枚舉 Enumerate

2023 SCIST x NHDK x 南 11 校寒訓

Koying

2023-01-30

目錄

枚舉入門

枚舉入門

- 什麼是枚舉？

枚舉入門

- 什麼是枚舉？
- 簡單來說就是暴力解

枚舉入門

- 什麼是枚舉？
- 簡單來說就是暴力解
- 利用迴圈或是遞迴等最樸素的方法，將所有可能的情況，也就是「狀態」都列出來，以找到答案
- 可說是競程中最基本的技巧，許多的演算法都是由簡單的枚舉演變而來

枚舉入門

我們先來個簡單的例題

TPR 16E. 倒水問題

有一杯 N 毫升的水，每次可以倒出 a 或是 b 毫升，但這兩種操作都只能各自使用最多 10 次
請問哪個方案最多可以倒出多少毫升的水，且水杯內至少來剩下 K 毫升？如果有多種答案，輸出使用第一種操作最多次的方案

- 相信聰明的大家都可以想到，只需要枚舉 a 和 b 的使用次數，然後計算出最大的答案即可

我們先來個簡單的例題

TPR 16E. 倒水問題

有一杯 N 毫升的水，每次可以倒出 a 或是 b 毫升，但這兩種操作都只能各自使用最多 10 次
請問哪個方案最多可以倒出多少毫升的水，且水杯內至少來剩下 K 毫升？如果有多種答案，輸出使用第一種操作最多次的方案

- 相信聰明的大家都可以想到，只需要枚舉 a 和 b 的使用次數，然後計算出最大的答案即可
- 但，這就代表枚舉很簡單嗎？

我們先來個簡單的例題

TPR 16E. 倒水問題

有一杯 N 毫升的水，每次可以倒出 a 或是 b 毫升，但這兩種操作都只能各自使用最多 10 次
請問哪個方案最多可以倒出多少毫升的水，且水杯內至少來剩下 K 毫升？如果有多種答案，輸出使用第一種操作最多次的方案

- 相信聰明的大家都可以想到，只需要枚舉 a 和 b 的使用次數，然後計算出最大的答案即可
- 但，這就代表枚舉很簡單嗎？
- 事實上，雖然最基本的枚舉技巧很簡單，但為了獲得最好的效能，所以有非常多種優化方法值得我們去學習

枚舉入門

我們先來個簡單的例題

TPR 16E. 倒水問題

有一杯 N 毫升的水，每次可以倒出 a 或是 b 毫升，但這兩種操作都只能各自使用最多 10 次
請問哪個方案最多可以倒出多少毫升的水，且水杯內至少來剩下 K 毫升？如果有多種答案，輸出使用第一種操作最多次的方案

- 相信聰明的大家都可以想到，只需要枚舉 a 和 b 的使用次數，然後計算出最大的答案即可
- 但，這就代表枚舉很簡單嗎？
- 事實上，雖然最基本的枚舉技巧很簡單，但為了獲得最好的效能，所以有非常多種優化方法值得我們去學習
- 像講師我就是為了學枚舉才來當枚舉講師的

簡單的題目

給一整數 n $(n \leq 10^{15})$，求出 n 的所有因數

- 一個最樸素的方法便是枚舉所有數字，看有幾個數字能夠整除 n
- 而這樣的時間複雜度會是 $\mathcal{O}(n)$，這樣顯然太慢了

簡單的題目

給一整數 n $(n \leq 10^{15})$，求出 n 的所有因數

- 一個最樸素的方法便是枚舉所有數字，看有幾個數字能夠整除 n
- 而這樣的時間複雜度會是 $\mathcal{O}(n)$，這樣顯然太慢了
- 但有學過國中數學（? 的就會知道，只需要枚舉 $2 \sim \sqrt{n}$ 就可以了
- 因為只要求出 $\leq \sqrt{n}$ 的所有因數，那麼 $> \sqrt{n}$ 的因數也就可以找到

枚舉入門

簡單的題目

給一整數 n $(n \leq 10^{15})$，求出 n 的所有因數

- 一個最樸素的方法便是枚舉所有數字，看有幾個數字能夠整除 n
- 而這樣的時間複雜度會是 $\mathcal{O}(n)$，這樣顯然太慢了
- 但有學過國中數學（? 的就會知道，只需要枚舉 $2 \sim \sqrt{n}$ 就可以了
- 因為只要求出 $\leq \sqrt{n}$ 的所有因數，那麼 $> \sqrt{n}$ 的因數也就可以找到
- 如此一來時間複雜度就降到了 $\mathcal{O}(\sqrt{n})$

有限度的枚舉

有限度的枚舉

- 剛剛的例子，其實就是一種「有限度的枚舉」
- 透過一些數學技巧或是一些關鍵性值，將枚舉的範圍縮小，進而優化時間複雜度

有限度的枚舉

CF 1490C. Sum of Cubes

給一正整數 n $(n \leq 10^{12})$，求是否存在兩個正整數 a, b，使得 $a^3 + b^3 = n$

- 一樣先想樸素解，我們可以枚舉 a, b，是否符合條件，時間複雜度 $\mathcal{O}(10^{12^2})$

有限度的枚舉

CF 1490C. Sum of Cubes

給一正整數 n $(n \leq 10^{12})$，求是否存在兩個正整數 a, b，使得 $a^3 + b^3 = n$

- 一樣先想樸素解，我們可以枚舉 a, b，是否符合條件，時間複雜度 $\mathcal{O}(10^{12^2})$
- 有沒有辦法將範圍縮小呢？

有限度的枚舉

CF 1490C. Sum of Cubes

給一正整數 n $(n \le 10^{12})$，求是否存在兩個正整數 a, b，使得 $a^3 + b^3 = n$

- 一樣先想樸素解，我們可以枚舉 a, b，是否符合條件，時間複雜度 $\mathcal{O}(10^{12^2})$
- 有沒有辦法將範圍縮小呢？
- 我們可以發現，由於 $n \le 10^{12}$ 當 $a, b > 10^4$ 時，$a^3 + b^3$ 就會超出範圍
- 因此，a, b 的枚舉範圍就被縮小到了 10^4，時間複雜度 $\mathcal{O}(10^8)$，還是可能會 TLE

有限度的枚舉

CF 1490C. Sum of Cubes

給一正整數 n $(n \le 10^{12})$，求是否存在兩個正整數 a, b，使得 $a^3 + b^3 = n$

- 一樣先想樸素解，我們可以枚舉 a, b，是否符合條件，時間複雜度 $\mathcal{O}(10^{12^2})$
- 有沒有辦法將範圍縮小呢？
- 我們可以發現，由於 $n \le 10^{12}$ 當 $a, b > 10^4$ 時，$a^3 + b^3$ 就會超出範圍
- 因此，a, b 的枚舉範圍就被縮小到了 10^4，時間複雜度 $\mathcal{O}(10^8)$，還是可能會 TLE
- 我們可以發現，原本的數學式 $a^3 + b^3 = n$ 經過移項之後，可以變成 $n - a^3 = b^3$
- 因此，我們只需要枚舉 a，最後檢查 b 是否存在就可以了！時間複雜度 $\mathcal{O}(10^4)$，AC！
- 至於如何檢查 b 是否存在，則可以使用二分搜或是預先建立立方表的形式

有限度的枚舉

CF 1541B. Pleasant Pairs

給 n 個正整數 $a_1, a_2, \ldots, a_n$ $(1 \leq a_i \leq 2 \cdot n)$，且每個數都不相同，求出有多少對 (i, j) 滿足 $i < j, a_i \cdot a_j = i + j$

- 直接枚舉 i, j 的話複雜度 $\mathcal{O}(n^2)$，TLE

CF 1541B. Pleasant Pairs

給 n 個正整數 $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 2 \cdot n)$，且每個數都不相同，求出有多少對 (i, j) 滿足 $i < j, a_i \cdot a_j = i + j$

- 直接枚舉 i, j 的話複雜度 $\mathcal{O}(n^2)$，TLE
- 那這題有什麼關鍵性質嗎？

有限度的枚舉

CF 1541B. Pleasant Pairs

給 n 個正整數 $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 2 \cdot n)$，且每個數都不相同，求出有多少對 (i, j) 滿足 $i < j, a_i \cdot a_j = i + j$

- 直接枚舉 i, j 的話複雜度 $\mathcal{O}(n^2)$，TLE
- 那這題有什麼關鍵性質嗎？
- 觀察一下範圍，發現 $a_i \le 2 \cdot n$ 且每個數都不相同好像很可疑

CF 1541B. Pleasant Pairs

給 n 個正整數 $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 2 \cdot n)$，且每個數都不相同，求出有多少對 (i, j) 滿足 $i < j, a_i \cdot a_j = i + j$

- 直接枚舉 i, j 的話複雜度 $\mathcal{O}(n^2)$，TLE
- 那這題有什麼關鍵性質嗎？
- 觀察一下範圍，發現 $a_i \le 2 \cdot n$ 且每個數都不相同好像很可疑
- 再觀察一下式子，發現 $i + j$ 最多就是 $2n$

有限度的枚舉

CF 1541B. Pleasant Pairs

給 n 個正整數 $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 2 \cdot n)$，且每個數都不相同，求出有多少對 (i, j) 滿足 $i < j, a_i \cdot a_j = i + j$

- 直接枚舉 i, j 的話複雜度 $\mathcal{O}(n^2)$，TLE
- 那這題有什麼關鍵性質嗎？
- 觀察一下範圍，發現 $a_i \le 2 \cdot n$ 且每個數都不相同好像很可疑
- 再觀察一下式子，發現 $i + j$ 最多就是 $2n$
- 利用這個關鍵性質縮小枚舉的範圍，也就是對於任意一個 a_i，我們就只需要枚舉符合 $a_j \le \lceil \frac{2n}{a_i} \rceil$ 的 j 即可，根據調和級數，時間複雜度 $\mathcal{O}(n \log n)$

CF 1541B. Pleasant Pairs

給 n 個正整數 $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 2 \cdot n)$，且每個數都不相同，求出有多少對 (i, j) 滿足 $i < j, a_i \cdot a_j = i + j$

- 直接枚舉 i, j 的話複雜度 $\mathcal{O}(n^2)$，TLE
- 那這題有什麼關鍵性質嗎？
- 觀察一下範圍，發現 $a_i \le 2 \cdot n$ 且每個數都不相同好像很可疑
- 再觀察一下式子，發現 $i + j$ 最多就是 $2n$
- 利用這個關鍵性質縮小枚舉的範圍，也就是對於任意一個 a_i，我們就只需要枚舉符合 $a_j \le \lceil \frac{2n}{a_i} \rceil$ 的 j 即可，根據調和級數，時間複雜度 $\mathcal{O}(n \log n)$
- 至於如何找到適合的 j？簡單，pair / struct 排序！

例題

CF 1490C. Sum of Cubes

給一正整數 n $(n \leq 10^{12})$，求是否存在兩個正整數 a, b，使得 $a^3 + b^3 = n$

CF 1541B. Pleasant Pairs

給 n 個正整數 $a_1, a_2, \ldots, a_n$ $(1 \leq a_i \leq 2 \cdot n)$，且每個數都不相同，求出有多少對 (i, j) 滿足 $i < j, a_i \cdot a_j = i + j$

TPR 20A. 飲品調配

有三變數 a, b, c 滿足 $a + b + c = N, 0 \leq a, b, c \leq N, a, b, c \in \mathbb{Z}_0^+$，求 $2022 + |b - c| + ab + bc + c^2 - |b^2 - a^2|$

聰明的枚舉

聰明的枚舉

CSES Common Divisors

給 n 個數字 $x_1, x_2, ..., x_n$ $(n \leq 2 \times 10^5, x_i \leq 10^6)$
求出一個最大的數字 ans，滿足其為 x 中任意兩個數字 x_i, x_j 的最大公因數

- 枚舉 i, j，時間複雜度 $\mathcal{O}(n^2)$，TLE

聰明的枚舉

CSES Common Divisors

給 n 個數字 $x_1, x_2, ..., x_n$ $(n \leq 2 \times 10^5, x_i \leq 10^6)$
求出一個最大的數字 ans，滿足其為 x 中任意兩個數字 x_i, x_j 的最大公因數

- 枚舉 i, j，時間複雜度 $\mathcal{O}(n^2)$，TLE
- 這時候枚舉答案就派上用場了

聰明的枚舉

CSES Common Divisors

給 n 個數字 $x_1, x_2, ..., x_n$ $(n \leq 2 \times 10^5, x_i \leq 10^6)$
求出一個最大的數字 ans，滿足其為 x 中任意兩個數字 x_i, x_j 的最大公因數

- 枚舉 i, j，時間複雜度 $\mathcal{O}(n^2)$，TLE
- 這時候枚舉答案就派上用場了
- 假設有一個函數 $f(x)$ 代表 x 出現的次數
- 那麼對於一個數 a，其滿足 a 為 $\gcd(x_i, x_j)$ 的條件就是：$\sum_{k=1}^{\lceil \frac{10^6}{a} \rceil} f(ak) \geq 2$

CSES Common Divisors

給 n 個數字 $x_1, x_2, \ldots, x_n$ $(n \le 2 \times 10^5, x_i \le 10^6)$
求出一個最大的數字 ans，滿足其為 x 中任意兩個數字 x_i, x_j 的最大公因數

- 枚舉 i, j，時間複雜度 $\mathcal{O}(n^2)$，TLE
- 這時候枚舉答案就派上用場了
- 假設有一個函數 $f(x)$ 代表 x 出現的次數
- 那麼對於一個數 a，其滿足 a 為 $\gcd(x_i, x_j)$ 的條件就是：$\sum_{k=1}^{\lceil \frac{10^6}{a} \rceil} f(ak) \ge 2$
- 根據調和級數，$\sum_{i=1}^{n} \frac{n}{i} \approx n\log n$，AC！

聰明的枚舉

CSES Common Divisors

給 n 個數字 $x_1, x_2, \ldots, x_n$ $(n \leq 2 \times 10^5, x_i \leq 10^6)$
求出一個最大的數字 ans，滿足其為 x 中任意兩個數字 x_i, x_j 的最大公因數

- 枚舉 i, j，時間複雜度 $\mathcal{O}(n^2)$，TLE
- 這時候枚舉答案就派上用場了
- 假設有一個函數 $f(x)$ 代表 x 出現的次數
- 那麼對於一個數 a，其滿足 a 為 $\gcd(x_i, x_j)$ 的條件就是：$\sum_{k=1}^{\lceil \frac{10^6}{a} \rceil} f(ak) \geq 2$
- 根據調和級數，$\sum_{i=1}^{n} \frac{n}{i} \approx n\log n$，AC！
- 至於 $f()$ 的計算，則是利用陣列即可

例題

CSES Common Divisors

給 n 個數字 $x_1, x_2, ..., x_n$ $(n \le 2 \times 10^5, x_i \le 10^6)$
求出一個最大的數字 ans，滿足其為 x 中任意兩個數字 x_i, x_j 的最大公因數

ARC 123B

給兩個數列 a, b，

CF 1494B. Berland Crossword

給 5 個正整數 n, U, D, R, L，代表這是一個 $n \cdot n$ ，由黑白格子組成的棋盤，最上面那排有 U 個黑格子、最下面那排有 D 個 $\ldots$ 以此類推
求任意一種塗色方法是否能滿足條件

位元枚舉

位元

- 首先，我們得先了解一下二進位制
- 二進位的每一位都是 `0` 或 `1`，分別代表 `2` 的每個冪次是有還是沒有
- 舉個例子：$5 = 101_{(2)}$ 意思就是他是 $1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$ 組成的

位元

- 首先，我們得先了解一下二進位制
- 二進位的每一位都是 `0` 或 `1`，分別代表 `2` 的每個冪次是有還是沒有
- 舉個例子：$5 = 101_{(2)}$ 意思就是他是 $1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$ 組成的
- 我們經常利用二進位來表示第 i 個元素的狀態，例如 $101_{(2)}$ 就代表第 $1, 3$ 個元素是有的，第 2 個元素是沒有的
- 這種特性剛好能夠用來處理枚舉中「需要列出所有狀態」的問題

位元的各種操作

- 2^i：`1 << i`，`<<` 代表的是左移，也就是乘上 2
- 取出第 i 位的狀態：`x & (1 << i)`，如果第 i 位是 1，則會回傳 2^i，否則為 0
- 將第 i 位設為 1：`x |= (1 << i)`
- 將第 i 位設為 0：`x &= ~(1 << i)`

CSES 1623 Apple Division

你有 $n, (n \leq 20)$ 顆蘋果，每顆蘋果的重量為 p_i，請將這些蘋果分成兩堆，使得兩堆的重量差最小

- 不難觀察到，蘋果只有兩種狀態：在第一堆或是第二堆

CSES 1623 Apple Division

你有 $n, (n \leq 20)$ 顆蘋果，每顆蘋果的重量為 p_i，請將這些蘋果分成兩堆，使得兩堆的重量差最小

- 不難觀察到，蘋果只有兩種狀態：在第一堆或是第二堆
- 以 0 代表在第一堆，1 代表在第二堆

CSES 1623 Apple Division

你有 $n, (n \le 20)$ 顆蘋果，每顆蘋果的重量為 p_i，請將這些蘋果分成兩堆，使得兩堆的重量差最小

- 不難觀察到，蘋果只有兩種狀態：在第一堆或是第二堆
- 以 0 代表在第一堆，1 代表在第二堆
- 對於每種狀態，利用迴圈算出兩堆的重量差，並更新答案，時間複雜度 $\mathcal{O}(2^n)$

位元枚舉

CSES 1623 Apple Division

你有 $n, (n \leq 20)$ 顆蘋果，每顆蘋果的重量為 p_i，請將這些蘋果分成兩堆，使得兩堆的重量差最小

- 不難觀察到，蘋果只有兩種狀態：在第一堆或是第二堆
- 以 0 代表在第一堆，1 代表在第二堆
- 對於每種狀態，利用迴圈算出兩堆的重量差，並更新答案，時間複雜度 $\mathcal{O}(2^n)$
- 啪！AC，位元枚舉就是這麼簡單

關於位元枚舉

- 通常位元枚舉的複雜度都是 $\mathcal{O}(2^{\mathrm{n}})$（如果狀態是 0、1、2 的話可能會是 3^{n}），因此看到 $\mathrm{n} \leq 25$ 都可能是位元枚舉的題目
- 位元枚舉也經常配合其他演算法，像是位元 DP 等
- 有時候也能使用枚舉 II 會講到的遞迴枚舉來代替位元枚舉，只是遞迴的常數會較大

例題

CSES 1623 Apple Division

你有 n 顆蘋果，每顆蘋果的重量為 p_i，請將這些蘋果分成兩堆，使得兩堆的重量差最小

ABC 197C - ORXOR

對於一個有 n 個數字的數列 a $(n \le 20, a_i \le 2^30)$，請將其分為數個連續區間，先將每個區間內的元素做 OR 運算，再將所有區間的結果做 XOR 運算，使得最後的結果最大

ABC 100D - Patisserie ABC

總共有 n 個蛋糕，對於第 i 個蛋糕，有 3 個數值 x_i, y_i, z_i，請選出 m 個蛋糕，使得 $|\sum x_i| + |\sum y_i| + |\sum z_i|$ 最大

例題

ZJ F162 (108 全國能競 P4)

見原題

NHDK TPR 17D. Coding 夢之國的過年分配問題

位元枚舉作為子題解法的例子

折半枚舉

AP325 子集合的和

有一個長度 ≤ 38 的數列 A，請求出一個子集合 S 使得 S 的和最接近 P

- 前面提到，位元枚舉適用於 $n \leq 25$ 的情況，那如果 $n \leq 40$ 呢？

1. 將數列分成兩半
2. 分別位元枚舉，並將前半的結果記錄下來
3. 利用二分搜，為後半找到最好的前半

折半枚舉

AP325 子集合的和

有一個長度 ≤ 38 的數列 A，請求出一個子集合 S 使得 S 的和最接近 P

- 前面提到，位元枚舉適用於 $n \leq 25$ 的情況，那如果 $n \leq 40$ 呢？
- 這時候就是折半枚舉派上用場的地方了，步驟如下：

1. 將數列分成兩半
2. 分別位元枚舉，並將前半的結果記錄下來
3. 利用二分搜，為後半找到最好的前半

折半枚舉

- 折半枚舉算是一個比較不特別的技巧，因此在比賽中也比較少獨立出現
- 但折半枚舉經常會搭配一些 STL，或者是出現在子任務中，學會這個技巧經常可以在一些關鍵時刻派上用場

例題

AP325 子集合的和

有一個長度 ≤ 38 的數列 A，請求出一個子集合 S 使得 S 的和最接近 P

CSES Meet in the Middle

有 n 個數字 $t_1, t_2, \ldots, t_n (n \leq 40)$，求有幾種子集 S 滿足 $\sum S = x$

CF 888E. Maximum Subsequence

有 n 個元素 $(n \leq 35)$，求一個子集合 b 滿足 $\sum b_i$ mod m 最大

CF 1006F. Xor-Paths

有一張 $n \times m$ $(n, m \leq 20)$ 的方格圖，每個格子上都有一個數字，求 $(1, 1) \to (n, m)$ 途中經過所有數字 xor 起來的值最大

全排列枚舉

全排列枚舉

- 全排列指的是將一個數列或是元素的所有排列方式
- 例如 $\{1, 2, 3\}$ 的全排列就有 6 種

全排列枚舉

- 全排列指的是將一個數列或是元素的所有排列方式
- 例如 $\{1, 2, 3\}$ 的全排列就有 6 種
- 而 C++ 中有兩種函式可以幫助我們產生全排列
  - next_permutation(begin, end)：字典序由小到大生成
  - prev_permutation(begin, end)：字典序由大到小生成

全排列枚舉

- 全排列指的是將一個數列或是元素的所有排列方式
- 例如 $\{1, 2, 3\}$ 的全排列就有 6 種
- 而 `C++` 中有兩種函式可以幫助我們產生全排列
  - `next_permutation(begin, end)`：字典序由小到大生成
  - `prev_permutation(begin, end)`：字典序由大到小生成
- 雖然不常有滿分解是全排列枚舉的題目，但是是個很好的拿分技巧

例題

ZJ e446. 排列生成

排列出 $1 \sim \mathrm{N}$ 的所有排列，依照字典序由小到大排列

TPR 12H2. 奇數偶數全排列

請見原題，利用全排列枚舉寫出簡短的答案