n! - \sum_{i=1}^{n} \binom{n}{i}(n-i)!$

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

- 首先，我們會發現到每個價值的機率其實就是 $(\frac{1}{m})^n$
- 因此，我們只要能夠計算所有情況的價值總和即可

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

- 首先，我們會發現到每個價值的機率其實就是 $(\frac{1}{m})^n$
- 因此，我們只要能夠計算所有情況的價值總和即可
- 那我們要怎麼計算這個答案呢?

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

- 首先，我們會發現到每個價值的機率其實就是 $(\frac{1}{m})^n$
- 因此，我們只要能夠計算所有情況的價值總和即可
- 那我們要怎麼計算這個答案呢?
- 我們將最大值為 x 的狀態分開思考

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

- 假設 n 次骰出來的最大值為 x，那總共有多少可能呢?
- 排容!

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

- 假設 n 次骰出來的最大值為 x，那總共有多少可能呢?
- 排容!
- 答案其實就是 (其中一次骰出 x) - (其中兩次骰出 x) + (其中三次骰出 x) - ...

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

- 假設 n 次骰出來的最大值為 x，那總共有多少可能呢?
- 排容!
- 答案其實就是 (其中一次骰出 x) - (其中兩次骰出 x) + (其中三次骰出 x) - ...
- 不過其實有更簡單的方式!
- 答案就是全部 - (沒有骰出 x)

例題

Codeforces 453A - Little Pony and Expected Minimum

Colten 有一個 m 面的均勻骰子，第 i 面上有 i 個點，接著他會骰這個骰子 n 次。而他會將取這 n 次骰出的最大值作為他的分數。請問他得到的分數期望值會是多少?

期望值的計算方式為 $\sum X_i \times P_i$

- 假設 n 次骰出來的最大值為 x，那總共有多少可能呢?
- 排容!
- 答案其實就是 (其中一次骰出 x) - (其中兩次骰出 x) + (其中三次骰出 x) - ...
- 不過其實有更簡單的方式!
- 答案就是全部 - (沒有骰出 x)
- 用一個迴圈跑過每種可能的 x 即可

例題

AtCoder Beginner Contest 178D - Redistribution

現在你有一個數字 S，請問有幾個不同的序列滿足其中所有的數字皆 ≥ 3，且該序列的總和為 S

例題

AtCoder Beginner Contest 178D - Redistribution

現在你有一個數字 S，請問有幾個不同的序列滿足其中所有的數字皆 ≥ 3，且該序列的總和為 S

- 我們可以直接去枚舉序列的長度
- 接著套 $S-3$ 次的重複組合公式即可

例題

Codeforces 1288C - Two Arrays

現在給你兩個數字 n, m，請找出總共有多少對的陣列 (A, B) 滿足

- A, B 的長度皆為 m
- $1 \leq A_i, B_i \leq n$
- 對於所有 i，$A_i \leq B_i$
- A 陣列是非遞減的 $(A_i \leq A_{i+1})$
- B 陣列是非遞增的 $(B_i \geq B_{i+1})$

例題

Codeforces 1288C - Two Arrays

現在給你兩個數字 n, m，請找出總共有多少對的陣列 (A, B) 滿足

- A, B 的長度皆為 m
- $1 \leq A_i, B_i \leq n$
- 對於所有 i，$A_i \leq B_i$
- A 陣列是非遞減的 $(A_i \leq A_{i+1})$
- B 陣列是非遞增的 $(B_i \geq B_{i+1})$

- 首先，我們觀察到 $A_n \leq B_n$，而且 A 是非遞減，B 是非遞增的
- 如果我們知道 A_i, B_i 會有哪些元素，則我們可以知道要怎麼分配這些數字

例題

Codeforces 1288C - Two Arrays

現在給你兩個數字 n, m，請找出總共有多少對的陣列 (A, B) 滿足

- A, B 的長度皆為 m
- $1 \leq A_i, B_i \leq n$
- 對於所有 i，$A_i \leq B_i$
- A 陣列是非遞減的 $(A_i \leq A_{i+1})$
- B 陣列是非遞增的 $(B_i \geq B_{i+1})$

- 首先，我們觀察到 $A_n \leq B_n$，而且 A 是非遞減，B 是非遞增的
- 如果我們知道 A_i, B_i 會有哪些元素，則我們可以知道要怎麼分配這些數字
- $A_1, \ldots, A_n, B_n, \ldots, B_1$ 會是一個非遞減的陣列

例題

Codeforces 1288C - Two Arrays

現在給你兩個數字 n, m，請找出總共有多少對的陣列 (A, B) 滿足

- A, B 的長度皆為 m
- $1 \leq A_i, B_i \leq n$
- 對於所有 i，$A_i \leq B_i$
- A 陣列是非遞減的 $(A_i \leq A_{i+1})$
- B 陣列是非遞增的 $(B_i \geq B_{i+1})$

- 首先，我們觀察到 $A_n \leq B_n$，而且 A 是非遞減，B 是非遞增的
- 如果我們知道 A_i, B_i 會有哪些元素，則我們可以知道要怎麼分配這些數字
- $A_1, \ldots, A_n, B_n, \ldots, B_1$ 會是一個非遞減的陣列
- 我們可以發現 $1 \sim n$ 之間的數字總共出現的頻率會是 $2m$

例題

Codeforces 1288C - Two Arrays

現在給你兩個數字 n, m，請找出總共有多少對的陣列 (A, B) 滿足

- A, B 的長度皆為 m
- $1 \leq A_i, B_i \leq n$
- 對於所有 i，$A_i \leq B_i$
- A 陣列是非遞減的 $(A_i \leq A_{i+1})$
- B 陣列是非遞增的 $(B_i \geq B_{i+1})$

- 首先，我們觀察到 $A_n \leq B_n$，而且 A 是非遞減，B 是非遞增的
- 如果我們知道 A_i, B_i 會有哪些元素，則我們可以知道要怎麼分配這些數字
- $A_1, \ldots, A_n, B_n, \ldots, B_1$ 會是一個非遞減的陣列
- 我們可以發現 $1 \sim n$ 之間的數字總共出現的頻率會是 $2m$
- $f_1 + f_2 + \cdots + f_n = 2m$ 的非負整數解數量!

例題

Codeforces 893E - Counting Arrays

給你兩個數字 x, y，請找出有幾個整數陣列 F 滿足

- F 的長度為 y
- $\prod_{i=1}^{y} F_i = x$

例題

Codeforces 893E - Counting Arrays

給你兩個數字 x, y，請找出有幾個整數陣列 F 滿足

- F 的長度為 y
- $\prod_{i=1}^{y} F_i = x$

- 將不同的質因數分開思考，假設 p 在 x 的質因數分解中，次方一共是 cnt_p 個
- 那我們可以把 cnt_p 分配到 y 個格子裡
- 直接使用重複組合的公式即可!

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 這題實際上的作法是動態規劃 + 矩陣快速冪
- 不過前年在比賽中遇到這題時，被我用了數學解掉了
- 讓大家思考看看這題要怎麼處理

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 這題 n 很大，不過我們先想想看，如果 $n \leq 10^3$ 要怎麼處理呢?

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 這題 n 很大，不過我們先想想看，如果 $n \leq 10^3$ 要怎麼處理呢?
- 枚舉國王與皇后的出現次數!

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 這題 n 很大，不過我們先想想看，如果 $n \leq 10^3$ 要怎麼處理呢?
- 枚舉國王與皇后的出現次數!
- 答案會是 $\sum_{0 \leq i \leq n, i \text{ odd}} \binom{n}{i} \sum_{0 \leq j \leq n-i, j \text{ odd}} \binom{n-i}{j} 4^{n-i-j}$

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- **如果 $n \leq 10^6$ 呢**?

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- **如果 $n \leq 10^6$ 呢?**
- **我們把裡面的東西抓出來看看**
- $$\sum_{0 \leq j \leq n-i, j \text{ odd}} \binom{n-i}{j} 4^{n-i-j}$$

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 如果 $n \leq 10^6$ 呢?
- 我們把裡面的東西抓出來看看
- $$\sum_{0 \leq j \leq n-i, j \text{ odd}} \binom{n-i}{j} 4^{n-i-j}$$
- 欸? 是不是很像二項式定理呢? $((1+x)^n = \sum_{i=0}^{n} \binom{n}{i} x^i)$

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 如果 $n \leq 10^6$ 呢?
- 我們把裡面的東西抓出來看看
- $$\sum_{0 \leq j \leq n-i, j \text{ odd}} \binom{n-i}{j} 4^{n-i-j}$$
- 欸? 是不是很像二項式定理呢? $((1+x)^n = \sum_{i=0}^{n} \binom{n}{i} x^i)$
- 其實上式可以被化簡為 $((1+4)^{n-i} + (1-4)^{n-i})/2$ (n 是偶數)
- 或 $((1+4)^{n-i} - (1-4)^{n-i})/2$ (n 是奇數)

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 因此在 $n \leq 10^6$ 時
- 直接套 $\sum_{0 \leq i \leq n, i \text{ odd}} \binom{n}{i} ((1+4)^{n-i} + (1-4)^{n-i})/2$ (n 是偶數)
- 或 $\sum_{0 \leq i \leq n, i \text{ odd}} \binom{n}{i} ((1+4)^{n-i} - (1-4)^{n-i})/2$ (n 是奇數) 即可

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 那在 $n \leq 10^9$ 的時候怎麼辦呢?

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 那在 $n \leq 10^9$ 的時候怎麼辦呢?
- 我們將剛剛化簡的式子再進一步地化簡 (在此省略 n 是奇數的 case)

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- **那在 $n \leq 10^9$ 的時候怎麼辦呢?**
- **我們將剛剛化簡的式子再進一步地化簡 (在此省略 n 是奇數的 case)**
- $\frac{1}{2}(\sum_{0\leq i\leq n, i \text{ odd}} \binom{n}{i}(1+4)^{n-i} + \sum_{0\leq i\leq n, i \text{ odd}} \binom{n}{i}(1-4)^{n-i})$

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 那在 $n \leq 10^9$ 的時候怎麼辦呢?
- 我們將剛剛化簡的式子再進一步地化簡 (在此省略 n 是奇數的 case)
- $\frac{1}{2}(\sum_{0\leq i\leq n, i \text{ odd}} \binom{n}{i}(1+4)^{n-i} + \sum_{0\leq i\leq n, i \text{ odd}} \binom{n}{i}(1-4)^{n-i})$
- 再套一次二項式定理!

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

$$\frac{1}{2}(\sum_{0\leq i\leq n, i \text{ odd}} \binom{n}{i}(1+4)^{n-i} + \sum_{0\leq i\leq n, i \text{ odd}} \binom{n}{i}(1-4)^{n-i})$$

$$\frac{1}{2}(((1+5)^n-(1-5)^n)/2 + ((1-3)^n-(1+3)^n)/2)$$

$$\frac{1}{4}((6^n-(-4)^n)+((-2)^n-4^n))$$

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 因此最後的一般式就是 $\frac{1}{4}(6^n - (-4)^n + (-2)^n - 4^n)$ (n 是偶數)
- 或 $\frac{1}{4}(6^n + (-4)^n - (-2)^n - 4^n)$ (n 是奇數)
- 只要使用快速冪就能在 $O(\log n)$ 計算完成了!

例題

2020 臺北市資訊學科能力競賽 pC

總共有 n 個格子，你可以在每一格上放上 6 種棋子 (國王、皇后、騎士、主教、城堡、士兵) 其中之一，請問總共有幾種放法可以讓國王與皇后的出現次數皆為奇數。

測資範圍: $n \leq 10^9$

- 因此最後的一般式就是 $\frac{1}{4}(6^n - (-4)^n + (-2)^n - 4^n)$ (n 是偶數)
- 或 $\frac{1}{4}(6^n + (-4)^n - (-2)^n - 4^n)$ (n 是奇數)
- 只要使用快速冪就能在 $O(\log n)$ 計算完成了!
- 我們學到了什麼? 有時候數學可以幫你解掉你不會寫的題目

含有莫比烏斯函數的排容

莫比烏斯函數

定義莫比烏斯函數為 $\mu(x)$

$$\mu(n) = \begin{cases} 1 & \text{, when } n = 1 \\ (-1)^k & \text{, when } n \text{ does not have a squared prime factor and } n = p_1 p_2 \cdots p_k \\ 0 & \text{, when } n \text{ has a squared prime factor} \end{cases}$$

莫比烏斯函數的性質

$$\sum_{d|n} \mu(d) = [n = 1]$$

莫比烏斯函數的性質

$$\sum_{d|n} \mu(d) = [n = 1]$$

證明

假設 n 有 k 個不同的質因數，則根據二項式定理

$$\sum_{d|n} \mu(d) = \binom{k}{0} - \binom{k}{1} + \cdots + \binom{k}{k} = (1 + (-1))^k = 0$$

含有莫比烏斯函數的排容

CSES - Counting Coprime Pairs

現在你有一個 n **項的陣列** $a_1, a_2, \leq, a_n$**，請找到有幾對** (i, j) **能使得** $gcd(a_i, a_j) = 1$**。**

測資範圍:

- $1 \leq n \leq 10^6$
- $1 \leq a_i \leq 10^6$

含有莫比烏斯函數的排容

CSES - Counting Coprime Pairs

現在你有一個 n 項的陣列 $a_1, a_2, \leq, a_n$，請找到有幾對 (i, j) 能使得 $gcd(a_i, a_j) = 1$。

測資範圍:

- $1 \leq n \leq 10^6$
- $1 \leq a_i \leq 10^6$

我們可以將最暴力的式子寫出來

$$\sum_{i=1}^{n} \sum_{j=1}^{n} [gcd(a_i, a_j) = 1]$$

含有莫比烏斯函數的排容

接著，我們來想想看如何優化這個式子

$$\sum_{i=1}^{n}\sum_{j=1}^{n}[gcd(a_i,a_j)=1]$$

根據剛剛的結論，我們可以將 $[gcd(a_i,a_j)=1]$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{d|a_i,d|a_j}\mu(d)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{d|a_i,d|a_j}[d|a_i][d|a_j]\mu(d)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{d|a_i,d|a_j}[d|a_i][d|a_j]\mu(d)$$

含有莫比烏斯函數的排容

將整個陣列中，d 的倍數寫成 $cnt[d]$ 的話，答案就會是

$$\sum_{i=1}^{n} \mu(d) \binom{cnt[d]}{2}$$

而這個我們就可以使用 $O(n \log n)$ 的枚舉完成這題了!

矩陣

矩陣 (Matrix)

矩陣是一個 n 列 m 行的結構，第 i 列第 m 行的元素會寫成 a_{ij}

$$\begin{bmatrix} a_{11} & \dots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nm} \end{bmatrix}$$

矩陣 (Matrix)

矩陣運算

1. 若 $C = A + B$，則 $c_{ij} = a_{ij} + b_{ij}$ (若 A, B 皆為 $n \times m$ 矩陣，則 C 也是 $n \times m$ 的矩陣)
2. 若 $C = A - B$，則 $c_{ij} = a_{ij} - b_{ij}$ (若 A, B 皆為 $n \times m$ 矩陣，則 C 也是 $n \times m$ 的矩陣)
3. 若 $C = AB$，則 $c_{ij} = \sum_{i=k}^{m} a_{ik} \cdot b_{kj}$

   (若 A 為 $n \times m$ 矩陣，B 為 $m \times p$ 矩陣，則 C 是 $n \times p$ 的矩陣)

矩陣滿足的性質

1. $(AB)C = A(BC)$
2. $(A + B)C = AC + BC$
3. $C(A + B) = CA + CB$
4. AB 不見得等於 BA

單位方陣 (Identity Matrix)

一個主對角線為 1，其他元素皆為 0 的方陣 ($n \times n$ 的矩陣)，特點是 $I_n A = A = A I_m$

$$I_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

零矩陣 (Zero Matrix)

對於一個 $n \times m$ 的矩陣 A，所有位置的元素皆為 0，與任何矩陣相乘皆為零矩陣

逆方陣 (Inverse Matrix)

對於一個 $n \times n$ 的矩陣 A，他的反矩陣 A^{-1} 會使得 $AA^{-1} = I_n$

高斯消去法 (Gaussian Elimination)

- 找 n 元一次聯立方程式方程式的解，該怎麼做?

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = & b_2 \\ \vdots & = & \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n & = & b_n \end{cases}$$

高斯消去法 (Gaussian Elimination)

- 找 n 元一次聯立方程式方程式的解，該怎麼做?

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ \vdots & = \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{cases}$$

- 我們可以將他寫成下列矩陣的形式:

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array}\right]$$

高斯消去法 (Gaussian Elimination)

列運算 (Row Operation)

1. 將兩列交換
2. 將某一列乘上某個常數 c
3. 將某一列乘上某個常數 c 加到另一列

- 藉由這三種列運算將矩陣消成階梯形矩陣 (Row-Echelon Form)
- 就可以很輕易地得到方程式的解了

高斯消去法 (Gaussian Elimination)

高斯消去法的步驟

1. 對於 x_i，如果 x_i 的係數為 0，從其他列換過來，係數皆為 0 的話，表示無解或無限多組解
2. 依序跑過，將其他列 x_i 的係數消掉
3. 往下一個 x_i 繼續消除
4. 如果最後產生了某一列皆為 0，則有無限多組解，若除了最後一項，其他皆為 0，則無解

- 我們手動來做一次看看
- 寫成程式碼後，這個演算法會是 $O(n^3)$

例題

2019 臺北市資訊學科能力競賽 pB - 地圖編修 (Map)

給你一個在 n 維坐標系下的座標，現在你要把坐標軸改成題目所給的 n 個向量，請找到以這 n 個向量作為坐標軸時的座標。

提示: 線性組合、變換基底

總結

今天我們介紹了各種競賽中可能會遇到的數學，事實上，還有更多更多可能會在競賽上遇到的數學，如果大家有機會的話也可以自己去學習更多不同的內容。