枚舉

陳俊安 Colten

2022 新化高中 x 嘉義高中 x 薇閣高中 資研社暑期培訓營隊

2022.07.05

自我介紹

- 陳俊安 Colten
- 111 特殊選才錄取成大資工（甲組）
- 第 37 屆台南女中資訊研究社 C++ 進階班講師
- 我 APCS 雖然實作滿級，但我觀念只有 4，我太笨了
- 營隊期間有不會的東西或是有問題歡迎隨時找我
- 然後名義上我是負責 E 組的，但是其實我是負責 A、B、C、D、E、F 的

枚舉

枚舉

- 俗稱硬幹
- 把所有組合與可能全部跑過一次
- 最基本的技巧，如果你能用最簡單的方式取得 AC 那為什麼不好好把握呢？

枚舉例題

給你一個正整數 x $(1 \leq x \leq 1000)$，請你找到共有幾組非負整數 a, b 滿足 $a \times b = x$

枚舉

枚舉例題

給你一個正整數 x $(1 \leq x \leq 1000)$，請你找到共有幾組非負整數 a, b 滿足 $a \times b = x$

大家可能第一時間的直覺是用兩個迴圈來枚舉 a, b，但這樣時間複雜度為 $O(x^2)$，但我們有需要枚舉這麼多次嗎？

枚舉

枚舉例題

給你一個正整數 x $(1 \leq x \leq 10^6)$，請你找到共有幾組非負整數 a, b 滿足 $a \times b = x$

有學員能想出一個時間複雜度只需要 $O(x)$ 的做法嗎？沒有獎品

枚舉例題

給你一個正整數 x $(1 \leq x \leq 10^6)$，請你找到共有幾組非負整數 a, b 滿足 $a \times b = x$

觀察一下你會發現我們只需要枚舉 a 或 b 其中一個就可以了，原因是我們知道其中一個是誰，就可以知道另外一個人應該要是多少

舉個例子來說，假設 $x = 10$，那當 $a = 2$ 時，我們就可以確定 $b = 5$，如此一來我們就可以只用一個迴圈來枚舉 a，藉由 a 來得出 b，時間複雜度為 $O(x)$

APCS 2020 06 月第一題: 人力分配

有一間工廠有 n 個員工，現在你要把這些員工分配到兩個工廠裡面，每一個人一定要在其中一個工廠，如果第一個工廠有 X_1 人，那麼第一間工廠的收益為 $A_1 \times X_1^2 + B_1 \times X_1 + C_1$，如果第二個工廠有 X_2 人，那麼第二間工廠的收益是 $A_2 \times X_2^2 + B_2 \times X_2 + C_2$，給定 $n, A_1, B_1, C_1, A_2, B_2, C_2$ 且 $(1 \leq n \leq 100)$ 求兩間工廠總收益的最大值

你會發現這題可以 $O(n^2)$ 輕鬆的做掉，但如果用我們前面那一個例題的概念，實際上可以做到 $O(n)$

枚舉

```
14      int a,b,c,d,e,f,n,i,k,ans=-1e9;
15
16      cin >> a >> b >> c >> d >> e >> f >> n;
17
18      for(i=0;i≤n;i++)
19      {
20          int k = n - i;
21
22          int u = a * (i*i) + b * i + c;
23
24          int y = d * (k*k) + e * k + f;
25
26          ans = max(ans,u+y);
27      }
28
29      cout << ans << "\n";
30  }
```

有限度的枚舉

Ten Point Round #20 台南女中資訊研究社進階班練習賽 pA. 飲品調配

給定一個正整數 N $(1 \leq N \leq 5000)$，選定三個非負整數 a, b, c 滿足 $(0 \leq a, b, c \leq N)$ 且 $a + b + c = N$，找到 $2022 + |b - c| + ab + bc + c^2 - |b^2 - a^2|$ 的最大值

有限度的枚舉

Ten Point Round #20 台南女中資訊研究社進階班練習賽 pA. 飲品調配

給定一個正整數 N $(1 \le N \le 5000)$，選定三個非負整數 a, b, c 滿足 $(0 \le a, b, c \le N)$ 且 $a + b + c = N$，找到 $2022 + |b - c| + ab + bc + c^2 - |b^2 - a^2|$ 的最大值

如果你乖乖的枚舉 a, b, c 每一個的值那就太急了 >< 先估估看這樣做的時間複雜度會是多少，很明顯會是 $O(N^3)$

有限度的枚舉

Ten Point Round #20 台南女中資訊研究社進階班練習賽 pA. 飲品調配

給定一個正整數 N $(1 \leq N \leq 5000)$，選定三個非負整數 a, b, c 滿足 $(0 \leq a, b, c \leq N)$ 且 $a + b + c = N$，找到 $2022 + |b - c| + ab + bc + c^2 - |b^2 - a^2|$ 的最大值

那用我們之前的那一個概念，如果我們只枚舉 a, b 是不是就能知道 c 是多少了呢？
如此一來時間複雜度就會是 $O(N^2)$ 了！可以順利的通過這一題

Ten Point Round #20 台南女中資訊研究社進階班練習賽 pA. 飲品調配

給定一個正整數 N $(1 \le N \le 5000)$，選定三個非負整數 a, b, c 滿足 $(0 \le a, b, c \le N)$ 且 $a + b + c = N$，找到 $2022 + |b - c| + ab + bc + c^2 - |b^2 - a^2|$ 的最大值

其實這一題有 $O(1)$ 的做法，但不好觀察

有限度的枚舉

Codeforces Round #703 (Div. 3) pC. Sum of Cubes

有 Q $(1 \leq Q \leq 100)$ 詢問，每組詢問給定一個正整數 x $(1 \leq x \leq 10^{12})$，詢問是否找的到兩個正整數 $a, b (1 \leq a, b)$ 滿足 $a^3 + b^3 = x$

有限度的枚舉

Codeforces Round #703 (Div. 3) pC. Sum of Cubes

有 Q $(1 \le Q \le 100)$ 詢問，每組詢問給定一個正整數 x $(1 \le x \le 10^{12})$，詢問是否找的到兩個正整數 $a, b (1 \le a, b)$ 滿足 $a^3 + b^3 = x$

我們可以枚舉 a 來得出 $b = x - a^3$，因此很明顯 $a \le x^{\frac{1}{3}}$，所以我們只需要枚舉到 $x^{\frac{1}{3}}$
接下來我們要判斷 $x - a^3$（也就是 b）是否為完全立方數，那我們應該要怎麼判斷呢？
有一個經典的做法是二分搜（Binary Search），這一個技巧我們後面一點會教

有限度的枚舉

Codeforces Round #703 (Div. 3) pC. Sum of Cubes

有 Q $(1 \le Q \le 100)$ 詢問，每組詢問給定一個正整數 x $(1 \le x \le 10^{12})$，詢問是否找的到兩個正整數 $a, b (1 \le a, b)$ 滿足 $a^3 + b^3 = x$

- `cmath` 這一個標頭檔裡有的好用的工具是 `cbrt(n)` 可以把 n 直接開立方根，或是你也可以使用 `pow(n,1/3)`
- 開完立方根之後是一個浮點數
- 我們可以確定如果有一個數字是完全立方數，那麼那一個數字被開立方根出來之後小數點後一定都是 0
- 所以如果我們把 `cbrt`$(x - a^3)$ 的結果強制轉型成整數之後，檢查強制轉型後的結果的立方回去是不是 $x - a^3$ 就可以知道 $x - a^3$ 是不是完全立方數了
- 換句話說，如果 `cbrt`$(x - a^3)$ 被強制轉型成整數的結果為 p，那麼我就檢查 p^3 是否等於 $x - a^3$，就可以知道 $x - a^3$ 是不是完全立方數了

有限度的枚舉

Codeforces Round #703 (Div. 3) pC. Sum of Cubes

有 Q $(1 \le Q \le 100)$ 詢問，每組詢問給定一個正整數 x $(1 \le x \le 10^{12})$，詢問是否找的到兩個正整數 $a, b (1 \le a, b)$ 滿足 $a^3 + b^3 = x$

```
15      for(int a=1;a*a*a≤x;a++)
16      {
17          int b = x - a;
18          int u = cbrt(b); // 強制轉型成整數
19
20          // 檢查 u * u * u
21      }
```

特別注意：由於我是一個習慣 `#define int long long` 的人，因此我開 `int` 就自動會有 `long long` 的效果，在寫的時候記得注意範圍跟 `long long` 的問題

有限度的枚舉

```
21          bool ans = false;
22
23          for(int i=1;i*i*i≤n;i++)
24          {
25              int u = i * i * i;
26              int u2 = n - u;
27              int h = cbrt(u2);
28
29              if( h * h * h ≠ u2 )
30              {
31                  continue;
32              }
33              if( u2 == 0 )
34              {
35                  continue;
36              }
37
38              ans = 1;
39              break;
40          }
41
42          cout ≪ ( ans == 1 ? "Yes\n" : "No\n" );
```

練習題

Ten Point Round #7 pD. 三個骰子

你有三個六面骰，骰子的點數為 $1 \sim 6$，給你一個正整數 n $(1 \leq n \leq 18)$，請你求出共有幾種可能會使三個骰子的點數和為 n

Ten Point Round #16 台南女中資訊研究社期末測驗 pE. 倒水問題

Colten 現在有一杯 n 毫升的水，有兩種操作，每次操作可以將水倒出 a 毫升，或是將水倒出 b 毫升，這兩種操作可以各使用 10 次，倒完水後剩下的水量不能少於 K 毫升，定義一個值叫做所剩差距，如果 Colten 剩下的水量是 d 毫升，那麼所剩差距為 $d - K$，Colten 希望所剩差距越小越好，請你幫他找到最小的所剩差距，以及他應該倒幾次 a 毫升的水與倒幾次 b 毫升的水，如果有多種方式可以達到最小的所剩差距，那麼 Colten 會優先選擇第一種操作比較多的方式

迴圈枚舉進階題

Codeforces Round #735 pB. Cobb

給你一個長度為 n 的序列 a 與一個正整數 k，你可以選擇兩個不同的正整數 (i, j) $(1 \leq i < j \leq n)$，請你找到 $i \times j - k \times (a_i | a_j)$ 的最大值。
$(0 \leq a_i \leq n), (1 \leq k \leq \textbf{min}(k, 100)), (2 \leq n \leq 10^5)$

枚舉的時間複雜度會是 $O(n^2)$，很明顯是不行的。

迴圈枚舉進階題

Codeforces Round #735 pB. Cobb

給你一個長度為 n 的序列 a 與一個正整數 k，你可以選擇兩個不同的正整數 (i,j) $(1 \le i < j \le n)$，請你找到 $i \times j - k \times (a_i | a_j)$ 的最大值。
$(0 \le a_i \le n), (1 \le k \le \mathbf{min}(k, 100)), (2 \le n \le 10^5)$

這題 k 的範圍很特別，我們來觀察看看有什麼特別的東西吧！

我們先用最直接的想法，如果我們想要數值越大越好，那我們先用 $i = n-1, j = n$ 這兩個來試試看

迴圈枚舉進階題

Codeforces Round #735 pB. Cobb

給你一個長度為 n 的序列 a 與一個正整數 k，你可以選擇兩個不同的正整數 (i,j) $(1 \le i < j \le n)$，請你找到 $i \times j - k \times (a_i | a_j)$ 的最大值。
$(0 \le a_i \le n), (1 \le k \le \mathbf{min}(k, 100)), (2 \le n \le 10^5)$

- $a_i | a_j < 2n$

證明：以二進位的角度來思考，如果 $a_i = n$，且這個 n 是二的冪次，那就表示他的二進位的最左邊第一位是 1 後面全部都是 0，那如果現在 $a_j = n - 1$ 就表示 $n - 1$ 的二進位全部都會是 1 且總位數比 a_i 的二進位少一位，那麼 OR 起來後，會使 a_i 二進位中後面所有的 0 都轉成 1，那麼如果原本 $a_i = 2^k$，那麼被 OR 完之後就變成了 $2^{k+1} - 1$，由此可證 $a_i | a_j < 2n$

Codeforces Round #735 pB. Cobb

給你一個長度為 n 的序列 a 與一個正整數 k，你可以選擇兩個不同的正整數 (i,j) $(1 \le i < j \le n)$，請你找到 $i \times j - k \times (a_i | a_j)$ 的最大值。
$(0 \le a_i \le n), (1 \le k \le \mathbf{min}(k, 100)), (2 \le n \le 10^5)$

- 既然 $a_i | a_j < 2n$，我們就用最極端的方式來看，先假設他是 $2n$，會讓我們的算式比較乾淨一點
- 我們原本選擇 $i = n-1, j = n$，代入題目給的式子得到的結果會是 $n(n-1) - 2kn = n^2 - n - 2kn$
- 那麼如果我們要找到另一組 (i, n) 比原本我們選的越好，那就表示 $in - k(a_i | a_n) > n^2 - n - 2kn$
- 我們假設我們能找到其中一組 $a_i | a_n = 0$
- 代入式子後： $in - k \times 0 > n^2 - n - 2kn$

迴圈枚舉進階題

Codeforces Round #735 pB. Cobb

給你一個長度為 n 的序列 a 與一個正整數 k，你可以選擇兩個不同的正整數 (i,j) $(1 \le i < j \le n)$，請你找到 $i \times j - k \times (a_i | a_j)$ 的最大值。
$(0 \le a_i \le n), (1 \le k \le \mathbf{min}(k, 100)), (2 \le n \le 10^5)$

- 代入式子後： $in - k \times 0 > n^2 - n - 2kn$
- 整理一下式子可以得到 $in > n^2 - n - 2kn$ 或是 $i > n - 2k - 1$，所以你可以發現如果 $j = n$，那麼 $i \le n - 2k - 1$ 時答案一定不會比我們選 $(n-1, n)$ 還要來的好，所以我們可以非常確定我們所選擇的 i, j 要滿足 $i, j > n - 2k - 1$
- 所以我們就只需要枚舉 $n - 2k \sim n$ 就可以了，時間複雜度 $O(k^2)$

迴圈枚舉進階題

```
13      int q;
14      cin >> q;
15      while(q--)
16      {
17          int n,k;
18          cin >> n >> k;
19          vector<int> a(n);
20          for(int i=0;i<n;i++) cin >> a[i];
21
22          int ans = -1e18;
23
24          for(int i=max(0LL,n-2*k-1);i<n;i++)
25          {
26              for(int j=i+1;j<n;j++)
27              {
28                  ans = max(ans,(i+1)*(j+1)-k*(a[i]|a[j]));
29              }
30          }
31
32          cout << ans << "\n";
33      }
```

Codeforces Round #728 pB. Pleasant Pairs

給一一組長度為 n $(2 \le n \le 10^5)$ 的序列 a，元素 a_i 滿足 $1 \le a_i \le 2n$，請你找到有幾組 (i, j) $(1 \le i < j \le n)$ 滿足 $a_i \times a_j = i + j$

大家來試著想想這一題吧！

Codeforces Round #728 pB. Pleasant Pairs

給一一組長度為 n $(2 \le n \le 10^5)$ 的序列 a，元素 a_i 滿足 $1 \le a_i \le 2n$，請你找到有幾組 (i, j) $(1 \le i < j \le n)$ 滿足 $a_i \times a_j = i + j$

- $i + j < 2n$

Codeforces Round #728 pB. Pleasant Pairs

給一一組長度為 n $(2 \le n \le 10^5)$ 的序列 a，元素 a_i 滿足 $1 \le a_i \le 2n$，請你找到有幾組 (i, j) $(1 \le i < j \le n)$ 滿足 $a_i \times a_j = i + j$

- $i + j < 2n$
- 最多只會有 $\frac{2n}{a_i}$ 個 j 會使 $a_i \times a_j \le 2n$

Codeforces Round #728 pB. Pleasant Pairs

給一組長度為 n $(2 \le n \le 10^5)$ 的序列 a，元素 a_i 滿足 $1 \le a_i \le 2n$，且序列中每一個元素都不重複出現，請你找到有幾組 (i, j) $(1 \le i < j \le n)$ 滿足 $a_i \times a_j = i + j$

- $i + j < 2n$
- 最多只會有 $\frac{2n}{a_i}$ 個 j 會使 $a_i \times a_j \le 2n$
- 每一個元素都不重複出現
- 在最壞的情況下我們會需要跑 $\frac{2n}{1} + \frac{2n}{2} + ... + \frac{2n}{n}$ 次迴圈
- 式子整理一下後會得到 $2n(1 + \frac{1}{1} + \frac{1}{2} + ... + \frac{1}{n}) \approx n \log n$

枚舉關鍵點

枚舉關鍵點

Educational Codeforces Round 105 pB. Berland Crossword

給定五個非負整數 n, U, D, L, R $(1 \leq n \leq 100, 0 \leq U, D, L, R \leq n)$，你現在可以在一張 $n \times n$ 全白的的網格上選擇一些格子把它塗黑，詢問是否有辦法塗成最上排有 U 格是黑的、最左排有 L 格是黑的、最下排有 D 格是黑的、最右排有 R 格是黑的

先試著想想看這題的關鍵在哪裡

枚舉關鍵點

Educational Codeforces Round 105 pB. Berland Crossword

給定五個非負整數 n, U, D, L, R $(1 \leq n \leq 100, 0 \leq U, D, L, R \leq n)$，你現在可以在一張 $n \times n$ 全白的的網格上選擇一些格子把它塗黑，詢問是否有辦法塗成最上排有 U 格是黑的、最左排有 L 格是黑的、最下排有 D 格是黑的、最右排有 R 格是黑的

- 很明顯這題的關鍵會是在網格角落的那四個點
- 我們只需要枚舉這四個格子有沒有塗成黑色就可以簡單的判斷是否存在一種方式達成題目要求了

練習題

Ten Point Round #12 pH2. 奇數偶數全排列 (Hard Version)

此題敘述較複雜，請到原題目觀看完整題敘

CodeCraft-17 and Codeforces Round #391 pB. Bash's Big Day

給你一個長度為 n $(1 \leq n \leq 10^5)$ 的序列 a，你要從這一個序列中選一些數字，這些數字的最大公因數不能是 1，請問你最多能選幾個數字

Atcoder Beginner Contest 100 pD. Patisserie ABC

給你三個長度為 n $(1 \leq n \leq 1000)$ 的序列 x, y, z，選擇 m $(0 \leq m \leq n)$ 個正整數 t_i，找出 $|\sum_{i=1}^{m} x_{t_i}| + |\sum_{i=1}^{m} y_{t_i}| + |\sum_{i=1}^{m} z_{t_i}|$ 的最大值

(提示：絕對值前的正負跟每個元素帶給的貢獻度)

位元枚舉

位元枚舉

- 使用進位制維護枚舉的技巧
- 優點是在某些情況下使用位元枚舉可以省略掉遞迴，讓時間複雜度的常數變小
- 不過會比一般遞迴多一個拆解二進位的時間，因此在數字稍微大的時候還是遞迴會比較快
- 好實作，程式碼量短
- 最方便的使用情況是在枚舉的部分只有 `0/1` 兩種可能的時候，由於我們在程式中做二進位的運算非常方便

CSES Problem Set - Apple Divison

你現在有 n $(1 \leq n \leq 20)$ 個蘋果，每一個蘋果都有一個重量 p_i $(1 \leq p_i \leq 10^9)$，現在有兩個籃子，你必須把所有蘋果都放到這兩個籃子裡面，你可以選擇每一顆蘋果要放到哪一個籃子裡，但最後這兩個籃子的總重量要越接近越好，詢問最後這兩個籃子重量的最小差

位元枚舉

CSES Problem Set - Apple Divison

你現在有 n $(1 \leq n \leq 20)$ 個蘋果，每一個蘋果都有一個重量 p_i $(1 \leq p_i \leq 10^9)$，現在有兩個籃子，你必須把所有蘋果都放到這兩個籃子裡面，你可以選擇每一顆蘋果要放到哪一個籃子裡，但最後這兩個籃子的總重量要越接近越好，詢問最後這兩個籃子重量的最小差

- n 超級小
- 最多只會有 2^n 種狀態
- 一定可以枚舉！
- 每一種蘋果只有兩種可能，不是放第一個籃子就是放第二個籃子
- 我們用二進位來維護我們的狀態，0 表示放第一個籃子，1 表示放第二個籃子

CSES Problem Set - Apple Divison

你現在有 n $(1 \leq n \leq 20)$ 個蘋果，每一個蘋果都有一個重量 p_i $(1 \leq p_i \leq 10^9)$，現在有兩個籃子，你必須把所有蘋果都放到這兩個籃子裡面，你可以選擇每一顆蘋果要放到哪一個籃子裡，但最後這兩個籃子的總重量要越接近越好，詢問最後這兩個籃子重量的最小差

- 最多只會有 2^n 種可能，2^n 轉成二進位後剛好有 $n+1$ 位數字
- 2^n-1 轉成二進位後剛好有 n 位數字，且這 n 位數字全部都是 1

位元枚舉

CSES Problem Set - Apple Divison

你現在有 n $(1 \leq n \leq 20)$ 個蘋果，每一個蘋果都有一個重量 p_i $(1 \leq p_i \leq 10^9)$，現在有兩個籃子，你必須把所有蘋果都放到這兩個籃子裡面，你可以選擇每一顆蘋果要放到哪一個籃子裡，但最後這兩個籃子的總重量要越接近越好，詢問最後這兩個籃子重量的最小差

- 最多只會有 2^n 種可能，2^n 轉成二進位後剛好有 $n+1$ 位數字
- 2^n-1 轉成二進位後剛好有 n 位數字，且這 n 位數字全部都是 1
- 我們從 0 跑到 2^n-1 剛好有 $(2^n-1)-0+1=2^n$ 種不同的二進位數字
- 相當於我們從所有蘋果都放第一個籃子枚舉到所有蘋果都放在第二個籃子

位元枚舉

CSES Problem Set - Apple Divison

你現在有 n $(1 \leq n \leq 20)$ 個蘋果，每一個蘋果都有一個重量 p_i $(1 \leq p_i \leq 10^9)$，現在有兩個籃子，你必須把所有蘋果都放到這兩個籃子裡面，你可以選擇每一顆蘋果要放到哪一個籃子裡，但最後這兩個籃子的總重量要越接近越好，詢問最後這兩個籃子重量的最小差

- 最多只會有 2^n 種可能，2^n 轉成二進位後剛好有 $n+1$ 位數字
- 2^n-1 轉成二進位後剛好有 n 位數字，且這 n 位數字全部都是 1
- 我們從 0 跑到 2^n-1 剛好有 $(2^n-1)-0+1=2^n$ 種不同的二進位數字
- 相當於我們從所有蘋果都放第一個籃子枚舉到所有蘋果都放在第二個籃子
- 所以我們只要從 0 跑到 2^n-1，然後去看每一個數字的二進位的長相，就可以用來表示當前所有蘋果的狀態了，而這些狀態剛好有 2^n 種，所以就等價於我們將所有蘋果分配的可能性都枚舉了一遍
- 時間複雜度 $O(2^n \cdot n)$

位元枚舉

```
25      int ans = 1e18;
26
27      for(int i=0;i<( 1 << n );i++)
28      {
29          int num = i,index = 0,total = 0;
30
31          while( num =/= 0 )
32          {
33              if( ( num & 1 ) == 1 )
34              {
35                  total += a[index];
36              }
37
38              index++;
39
40              num >>= 1; // num = num / 2
41          }
42
43          int total2 = t - total; // 第一個籃子的重量 = 全部 - 第二個籃子
44
45          ans = min(ans,abs(total2-total1));
46      }
```

練習題

Atcoder Beginner Contest 197 pC. ORXOR

給你一個長度為 n $(1 \leq n \leq 20)$ 的序列，你可以在這個序列任意切幾刀，分成數個子陣列，求所有子陣列內的元素由左到右 OR 之後再與其他子陣列 XOR 起來的結果的最大值

全排列枚舉

next_permutation

- 一個可以把所有排列可能都列出來的工具
- 時間複雜度與遞迴實作一樣，好處是不用自己寫遞迴
- 使用前要先導入標頭檔 `algorithm`

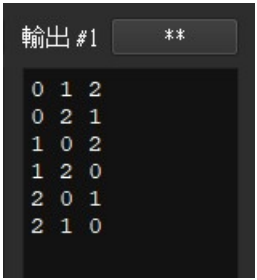
next_permutation

- 這個工具會把你當前傳入的東西照字典序的順序由小到大的做出每一種排列
- 在寫某些題目的時候就會非常好用，像是八皇后問題
- 用法： next_permutation(位置 1(左界 L), 位置 2(右界 R))
- 使用之後就會對 [L,R) 這個左閉右開的區間做全排列
- 回傳：如果當前排列已經是字典序最大的東西了回傳 false，否則回傳 true
- 通常搭配 do-while 迴圈來使用

next_permutation

```
21      vector<int> a;
22
23      for(int i=0;i<3;i++) a.push_back(i);
24
25      do
26      {
27          for(int i=0;i<3;i++) cout << a[i] << " ";
28          cout << "\n";
29      }while(next_permutation(a.begin(),a.end()));
```

next_permutation



練習題

Zerojudge e446 排列生成

給你一個正整數 n $(1 \leq n \leq 8)$，請你照字典序的順序輸出所有 $1 \sim n$ 的排列

Atcoder Beginner Contest 183 pC. Travel

現在有 n $(1 \leq n \leq 8)$ 的城市，給你每一個城市之間通行的所需時間，求從 1 開始，到 $n-1$ 個不同城市後回到 1 的最少所需時間

CSES Problem Set Chessboard and Queens（八皇后問題）

給你一個 8×8 的棋盤，某些格子不能放置皇后，如果某一格放了皇后，那麼那一個皇后的直排、橫排、左斜排、右斜排都不能放置其他皇后，請問共有幾種放置皇后的可能，可以讓你剛好放 8 個皇后（提示： 很明顯每一個橫排都只能放一個皇后，那麼我們可以枚舉每一個橫排的皇后要放在哪一個直排，因為每一個橫排的皇后不能在同一個直排所以等同於你現在要分配這八個皇后分別要在 $1 \sim 8$ 哪一個直排）

遞迴枚舉

遞迴枚舉

- 顧名思義，用遞迴實作枚舉（x
- 優點是寫起來很直接，在我還不會位元枚舉的時候都是用遞迴枚舉實作的

Weekly Training Farm 27 pB. Dreamoon and MRT

Dreamoon 在一條數線上會從某一個起點開始移動 m $(1 \leq m \leq 25)$ 次，每一次移動了 d_i $(1 \leq d_i \leq 10^5)$ 的距離，每一次移動有可能是往左也有可能是往右移動到數線上的某一個點，請問 Dreamoon 所在的數線上最少會有幾個點

Weekly Training Farm 27 pB. Dreamoon and MRT

Dreamoon 在一條數線上會從某一個起點開始移動 m $(1 \leq m \leq 25)$ 次，每一次移動了 d_i $(1 \leq d_i \leq 10^5)$ 的距離，每一次移動有可能是往左也有可能是往右移動到數線上的某一個點，請問 Dreamoon 所在的數線上最少會有幾個點

- 每一次移動不是往左就是往右
- 如果我們使用遞迴來枚舉的話，我們必須額外開一個陣列紀錄現在 Dreamoon 已經在數線上的那些點停了下來

Weekly Training Farm 27 pB. Dreamoon and MRT

Dreamoon 在一條數線上會從某一個起點開始移動 m $(1 \le m \le 25)$ 次，每一次移動了 d_i $(1 \le d_i \le 10^5)$ 的距離，每一次移動有可能是往左也有可能是往右移動到數線上的某一個點，請問 Dreamoon 所在的數線上最少會有幾個點

- 每一次移動不是往左就是往右
- 如果我們使用遞迴來枚舉的話，我們必須額外開一個陣列紀錄現在 Dreamoon 已經在數線上的那些點停了下來
- 如果現在 Dreamoon 在點 p，下一次移動是第 i 次移動，那麼如果往左移動，接下來 Dreamoon 會停留在點 $p - d_i$，反之往右移動則停留在 $p + d_i$

Weekly Training Farm 27 pB. Dreamoon and MRT

Dreamoon 在一條數線上會從某一個起點開始移動 m $(1 \le m \le 25)$ 次，每一次移動了 d_i $(1 \le d_i \le 10^5)$ 的距離，每一次移動有可能是往左也有可能是往右移動到數線上的某一個點，請問 Dreamoon 所在的數線上最少會有幾個點

- 如果現在 Dreamoon 在點 p，下一次移動是第 i 次移動，那麼如果往左移動，接下來 Dreamoon 會停留在點 $p - d_i$，反之往右移動則停留在 $p + d_i$
- 如果我們先遞迴走左邊的 Case，那麼我們會將陣列上 $p + d_i$ 這一個索引值給標記起來，表示數線上必須要有這一個點存在
- 如果現在這一個 Case 走完了，接下來我們要換枚舉走右邊的 Case，但在走右邊之前我們是不是忘了什麼？

Weekly Training Farm 27 pB. Dreamoon and MRT

Dreamoon 在一條數線上會從某一個起點開始移動 m $(1 \le m \le 25)$ 次，每一次移動了 d_i $(1 \le d_i \le 10^5)$ 的距離，每一次移動有可能是往左也有可能是往右移動到數線上的某一個點，請問 Dreamoon 所在的數線上最少會有幾個點

- 如果現在 Dreamoon 在點 p，下一次移動是第 i 次移動，那麼如果往左移動，接下來 Dreamoon 會停留在點 $p - d_i$，反之往右移動則停留在 $p + d_i$
- 如果我們先遞迴走左邊的 Case，那麼我們會將陣列上 $p + d_i$ 這一個索引值給標記起來，表示數線上必須要有這一個點存在
- 如果現在這一個 Case 走完了，接下來我們要換枚舉走右邊的 Case，但在走右邊之前我們是不是忘了什麼？
- 我們要把 $p + d_i$ 這一個點還原成沒有走過的點！因為 $p + d_i$ 這一個點會被標記是因為我們之前往左邊走，走到 $p + d_i$

Weekly Training Farm 27 pB. Dreamoon and MRT

Dreamoon 在一條數線上會從某一個起點開始移動 m $(1 \leq m \leq 25)$ 次，每一次移動了 d_i $(1 \leq d_i \leq 10^5)$ 的距離，每一次移動有可能是往左也有可能是往右移動到數線上的某一個點，請問 Dreamoon 所在的數線上最少會有幾個點

```
18      if( visited[now+d[index]] == 1 ) dfs(now+d[index],index+1,station); // 已經有站
19      else // 沒有站
20      {
21          visited[now+d[index]] = 1; // 標記成有站
22
23          dfs(now+d[index],index+1,station+1);
24
25          visited[now+d[index]] = 0; // 還原成沒有站
26      }
```

Depth First Search(DFS) 深度優先搜尋

DFS 深度優先搜尋

- 有路就走，沒有路就退回來
- 圖論的基本技巧
- 實作可以用 stack 或遞迴
- 大部分都使用遞迴實作

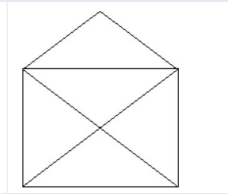
回朔法 Backtracking

回朔法 Backtracking

- 發現新枚舉的狀態不行了，就退回來原本的狀態
- 八皇后問題也是一個經典的回朔法題目

Uva 291 The House Of Santa Claus（經典問題：一筆畫問題）

給你一張圖，請你將字典序由小到大輸出所有能將這張圖一筆畫畫完的路徑，每一條邊只能被走過一次



Uva 291 The House Of Santa Claus（經典問題：一筆畫問題）

給你一張圖，請你將字典序由小到大輸出所有能將這張圖一筆畫畫完的路徑，每一條邊只能被走過一次

- 很明顯我們要遞迴跑過一次所有可能的走法
- 但是在那之前我們要先建邊（這一題的點很少，所以我們可以利用一個二維陣列來記錄哪些點之間有邊，例如：`edge[1][3] = 1` 表示 1 跟 3 之間有一條邊）

Uva 291 The House Of Santa Claus（經典問題：一筆畫問題）

給你一張圖，請你將字典序由小到大輸出所有能將這張圖一筆畫畫完的路徑，每一條邊只能被走過一次

- 很明顯我們要遞迴跑過一次所有可能的走法
- 但是在那之前我們要先建邊（這一題的點很少，所以我們可以利用一個二維陣列來記錄哪些點之間有邊，例如：`edge[1][3] = 1` 表示 1 跟 3 之間有一條邊）
- 開另外一個二維陣列紀錄哪一條邊已經被走過了（例如：`visited[3][2] = 1` 表示 3-2 這一條邊已經被走過了，如果等於 0 表示還沒走過）
- 從 1 開始遞迴枚舉接下來要走哪一條邊，在枚舉的時候要特別注意要從編號小的點開始枚舉（題目要求字典序由小到大）

Uva 291 The House Of Santa Claus（經典問題：一筆畫問題）

給你一張圖，請你將字典序由小到大輸出所有能將這張圖一筆畫畫完的路徑，每一條邊只能被走過一次

- 從 1 開始遞迴枚舉接下來要走哪一條邊，在枚舉的時候要特別注意要從編號小的點開始枚舉（題目要求字典序由小到大）
- 走的過程將走過的邊標記起來，回朔（當前這一個 Case 遞迴完了，要將原本走的那條邊還原成還沒有走過）的時候記得還原
- 在走的過程發現沒有路可以走了，但還沒有畫完，就退回原本的步驟（回朔法）
- 將走的順序紀錄在另外一個陣列上
- 畫完了就用陣列紀錄的資訊輸出你畫的路徑

DFS 深度優先搜尋與回朔法

```
14 void dfs(int point,int index)
15 {
16     // 遞迴終止(輸出答案的部分) 以下程式碼省略
17
18     for(int i=1;i≤5;i++)
19     {
20         if( pre[point][i] == true )
21         {
22             if( visited[point][i] == 1 || visited[i][point] == 1 )
23             {
24                 continue;
25             }
26
27             ans[index] = i;
28
29             visited[point][i] = 1;
30             visited[i][point] = 1; // 1 走過 , 0 沒有走過
31
32             dfs(i,index+1);
33
34             visited[point][i] = 0;
35             visited[i][point] = 0;
36         }
37     }
38 }
```

剪枝 Pruning

剪枝 Pruning

- 暴力美學的一種
- 跟回朔法的概念非常相近
- 跟回朔法不一樣的地方是，回朔法是建立在已經給定的規則，而當我們發現我們違反了那一個規則就退回去原本的步驟
- 剪枝則是我們早就知道某些情況一定完成不了最後的目的，就直接退回去，而這個過程有點像是把遞迴樹的某一個部分直接剪掉，所以叫做剪枝

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

- 這題是一題剪枝剪的出神入化的題目 ...
- 很明顯不做任何剪枝會 TLE
- 但是如果好好剪枝居然會可以在 1 秒內跑完

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

- 一開始我們一樣先寫暴力出來

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

- 一開始我們一樣先寫暴力出來
- 最簡單的剪枝是，如果我們提早走到終點，那我們一定不能達成題目要求
- 另外一個好觀察的是，如果下一步你的方向被指定，但是走了會超出網格或是之前已經走過了，那我們一定也無法達成題目要求

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

接下來我們思考看看，如果我們現在走到最下面了，而且當前左邊跟右邊都還有沒走過的，那很明顯我們一定無法達成題目要求，因為現在的路徑等同於我們把圖分成了左右兩半

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

那麼同個概念，如果我們現在走到最右邊了，表示圖會被我們分成上下兩半，如果現在你的上面跟下面都還有格子沒有被走過，那麼我們一定沒辦法達成題目要求

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

如果現在你只能往左右兩邊走（下面那一格已經走過了，而且上面那一格也已經走過），那是不是其實就跟我們走到最下面的道理一樣，如果現在左右兩邊都還沒走過，那我們一定無法達成題目要求

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

那我們用同樣的道理，如果當前你只能往上或往下走（右邊那一格已經走過，而且左邊那一格也已經走過），那如果現在你的上面跟下面都還沒有走過，那我們一定無法達成題目要求

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

其實這樣的剪枝還是不夠的，但是 ...

如果你用一下第一堂課教的（如果沒有教的話，那可能是第一堂課的講師出了一些狀況，你可以去罵他）硬體優化或編譯器優化之類的黑科技 ... 其實會 AC ...

我有幫大家測試過，會跑 970ms（題目時限: 1000ms）

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

究竟我們還需要剪什麼啊

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

你會發現，如果你現在走到最上面跟最左邊，其實道理跟前面的剪枝一樣，如果你走到最上面了，而且左右都還沒走過就表示圖被我們分成了左右兩半，如果現在走到最左邊，且上下我們都還沒走過，就表示圖被我們分成了上下兩半，所以這兩種情況我們也要剪枝

剪枝 Pruning

CSES Grid Paths

給你一個 7×7 的網格，你要從左上角走到左下角，一開始會輸入一個字串 s，如果第 i 個字元是'?' 表示第 i 步你可以隨意上下左右走，否則如果是 U, D, L, R 第 i 步就只能照指定的走，求有多少走法會將所有格子都走過剛好一次，在字元都是問號的情況下，走法會有 88418 種

接下來你會發現，你居然 1 秒內跑的完 88418 種可能，這就是剪枝強大的地方！

練習題

TIOJ 1025 數獨問題

所謂的數獨就是在一個九乘九的宮格（9 個三乘三宮格）裡，填入 1 到 9 等 9 個數字，讓每個數字在每一行、列及所在的三乘三宮格裡都只出現一次。謎題中會預先填入若干數字，其他宮格位則留白讓玩家自行填入，通常數獨題目都會設計成唯一解。但本題之測試輸入可能會有多解，請你通通將其輸出。你可以假設題目所給的數獨都是合法的，也就是不會有本身已經填好的數字不合規則的情形存在。題目的解數不會超過 50 組。

Breadth First Search 廣度優先搜尋

BFS 廣度優先搜尋

- 核心理念： 從一個起點開始向外擴散
- 常搭配 queue 實作
- 在圖論中的最短路徑問題扮演著不可或缺的角色，許多最短路徑的演算法都是以此為發想
- 這一個單元只會講跟枚舉有相關的 BFS 問題，與圖論相關的問題會在圖論的單元講到

Zerojudge d406. 倒水時間（BFS 實作）

水由上而下的流，現在給你水管之間的地圖水可以由上而下，可以往右流，往左流。可是呢 ... 有時候也可以往上流現在從地圖的最上方開始倒水，請輸出到的時間。(開始倒的地方只有 1 個且只在第一列倒)

BFS 實作

```
1
2  int mp[105][105];
3  int move_x[4] = {1,-1,0,0}; // 維護每一次 x 的變化
4  int move_y[4] = {0,0,1,-1}; // // 維護每一次 y 的變化
5  bool visited[105][105];
6  queue <pair<int,int>> u; // 哦然後，我的 x 跟 y 跟數學是反過來的
7
8  void bfs(int s) // s 是起點
9  {
10     while( u.size() != 0 )
11     {
12         int x = u.front().first;
13         int y = u.front().second;
14
15         u.pop();
16
17         for(int i=0;i<4;i++) // 枚舉要走上下左右哪一個
18         {
19             if(mp[x+move_y[i]][y+move_x[i]] == 1 && visited[x+move_y[i]][y+move_x[i]] == 0)
20             {
21                 u.push(make_pair(x+move_y[i],y+move_x[i]));
22
23                 mp[x+move_y[i]][y+move_x[i]] = mp[x][y] + 1;
24
25                 visited[x+move_y[i]][y+move_x[i]] = 1; // 走過了要標記成有走過了
26             }
27         }
28     }
29 }
```

TIOJ 1008 量杯問題

現在你有 n $(1 \leq n \leq 5)$ 種量杯，每一個量杯一開始都是空的，你可以做以下幾種操作

- 將一個量杯裝滿水
- 將一個量杯內的水倒光
- 將一個量杯的水倒倒另外一個量杯內，只能把另外一個量杯倒滿，或是把倒的那一個量杯的水倒完（水不能滿出來）

請問你最少需要花多少次的操作才能倒出 t $(1 \leq t \leq 50)$ 毫升的水，如果不可能達成輸出 -1

BFS 廣度優先搜尋經典問題

- 我們可以 BFS 所有操作，直到成功做出來
- 但是這一題有另外一個關鍵是，我們要判斷 -1 的情況
- 如果不判斷會一直不斷的做下去，最後 TLE
- 因此我們要用一些比較數學的方式來處理這一個問題（不是，怎麼變數學課了，數學講師出來啦）

- 假設我們現在有三個量杯，這些量杯的容量分別為 a, b, c

- 假設我們現在有三個量杯，這些量杯的容量分別為 a, b, c
- 接下來我們想想看，如果我現在在第二個燒杯裝滿水，其實就等同於現在總水量多了 b 毫升

BFS 廣度優先搜尋經典問題

- 假設我們現在有三個量杯，這些量杯的容量分別為 a, b, c
- 接下來我們想想看，如果我現在在第二個量杯裝滿水，其實就等同於現在總水量多了 b 毫升
- 那現在如果我把第二個量杯的水倒去第一個量杯，總容量不變，依舊是 b 毫升

BFS 廣度優先搜尋經典問題

- 假設我們現在有三個量杯，這些量杯的容量分別為 a, b, c
- 接下來我們想想看，如果我現在在第二個量杯裝滿水，其實就等同於現在總水量多了 b 毫升
- 那現在如果我把第二個量杯的水倒去第一個量杯，且 $a < b$，總容量不變，依舊是 b 毫升
- 那現在我把第一個燒杯的水全部倒掉，總水量就少了 a 毫升
- 式子：$a \times -1 + b \times 1 + c \times 0$

BFS 廣度優先搜尋經典問題

- 假設我們現在有三個量杯，這些量杯的容量分別為 a, b, c
- 接下來我們想想看，如果我現在在第二個量杯裝滿水，其實就等同於現在總水量多了 b 毫升
- 那現在如果我把第二個量杯的水倒去第一個量杯，且 $a < b$，總容量不變，依舊是 b 毫升
- 那現在我把第一個燒杯的水全部倒掉，總水量就少了 a 毫升
- 式子：$a \times -1 + b \times 1 + c \times 0$
- 你會發現由於題目有規定每一次倒水一定不會只倒一半，一定會把某一方倒滿或倒完
- 因此你會發現每一次量杯容量的變化，都會是以下這幾個數字
- $-a, a, -b, b, -c, c$

BFS 廣度優先搜尋經典問題

- 假設我們現在有三個量杯，這些量杯的容量分別為 a, b, c
- 接下來我們想想看，如果我現在在第二個量杯裝滿水，其實就等同於現在總水量多了 b 毫升
- 那現在如果我把第二個量杯的水倒去第一個量杯，且 $a < b$，總容量不變，依舊是 b 毫升
- 那現在我把第一個燒杯的水全部倒掉，總水量就少了 a 毫升
- 式子：$a \times -1 + b \times 1 + c \times 0$
- 你會發現由於題目有規定每一次倒水一定不會只倒一半，一定會把某一方倒滿或倒完
- 因此你會發現每一次量杯容量的變化，都會是以下這幾個數字
- $-a, a, -b, b, -c, c$
- 所以其實就等價於我們要找到一組解 x, y, z 滿足 $ax + by + cz = t$

BFS 廣度優先搜尋經典問題

- 所以其實就等價於我們要找到一組解 x, y, z 滿足 $ax + by + cz = t$
- 如果我們找不到整數解，那我們一定無法透過倒水達成題目要求
- 那我們應該要如何知道是否有整數解呢？

BFS 廣度優先搜尋經典問題

- 所以其實就等價於我們要找到一組解 x, y, z 滿足 $ax + by + cz = t$
- 如果我們找不到整數解，那我們一定無法透過倒水達成題目要求
- 那我們應該要如何知道是否有整數解呢？
- 貝祖定理 !!!

BFS 廣度優先搜尋經典問題

- 所以其實就等價於我們要找到一組解 x, y, z 滿足 $ax + by + cz = t$
- 如果我們找不到整數解，那我們一定無法透過倒水達成題目要求
- 那我們應該要如何知道是否有整數解呢？
- 貝祖定理 !!!
- 沒聽過嗎？沒關係，雖然這應該是數論講師要教的，但我先介紹給大家，針對貝祖定理的證明，我就放心的交給數論講師了 :D
- 如果他沒有教你們怎麼證明，那記得跟我反應

貝祖定理

對於任何的整數 a, b, m，關於未知數 x, y 線性丟番圖方程式 $ax + by = m$，如果有 x, y 有整數解，則 $\gcd(a, b) | m$
gcd 指的是最大公因數

- 因此我們想要知道 $ax + by + cz = t$ 是否有解，我們只要知道 m 是否可以整除 $\gcd(a, b, c)$

- 除此之外，如果我們量杯的容量全部都大於 t，那麼也一定無解
- 綜合以上，我們就可以先把 -1 的情況判斷掉了

BFS 廣度優先搜尋經典問題

- 除此之外，如果我們量杯的容量全部都小於 t，那麼也一定無解
- 綜合以上，我們就可以先把 -1 的情況判斷掉了
- 你說數學不好怎麼辦嗎
- 沒關係，那我教你怎麼唬爛（x

時間剪枝

時間剪枝

- 其實這是一個用來唬爛的技巧
- 有些題目如果出題者沒有好好生測資可能就會被假解過了
- 這一個技巧建議大家比賽的時候可以拿來唬爛看看，但是自己練習的時候建議不要這樣做，這樣你還是沒有學到題目的考點

時間剪枝

- 既然我們知道 1 秒大約可以跑 $10^8 \sim 10^9$，那我們額外開一個變數來記錄現在我們已經跑了幾次迴圈，當已經跑了大約 $10^7 \sim 10^8$ 左右的時候我們就直接結束掉迴圈的進行
- 以量杯問題這一題來說，如果我們不想用數學判斷答案，我們可以利用時間剪枝，當你發現你嘗試了很多次都無法讓你其中一個量杯有剛好 t 毫升的水量，那我們就直接中斷 BFS 輸出 -1
- 我有幫大家試過在 TIOJ 上使用時間剪枝會得到 AC，大家可以去體驗看看時間剪枝的力量