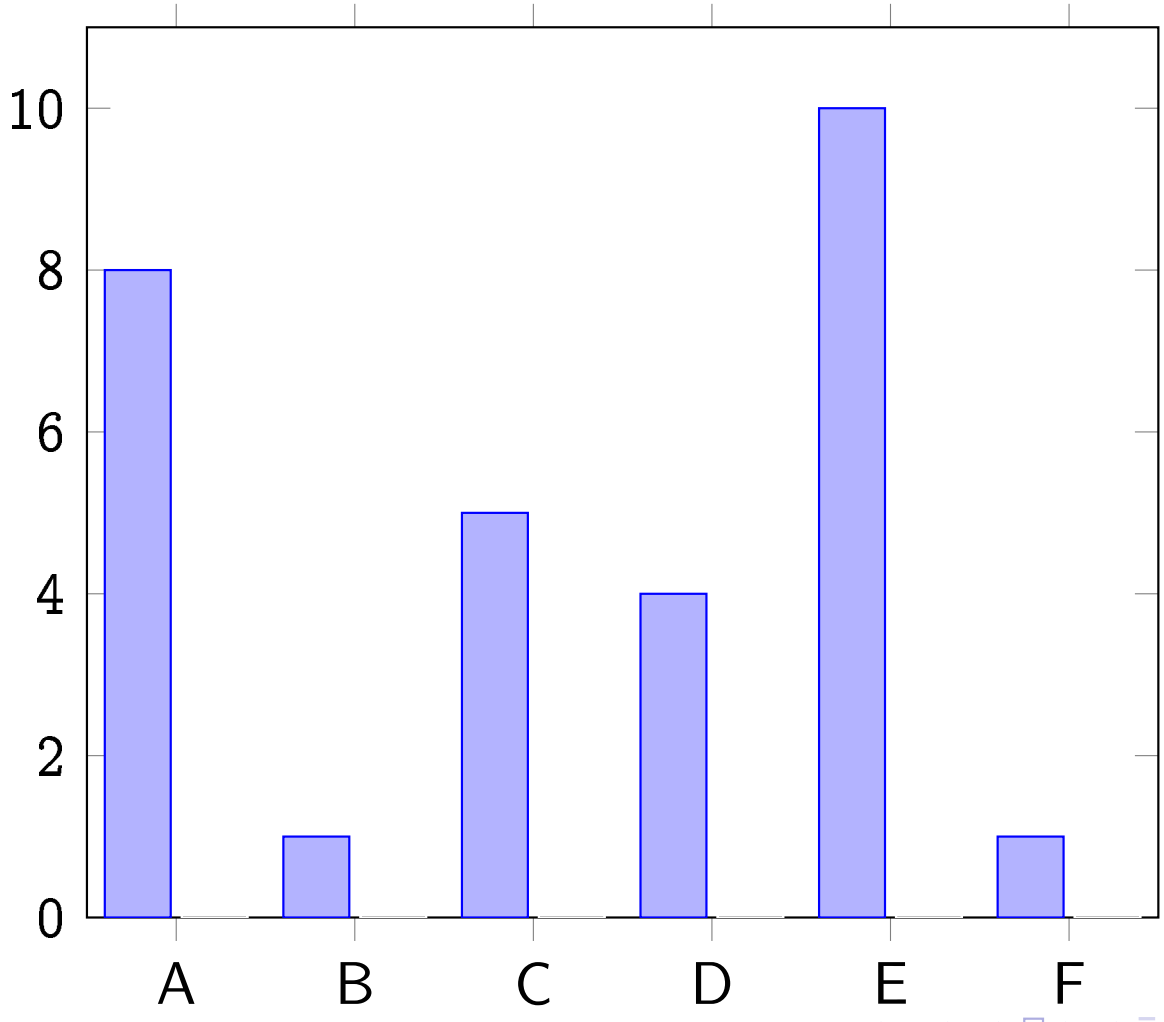
114 臺南一中學科能力競賽校內複選

題解

Sep 25 2025

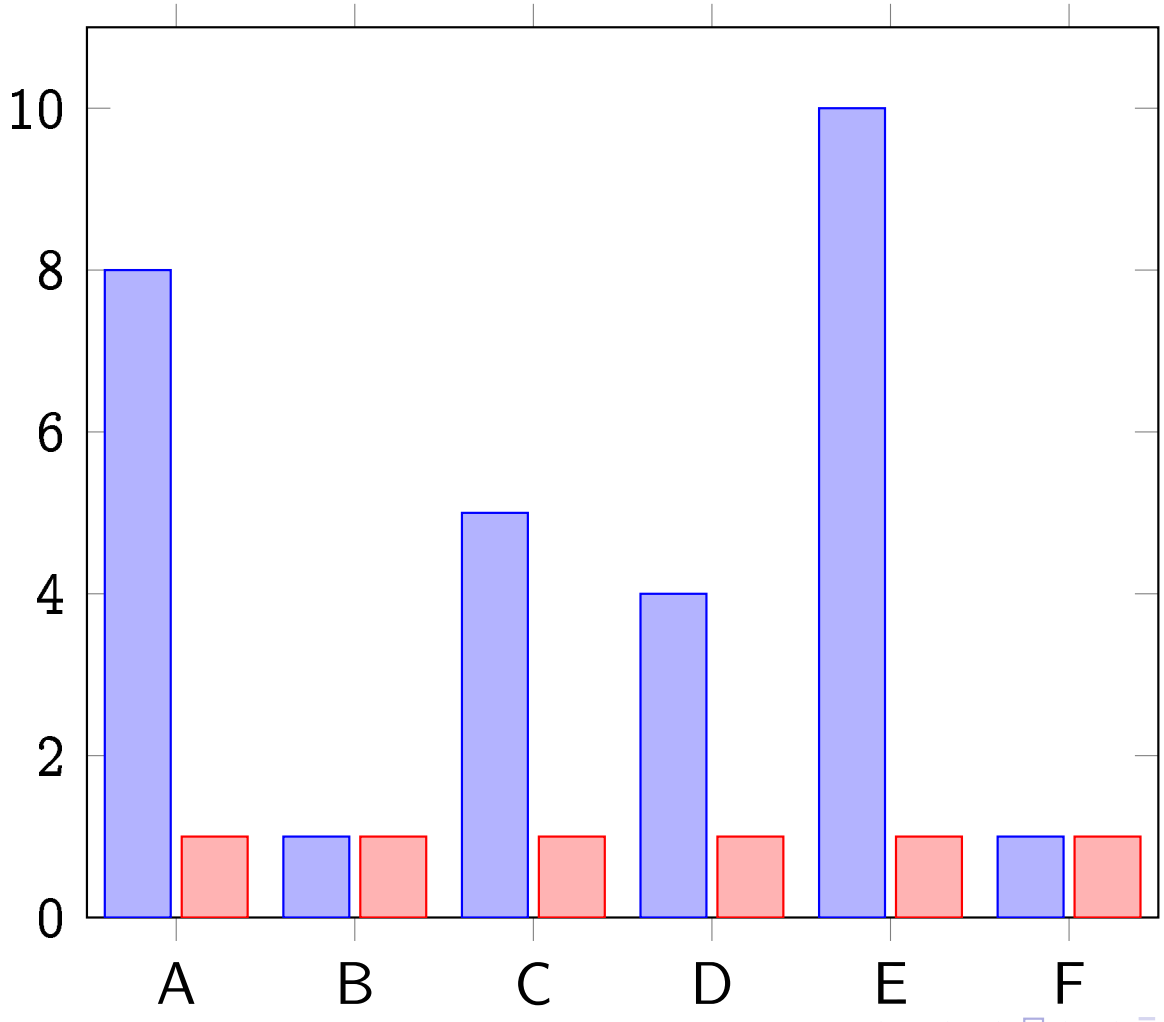
Overview – 預期解出人數

預測校隊線: 400



Overview – 實際解出人數

預測複選線: 400 實際複選線: 0



出題者想說的和 Fun Fact

出題者想說的和 Fun Fact – 前言

出題出很久，希望大家都有好好打

出題者想說的和 Fun Fact – 前言

出題出很久，希望大家都有好好打

我覺得題目出的很好。

出題者想說的和 Fun Fact – 前言

出題出很久，希望大家都有好好打

我覺得題目出的很好。

希望大家都能打得跟初選一樣高分。

出題者想說的和 Fun Fact – Fun Fact — pB

- 本來只有 $K > 1$，初選過發現大家都好強趕緊加難

出題者想說的和 Fun Fact – Fun Fact — pB

- 本來只有 $K > 1$，初選過發現大家都好強趕緊加難
- 出題者自己在這題燒雞超久，本來以為是類似 AI666 那題的 greedy 作法，對拍後才發現假解，又想了兩天才想到正解。

出題者想說的和 Fun Fact – Fun Fact — pB

- 本來只有 $K > 1$，初選過發現大家都好強趕緊加難
- 出題者自己在這題燒雞超久，本來以為是類似 AI666 那題的 greedy 作法，對拍後才發現假解，又想了兩天才想到正解。
- 花了 2 小時暴力跑過對拍，應該沒有又假解。

出題者想說的和 Fun Fact – Fun Fact — pD

- 想不到吧，又出了一題同個模板的。

出題者想說的和 Fun Fact – Fun Fact — pE

- 加難 pB 後發現整體題目好難，趕緊重出一題簽到題

出題者想說的和 Fun Fact – Fun Fact — pE

- 加難 pB 後發現整體題目好難，趕緊重出一題簽到題
- 範圍要讓大家以為是折半枚舉，希望不要有人被騙到

出題者想說的和 Fun Fact – Fun Fact — pE

- 加難 pB 後發現整體題目好難，趕緊重出一題簽到題
- 範圍要讓大家以為是折半枚舉，希望不要有人被騙到
- Hikaru 一次變出五隻城堡的影片
  https://www.youtube.com/watch?v=JW3FOQWzlvA

出題者想說的和 Fun Fact

1. A：構造題
2. B：單調隊列
3. C：貪心
4. D：greedy
5. E：排列組合
6. F：樹論

A. 部落衝突 (tribe)

A. 部落衝突 (tribe) – 題目敘述

給你 N 個點，每個點都有一個顏色 C_i，問你能不能把這 N 個點連成一棵樹且恰好 K 條邊的兩端是不同顏色。

A. 部落衝突 (tribe) – 子任務

1. $K = 0$
2. $K = N - 1$
3. $C_i \in \{1, 2\}$
4. 無額外限制

A. 部落衝突 (tribe) – $K = 0$

很顯然所有點都要是相同顏色才能達成。

A. 部落衝突 (tribe) – $K = 0$

很顯然所有點都要是相同顏色才能達成。

因為要是有不同顏色的兩個點，他們之間一定有至少一條邊連接不同顏色。

A. 部落衝突 (tribe) – $K = N - 1$

很顯然如果所有點都相同顏色就不可能達成。

A. 部落衝突 (tribe) – $K = N - 1$

很顯然如果所有點都相同顏色就不可能達成。

否則我們可以選兩個不同顏色的點當代表，對於每個點都連到顏色不一樣的點上面就好。

A. 部落衝突 (tribe) – $C_i \in \{1, 2\}$

綜合 subtask 1, 2，我們可以推論出如果只有一種顏色，只有 $K = 0$ 才能滿足。

A. 部落衝突 (tribe) – $C_i \in \{1, 2\}$

綜合 subtask 1, 2，我們可以推論出如果只有一種顏色，只有 $K = 0$ 才能滿足。

如果兩種顏色都有，那只要 $K \geq 1$ 都能達成，我們可以用類似 subtask 2 的方式構造。
先選兩個不同顏色的代表點連起來，接著如果目前還沒有 K 條相異顏色的邊，那我們就把點連到相異顏色的代表點，如果已經有 K 條了，那我們就把點連到相同顏色的代表點就好。

A. 部落衝突 (tribe) – 無額外限制

延續 subtask 3，我們把所有顏色都拿一個點起來當代表。

假設有 c 種不同顏色，那 $K \geq c-1$ 才會有解，因為我們一定要用 $c-1$ 條不同顏色的邊將這 c 種顏色連通。

A. 部落衝突 (tribe) – 無額外限制

延續 subtask 3，我們把所有顏色都拿一個點起來當代表。

假設有 c 種不同顏色，那 $K \geq c-1$ 才會有解，因為我們一定要用 $c-1$ 條不同顏色的邊將這 c 種顏色連通。

接著用 subtask 3 的構造方法，如果目前還沒有 K 條相異顏色的邊，那我們就把點連到相異顏色的點，如果已經有 K 條了，那我們就把點連到相同顏色的點就好。

B. 切蛋糕 2(cake2)

B. 切蛋糕 2(cake2) – 題目敘述

我們定義一個合法的切蛋糕方式為一系列的區間 $(l_1, r_1), (l_2, r_2), ..., (l_k, r_k)$ 滿足 $l_1 = 1$, $r_k = n$，並且對於所有 $1 \leq i \leq k$ 都有 $l_i \leq r_i$。且對於所有 $1 \leq i < k$，都有 $r_i + 1 = l_{i+1}$。而這個切蛋糕方式所得到的好感值總和為

$$\sum_{t=1}^{k} \begin{cases} 0 & \text{if } r_t - l_t + 1 < K \\ \min\limits_{l_t \leq i \leq r_t} C_i & \text{otherwise} \end{cases}$$

我們要找出所有合法的切蛋糕方式中好感值總和最大的是多少。

B. 切蛋糕 2(cake2) – 子任務

1. $N \leq 3000$
2. $K > 1$
3. $|C_i| \leq 1$
4. 無額外限制

B. 切蛋糕 2(cake2) – $N \leq 3000$

如果你有寫出初選的 pD，那你應該要會這個子任務

B. 切蛋糕 2(cake2) – $N \leq 3000$

如果你有寫出初選的 pD，那你應該要會這個子任務

定義 $dp[i]$ 為: 只看 1 到 i，分數總和最大會是多少。因此有轉移式

$$dp[i] = \max(dp[i-1], \max_{1 \leq j < i} (dp[j] + cost(i, j)))$$

其中 $cost(i, j)$ 為選區間 i 到 j 的貢獻

而最後 $dp[n]$ 即為答案

B. 切蛋糕 2(cake2) – $K > 1$

可以想到一個貪心的結論，一定有一種最佳解切的區間長度都為 K 或 1

B. 切蛋糕 2(cake2) – $K > 1$

可以想到一個貪心的結論，一定有一種最佳解切的區間長度都為 K 或 1

因為是取 min 的關係，大於 K 的區間都可以把他切成大小為 K 的加上一些大小為 1 的區間。

因此子任務 1 的轉移式就可以用單調對列之類的資料結構 $O(n)$ 轉移了

B. 切蛋糕 2(cake2) – $|C_i| \leq 1$

如果 $K > 1$，就用子任務 2 的方式做就好。
下面都討論 $K = 1$ 的 case。

B. 切蛋糕 2(cake2) – $|C_i| \leq 1$

如果 $K > 1$，就用子任務 2 的方式做就好。
下面都討論 $K = 1$ 的 case。

- 如果區間大小大於 1，那這個區間的兩端一定都是負數。

B. 切蛋糕 2(cake2) – $|C_i| \leq 1$

如果 $K > 1$，就用子任務 2 的方式做就好。
下面都討論 $K = 1$ 的 case。

- 如果區間大小大於 1，那這個區間的兩端一定都是負數。

顯然，如果兩端有正的，那把那些正的自己拿出來成為一個區間貢獻更大。

B. 切蛋糕 2(cake2) – $|C_i| \leq 1$

因此我們考慮以下做法，如果有連續的 -1，那可以把這些放在同一個區間合併成只有一個 -1。而如果有連續 k 個 1，我們可以各自一個區間，由於區間兩端不會是正數，可以直接把他們視為一個 k。而 0 可以選要自己一個區間還是跟 -1 的區間合併，兩者都不影響那個區間的貢獻，因此可以直接把 0 給刪掉。

B. 切蛋糕 2(cake2) – $|C_i| \leq 1$

如此一來，整個序列會變成 $a_1, -1, a_2, -1, a_3, \ldots$ 或 $-1, a_1, -1, a_2, -1, a_3, \ldots$，的形式，其中 $a_i > 0$

B. 切蛋糕 2(cake2) – $|C_i| \leq 1$

如此一來，整個序列會變成 $a_1, -1, a_2, -1, a_3, ...$ 或 $-1, a_1, -1, a_2, -1, a_3, ...$，的形式，其中 $a_i > 0$

而這樣切成的區間就會是答案，因為如果還需要將一些區間合併。假設還要將 $-1, a_l, -1, a_{l+1}, ..., -1, a_r, -1$ 給合併。因為 $a_i > 0$，原本的總和一定是一個大於等於 -1 的數，你合併後他的貢獻會變成 -1，只會變得更差而已，所以不會有這種狀況發生。

B. 切蛋糕 2(cake2) – 無額外限制

我們用 subtask 3 說到的方法，把一群負的合併成一個最小值，一群正的加起來。我們會得到一個 $a_1, b_1, a_2, b_2, \ldots$ 或 $b_1, a_1, b_2, a_2, \ldots$ 的序列，其中 $a_i > 0, b_i < 0$

B. 切蛋糕 2(cake2) – 無額外限制

我們用 subtask 3 說到的方法，把一群負的合併成一個最小值，一群正的加起來。我們會得到一個 $a_1, b_1, a_2, b_2, \ldots$ 或 $b_1, a_1, b_2, a_2, \ldots$ 的序列，其中 $a_i > 0, b_i < 0$

我們假設 $a_1, b_1, \ldots, a_{i-1}, b_{i-1}$ 這樣的切法是最好的

B. 切蛋糕 2(cake2) – 無額外限制

我們用 subtask 3 說到的方法，把一群負的合併成一個最小值，一群正的加起來。我們會得到一個 $a_1, b_1, a_2, b_2, \ldots$ 或 $b_1, a_1, b_2, a_2, \ldots$ 的序列，其中 $a_i > 0, b_i < 0$

我們假設 $a_1, b_1, \ldots, a_{i-1}, b_{i-1}$ 這樣的切法是最好的

顯然，再放一個正的在後面也會是最好的，所以 $a_1, b_1, \ldots, a_{i-1}, b_{i-1}, a_i$ 也會是最好的。

B. 切蛋糕 2(cake2) – 無額外限制

$a_1, b_1, ..., a_{i-1}, b_{i-1}, a_i$ 我們要放一個負的 b_i 在後面

B. 切蛋糕 2(cake2) – 無額外限制

$a_1, b_1, \ldots, a_{i-1}, b_{i-1}, a_i$ 我們要放一個負的 b_i 在後面

- 如果 $b_{i-1} > b_i$

B. 切蛋糕 2(cake2) – 無額外限制

$a_1, b_1, ..., a_{i-1}, b_{i-1}, a_i$ 我們要放一個負的 b_i 在後面

- 如果 $b_{i-1} > b_i$
  - 如果 $b_{i-1} + a_i \leq 0$
    我們可以直接把 b_{i-1}, a_i 跟 b_i 合併在一起並遞迴下去，這樣會讓代價從 $... + b_{i-1} + a_i + b_i$ 變為 $... + b_i$，不虧。

B. 切蛋糕 2(cake2) – 無額外限制

$a_1, b_1, ..., a_{i-1}, b_{i-1}, a_i$ 我們要放一個負的 b_i 在後面

- 如果 $b_{i-1} > b_i$
  - 如果 $b_{i-1} + a_i \leq 0$
    我們可以直接把 b_{i-1}, a_i 跟 b_i 合併在一起並遞迴下去，這樣會讓代價從 $... + b_{i-1} + a_i + b_i$ 變為 $... + b_i$，不虧。
  - $\rightarrow$ 像是 -8, 5, -4, 3 要放 -6 進去，可以變成 -8, 5 放 -6 進去

B. 切蛋糕 2(cake2) – 無額外限制

$a_1, b_1, ..., a_{i-1}, b_{i-1}, a_i$ 我們要放一個負的 b_i 在後面

- 如果 $b_{i-1} > b_i$
  - 如果 $b_{i-1} + a_i \leq 0$
    我們可以直接把 b_{i-1}, a_i 跟 b_i 合併在一起並遞迴下去，這樣會讓代價從 $... + b_{i-1} + a_i + b_i$ 變為 $... + b_i$，不虧。
  - $\rightarrow$ 像是 -8, 5, -4, 3 要放 -6 進去，可以變成 -8, 5 放 -6 進去
  - 否則 $b_{i-1} + a_i > 0$
    此時後面的數如果要合併要嘛和 b_i 合併，要嘛和更前面 $< b_i$ 的那個合併，不可能有區間斷在 b_{i-1} 左邊了
    因此我們可以直接把 a_{i-1} 看成 $a_{i-1} + b_{i-1} + a_i$ 並遞迴下去。

B. 切蛋糕 2(cake2) – 無額外限制

$a_1, b_1, ..., a_{i-1}, b_{i-1}, a_i$ 我們要放一個負的 b_i 在後面

- 如果 $b_{i-1} > b_i$
  - 如果 $b_{i-1} + a_i \leq 0$
    我們可以直接把 b_{i-1}, a_i 跟 b_i 合併在一起並遞迴下去，這樣會讓代價從 $... + b_{i-1} + a_i + b_i$ 變為 $... + b_i$，不虧。
  - → 像是 -8, 5, -4, 3 要放 -6 進去，可以變成 -8, 5 放 -6 進去
  - 否則 $b_{i-1} + a_i > 0$
    此時後面的數如果要合併要嘛和 b_i 合併，要嘛和更前面 $< b_i$ 的那個合併，不可能有區間斷在 b_{i-1} 左邊了
    因此我們可以直接把 a_{i-1} 看成 $a_{i-1} + b_{i-1} + a_i$ 並遞迴下去。
  - → 像是 -8, 5, -4, 9 要放 -6 進去，可以變成 -8, 10 放 -6 進去

B. 切蛋糕 2(cake2) – 無額外限制

- 否則 $b_{i-1} < b_i$
  - 如果 $a_i + b_i \leq 0$
    我們可以直接把 a_i, b_i 跟 b_{i-1} 合併在一起，這樣會讓代價從 $\ldots + b_{i-1} + a_i + b_i$ 變為 $\ldots + b_{i-1}$，不虧。

B. 切蛋糕 2(cake2) – 無額外限制

- 否則 $b_{i-1} < b_i$
  - 如果 $a_i + b_i \leq 0$
    我們可以直接把 a_i, b_i 跟 b_{i-1} 合併在一起，這樣會讓代價從 $\ldots + b_{i-1} + a_i + b_i$ 變為 $\ldots + b_{i-1}$，不虧。
  - $\rightarrow$ 像是 -8, 5 要放 -6 進去，那就變成 -8 然後不放東西

B. 切蛋糕 2(cake2) – 無額外限制

- 否則 $b_{i-1} < b_i$
  - 如果 $a_i + b_i \leq 0$
    我們可以直接把 a_i, b_i 跟 b_{i-1} 合併在一起，這樣會讓代價從 $\ldots + b_{i-1} + a_i + b_i$ 變為 $\ldots + b_{i-1}$，不虧。
  - $\rightarrow$ 像是 -8, 5 要放 -6 進去，那就變成 -8 然後不放東西
  - 否則 $a_i + b_i > 0$
    我們直接把 b_i 丟到後面就好

B. 切蛋糕 2(cake2) – 無額外限制

- 否則 $b_{i-1} < b_i$
  - 如果 $a_i + b_i \leq 0$
    我們可以直接把 a_i, b_i 跟 b_{i-1} 合併在一起，這樣會讓代價從 $\ldots + b_{i-1} + a_i + b_i$ 變為 $\ldots + b_{i-1}$，不虧。
  - → 像是 -8, 5 要放 -6 進去，那就變成 -8 然後不放東西
  - 否則 $a_i + b_i > 0$
    我們直接把 b_i 丟到後面就好
  - → 像是 -8, 5 要放 -3 進去，那就直接放進去變 -8, 5, -3

B. 切蛋糕 2(cake2) – 無額外限制

- 否則 $b_{i-1} < b_i$
  - 如果 $a_i + b_i \leq 0$
    我們可以直接把 a_i, b_i 跟 b_{i-1} 合併在一起，這樣會讓代價從 $\ldots + b_{i-1} + a_i + b_i$ 變為 $\ldots + b_{i-1}$，不虧。
  - → 像是 -8, 5 要放 -6 進去，那就變成 -8 然後不放東西
  - 否則 $a_i + b_i > 0$
    我們直接把 b_i 丟到後面就好
  - → 像是 -8, 5 要放 -3 進去，那就直接放進去變 -8, 5, -3

這樣一直做下去就會是好的，至於更詳細的證明這裡篇幅不夠就留給各位讀者回去當練習了。

C. 手錶 (clock)

C. 手錶 (clock) – 題目敘述

給 N 個 (A_i, B_i) 的 pair 和 M，你每次操作可以從這 N 個 pair 中選擇一個集合和一個整數 k，並將這個集合裡面的 A_i 變為 $(A_i + k) \mod M$，每個 pair 最多只能被選到一次，問你最少要幾次操作才能讓所有 A_i 大於等於 B_i。

C. 手錶 (clock) – 子任務

1. A_i 皆相同
2. $N \leq 2000$
3. B_i 皆相同
4. 無額外限制

△ 每個子任務中，如果並非最少次調整，但滿足輸出格式，可以獲得總分乘以 0.3 的分數。

C. 手錶 (clock) – 30 分

我們可以直接花 N 次調整，每次都把 A_i 調到 B_i 就好

C. 手錶 (clock) – A_i 皆相同

很顯然如果所有 A_i 一開始都大於等於 B_i 就不用調整。

C. 手錶 (clock) – A_i 皆相同

很顯然如果所有 A_i 一開始都大於等於 B_i 就不用調整。

如果要調整，那我們將所有 A_i 一起變為 $M - 1$ 就一定符合條件。

C. 手錶 (clock) – $N \leq 2000$

可能下方 greedy 可以 $O(n^2)$ 做，沒想到什麼特別的解。

C. 手錶 (clock) – B_i 皆相同

可能某些 greedy 的方法會對，沒想到甚麼特別的解。

C. 手錶 (clock) – 無額外限制

不考慮那些 A_i 一開始就大於等於 B_i 的。

列一下式子可以發現，每個可以選擇的 k 的範圍為 $B_i - A_i \leq k < M - A_i$。

C. 手錶 (clock) – 無額外限制

不考慮那些 A_i 一開始就大於等於 B_i 的。

列一下式子可以發現，每個可以選擇的 k 的範圍為 $B_i - A_i \leq k < M - A_i$。

因此我們可以轉換一下題目，給你 n 個線段，每個線段為 $[B_i - A_i, M - A_i)$。你要選盡可能少的點，使得每個線段都被至少一個點覆蓋到。

C. 手錶 (clock) – 無額外限制

這顯然是一個經典問題，答案會和選最多不相交的線段 (CSES Movie Festival) 一樣。

C. 手錶 (clock) – 無額外限制

我們可以對每條線段按右界排序的順序去看，一開始維護一個集合 S。
如果 S 為空，那我們就把目前的線段丟進去 S。

C. 手錶 (clock) – 無額外限制

我們可以對每條線段按右界排序的順序去看，一開始維護一個集合 S。
如果 S 為空，那我們就把目前的線段丟進去 S。
假設 S 裡右界最小的線段叫 A，我們現在看的線段叫做 B。
如果 B 的左界和 A 的右界有相交，那我們就把這個線段丟進 S 裡。

C. 手錶 (clock) – 無額外限制

我們可以對每條線段按右界排序的順序去看，一開始維護一個集合 S。
如果 S 為空，那我們就把目前的線段丟進去 S。
假設 S 裡右界最小的線段叫 A，我們現在看的線段叫做 B。
如果 B 的左界和 A 的右界有相交，那我們就把這個線段丟進 S 裡。
否則，A 和 B 一定不能在同一次選到，而因為 S 裡所有線段都和 A 的右界有相交，因此可以選那個右界來把 S 裡的線段都一起處理掉，因此把 S 清空並把 B 放進去。
大概證明是如果之後有一個線段 X 能和 A 這次一起處理掉，那線段 X 也能和 B 一起處理掉。至於更詳細的證明這裡篇幅不夠就留給各位讀者回去當練習了。

D. 隱藏的排列 2 (permutation2)

D. 隱藏的排列 2 (permutation2) – 題目敘述

互動題，有一個 1 到 n 的隱藏排列 p。每次你可以給兩個數字 l, r，詢問有多少數對 (i, j) 滿足 $l \leq i < j \leq r$ 且 $p_i > p_j$ (以下稱為 l, r 內的逆序數對數量)。

D. 隱藏的排列 2 (permutation2) – 子任務

1. $n = 3$
2. 無額外限制
3. △ 觀察一下連續給分的公式，如果我們詢問所有合法的 l, r，那我們可以拿到 59.59 分。

D. 隱藏的排列 2 (permutation2) – $n = 3$

跟初選那題 $n = 3$ 的做法一樣，因為就 6 種可能的排列，我們觀察一下每個排列的問 $(l, r) = (1, 2), (2, 3), (1, 3)$ 分別會是多少

D. 隱藏的排列 2 (permutation2) – $n = 3$

跟初選那題 $n = 3$ 的做法一樣，因為就 6 種可能的排列，我們觀察一下每個排列的問 $(l, r) = (1, 2), (2, 3), (1, 3)$ 分別會是多少

1 2 3 -> (0, 0, 0)
1 3 2 -> (0, 1, 1)
2 1 3 -> (1, 0, 1)
2 3 1 -> (0, 1, 2)
3 1 2 -> (1, 0, 2)
3 2 1 -> (1, 1, 3)

可以發現如果問 (1, 2) 和 (1, 3)，這些排列對應的值兩兩相異，因此我們可以用這兩個詢問找到對應的那個排列。

D. 隱藏的排列 2 (permutation2) – 59.59 分

如果我們知道了所有區間的逆序數對關係，那我們稍微排容一下就可以知道所有兩個數間的大小關係了。

之後可以使用 std::sort 之類的方法來還原出隱藏的序列。

D. 隱藏的排列 2 (permutation2) – 無額外限制

觀察一下分數，要拿到滿分我們最多只能問 $n-1$ 次，並且根據 subtask 1，我們大概可以通靈出是問所有 $[1,i]$ 區間。

D. 隱藏的排列 2 (permutation2) – 無額外限制

觀察一下分數，要拿到滿分我們最多只能問 $n-1$ 次，並且根據 subtask 1，我們大概可以通靈出是問所有 $[1,i]$ 區間。

我們可以從 p_1 看到 p_n，在看到第 p_i 時，我們只需要維護好 p_1 到 p_i 之間的大小關係就好。也就是在不改變相對關係的情況下我們把 p_1 到 p_i 重新編號成 1 到 i。

D. 隱藏的排列 2 (permutation2) – 無額外限制

觀察一下分數，要拿到滿分我們最多只能問 $n-1$ 次，並且根據 subtask 1，我們大概可以通靈出是問所有 $[1,i]$ 區間。

我們可以從 p_1 看到 p_n，在看到第 p_i 時，我們只需要維護好 p_1 到 p_i 之間的大小關係就好。也就是在不改變相對關係的情況下我們把 p_1 到 p_i 重新編號成 1 到 i。

假設我們知道這東西了，那我們把 $[1,i+1]$ 的逆序數對數量減去 $[1,i]$ 的逆序數對數量。就相當於 $[1,i]$ 中有幾個數大於 p_{i+1}，假設叫做 k，那我們只要把原先維護好的 $i-k+1$ 到 i 全部 $+1$，並將 p_{i+1} 設為 $i-k+1$ 就能維護好了。

E. 數城堡 (rook)

E. 數城堡 (rook) – 題目敘述

給你 N, K, M，問你在 $N \times N$ 的棋盤上放 K 隻城堡，使得任兩個城堡不互相攻擊到有多少種方法 $\mod M$

E. 數城堡 (rook) – 子任務

1. $N \leq 10$
2. $N \leq 20$
3. M 是質數
4. 無額外限制

E. 數城堡 (rook) – $N \leq 10$

暴力枚舉就好

E. 數城堡 (rook) – $N \leq 20$

可以位元 DP 去算。

E. 數城堡 (rook) – M 是質數

我們先選要放在哪 k 排，方法數會是 $\binom{n}{k}$

E. 數城堡 (rook) – M 是質數

我們先選要放在哪 k 排，方法數會是 $\binom{n}{k}$

我們在決定要放在這 k 排的哪些位置，他的方法數就會是 $n \times (n-1) \times ... \times (n-k+1)$。答案就是兩者相乘。

E. 數城堡 (rook) – 無額外限制

就 subtask 3 的做法但 $\binom{n}{k}$ 要用 $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$ 的方法去建。

F. 部落衝突 2 (tribe2)

F. 部落衝突 2 (tribe2) – 題目敘述

給你一棵樹，每個點都有一種顏色，有 Q 筆詢問。每次詢問問你從 A 到 B 的路徑上第一個遇到顏色為 C 的點是哪個。

F. 部落衝突 2 (tribe2) – 子任務

1. $N, Q \leq 2000$
2. $U_i = i, V_i = i + 1$
3. $C_i \in \{1, 2\}$
4. 無額外限制

F. 部落衝突 2 (tribe2) – $N, Q \leq 2000$

暴力

F. 部落衝突 2 (tribe2) – $U_i = i, V_i = i + 1$

我們可以對每個顏色維護好有哪些點是這個顏色。

由於圖是一條練，如果 $A \leq B$ 我們可以用 lowerbound 來找到第一個顏色為 C 且大於等於 A 的點就好。
$A > B$ 同理

F. 部落衝突 2 (tribe2) – $C_i \in \{1, 2\}$

由於詢問沒有修改，因此我們可以離線來做，我們先處理完顏色為 1 的再處理顏色為 2 的。

F. 部落衝突 2 (tribe2) – $C_i \in \{1, 2\}$

由於詢問沒有修改，因此我們可以離線來做，我們先處理完顏色為 1 的再處理顏色為 2 的。

對於目前要處理的顏色，相同顏色的點我們將那個點權設為 1，不同顏色的點權設為 0，如此一來我們可以用一條路徑的總和來看這條路徑上是否有出現相同顏色的點。

F. 部落衝突 2 (tribe2) – $C_i \in \{1, 2\}$

由於詢問沒有修改，因此我們可以離線來做，我們先處理完顏色為 1 的再處理顏色為 2 的。

對於目前要處理的顏色，相同顏色的點我們將那個點權設為 1，不同顏色的點權設為 0，如此一來我們可以用一條路徑的總和來看這條路徑上是否有出現相同顏色的點。

我們將 A 到 B 拆成 A 到 l 和 l 到 B，其中 l 是 A 跟 B 的 LCA，如果 A 到 l 有顏色 C，那我們找第一個出現的就好，如果沒有，那我們再到 B 到 l 上找最後一個出現顏色為 C 的點就好，而這些東西可以用倍增之類的方式來維護。

F. 部落衝突 2 (tribe2) – 無額外限制

延續 subtask 3，我們也是離線並且相同顏色的詢問一起處理，由於顏色很多，不可能每種顏色都建一個倍增表，我們需要一個能夠快速算出一條路徑上有多少東西並修改的資料結構。

F. 部落衝突 2 (tribe2) – 無額外限制

延續 subtask 3，我們也是離線並且相同顏色的詢問一起處理，由於顏色很多，不可能每種顏色都建一個倍增表，我們需要一個能夠快速算出一條路徑上有多少東西並修改的資料結構。

輕重鍊剖分完全符合這個條件。雖然輕重鍊剖分沒有超綱，但輕重鍊剖分又大又難寫，換個經典的替代方法，樹差分。

F. 部落衝突 2 (tribe2) – 無額外限制

延續 subtask 3，我們也是離線並且相同顏色的詢問一起處理，由於顏色很多，不可能每種顏色都建一個倍增表，我們需要一個能夠快速算出一條路徑上有多少東西並修改的資料結構。

輕重鍊剖分完全符合這個條件。雖然輕重鍊剖分沒有超綱，但輕重鍊剖分又大又難寫，換個經典的替代方法，樹差分。

可以紀錄每個點 dfs 進入跟離開的時間 in_i out_i 做樹壓平，這樣有個性質是對於 i 的子樹節點都會包含在 in_i out_i 之間，因此只要那個點要 $+1$，我們可以在 in_i 上 $+1$，out_i 上 - 1，如此一來，某個點 i 到根節點上的路徑總和就可以用 1 到 in_i 的總和求得，因為需要修改，所以可以用二元索引樹 (BIT) 來維護。

如此一來我們就可以用 subtask 3 的方式求出答案了。