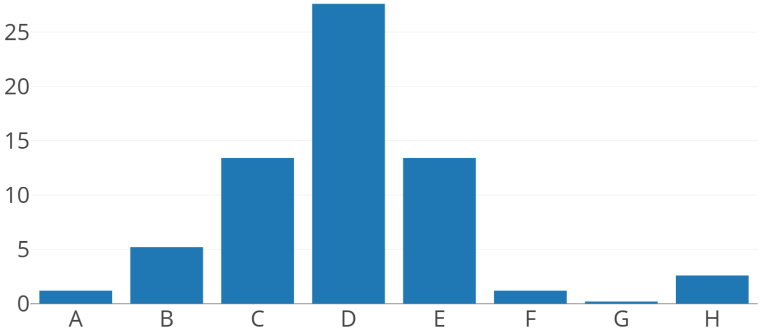
高中組題解

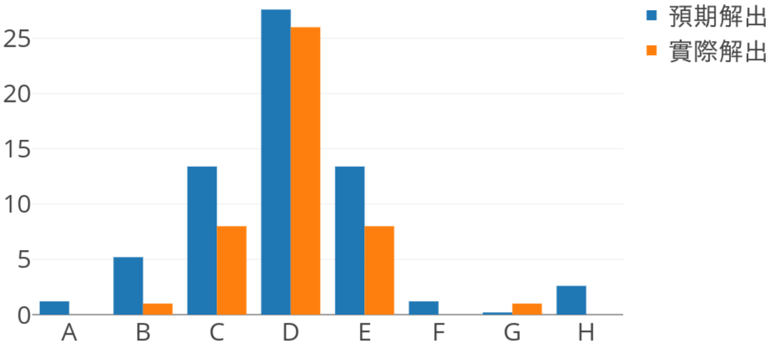
NPSC 2019 裁判組

December 7th, 2019

預期解題人數



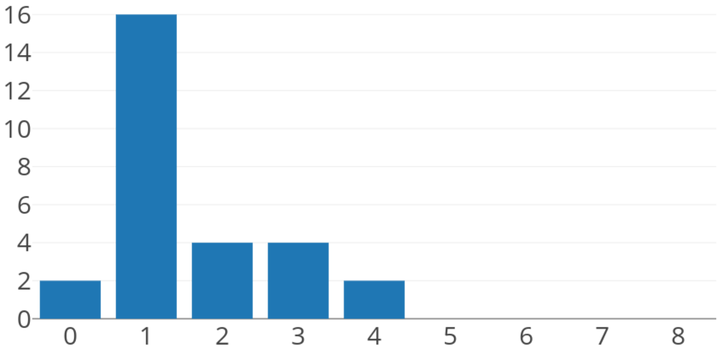
各題解出人數 @Freeze



各題首殺、Dirt 率 @Freeze

- Problem A: Unsolved
- Problem B: 204 mins by fakered
- Problem C: 136 mins by HardwareMaster
- Problem D: 7 mins by fakered
- Problem E: 30 mins by fakered
- Problem F: Unsolved
- Problem G: 131 mins by BiTCoin
- Problem H: Unsolved

題數 v.s. 隊伍數量 @Freeze



Problem D 回文樹

Problem D 回文樹 – 題目敘述

1. 給定輸入字串

Problem D 回文樹 – 題目敘述

1. 給定輸入字串
2. 重新排列成字典序最小的回文

Problem D 回文樹 – Solution

1. 若有超過一種字元出現奇數次，則不可能

Problem D 回文樹 – Solution

1. 若有超過一種字元出現奇數次，則不可能
2. 否則將最小的字元放一半在最前面、第二小的字元再放一半，以此類推，最後在倒序輸出

Problem D 回文樹 – Solution

1. 若有超過一種字元出現奇數次，則不可能
2. 否則將最小的字元放一半在最前面、第二小的字元再放一半，以此類推，最後在倒序輸出
3. 若有出現奇數次的字元的話要在中間輸出

Problem D 回文樹 – Solution

1. 若有超過一種字元出現奇數次，則不可能
2. 否則將最小的字元放一半在最前面、第二小的字元再放一半，以此類推，最後在倒序輸出
3. 若有出現奇數次的字元的話要在中間輸出

$$\text{AAAAABBCC} \Longrightarrow \text{AABCACBAA}$$

Problem C 最小三倍完全数

Problem C 最小三倍完全数 – Solution

1. 一定有解

Problem C 最小三倍完全数 – Solution

1. 一定有解
2. 本地爆搜

Problem C 最小三倍完全数 – Solution

1. 一定有解
2. 本地爆搜
3. 多開幾個同時跑可以在五個小時內跑完

Problem C 最小三倍完全数 – Solution

$$x = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n} \tag{1}$$

$$f(x) = \prod_{i=1}^{n} \sum_{j=0}^{a_i} p_i^j \tag{2}$$

Problem C 最小三倍完全数 – Solution

1. 發現當質數太大時，$f(x)$ 的增加速度會太緩慢，而 x 增加太快

Problem C 最小三倍完全数 – Solution

1. 發現當質數太大時，$f(x)$ 的增加速度會太緩慢，而 x 增加太快
2. 枚舉使用的質數的上限爆搜

Problem C 最小三倍完全数 – Solution

1. 發現當質數太大時，$f(x)$ 的增加速度會太緩慢，而 x 增加太快
2. 枚舉使用的質數的上限爆搜
3. 時限內 AC

Problem C 最小三倍完全数 – Solution

1. 最小的五倍完全數是 14182439040。

Problem C 最小三倍完全数 – Solution

1. 最小的五倍完全數是 14182439040。
2. 在範圍內的五倍完全數有 7 個。

Problem E 密碼鎖

Problem E 密碼鎖 – 題目敘述

1. 兩個長度為 N 的排列 p, q 的距離為

$$\sum_{i=1}^{N} \min(|p_i - q_i|, N - |p_i - q_i|)$$

Problem E 密碼鎖 – 題目敘述

1. 兩個長度為 N 的排列 p, q 的距離為

$$\sum_{i=1}^{N} \min(|p_i - q_i|, N - |p_i - q_i|)$$

2. 給多筆詢問 N, K, p ，問有幾個 q 使得 p, q 距離的個位數為 K

Problem E 密碼鎖 – 題目敘述

1. 兩個長度為 N 的排列 p, q 的距離為

$$\sum_{i=1}^{N} \min(|p_i - q_i|, N - |p_i - q_i|)$$

2. 給多筆詢問 N, K, p ，問有幾個 q 使得 p, q 距離的個位數為 K
3. $1 \leq N \leq 20, 0 \leq K < 9$

Problem E 密碼鎖 – Solution

1. 先考慮只有一筆詢問

Problem E 密碼鎖 – Solution

1. 先考慮只有一筆詢問
2. 令 $dp[i][j][S]$ 代表前 i 個位置，使用了 S 這個集合內的數字，且距離模 10 為 j 的方法數

Problem E 密碼鎖 – Solution

1. 先考慮只有一筆詢問
2. 令 $dp[i][j][S]$ 代表前 i 個位置，使用了 S 這個集合內的數字，且距離模 10 為 j 的方法數
3. 對於位置 i ，枚舉要放在這個位置的是哪個數字：

$$dp[i][j][S] = \sum_{x \in S} dp[i-1][j - \min(|x - p_i|, N - |x - p_i|)][S \setminus x]$$

Problem E 密碼鎖 – Solution

1. 時間複雜度：$O(10 \times N^2 \times 2^N)$

Problem E 密碼鎖 – Solution

1. 時間複雜度：$O(10 \times N^2 \times 2^N)$
2. 不過可以注意到一個 S 只會在一個 i 合法，所以其實是 $O(10 \times N \times 2^N)$

Problem E 密碼鎖 – Solution

1. 多筆詢問：答案只跟 N 與 K 有關！

Problem E 密碼鎖 – Solution

1. 多筆詢問：答案只跟 N 與 K 有關！
2. 對於一個排列 p ，若 q 與 $(1, 2, \ldots, N)$ 的距離為 x ，則 p 跟 $q' = (q_{p_1^{-1}}, q_{p_2^{-1}}, \ldots, q_{p_N^{-1}})$ 的距離也為 x

Problem E 密碼鎖 – Solution

1. 多筆詢問：答案只跟 N 與 K 有關！
2. 對於一個排列 p ，若 q 與 $(1, 2, \ldots, N)$ 的距離為 x ，則 p 跟 $q' = (q_{p_1^{-1}}, q_{p_2^{-1}}, \ldots, q_{p_N^{-1}})$ 的距離也為 x
3. 先用上面的 DP 對於所有 N, K 算好答案，再 $O(1)$ 回答

Problem E 密碼鎖 – Solution

1. 多筆詢問：答案只跟 N 與 K 有關！
2. 對於一個排列 p ，若 q 與 $(1, 2, \ldots, N)$ 的距離為 x ，則 p 跟 $q' = (q_{p_1^{-1}}, q_{p_2^{-1}}, \ldots, q_{p_N^{-1}})$ 的距離也為 x
3. 先用上面的 DP 對於所有 N, K 算好答案，再 $O(1)$ 回答
4. 如果 code 不夠快也可以本機打表

Problem B 忙碌的國度

1. 給定 N 家公司還有 M 間餐廳，每家公司跟餐廳都有一個座標。

Problem B 忙碌的國度 – 題目敘述

1. 給定 N 家公司還有 M 間餐廳，每家公司跟餐廳都有一個座標。
2. 公司有下班時間 t_i，餐廳有打烊時間 c_i。

Problem B 忙碌的國度 – 題目敘述

1. 給定 N 家公司還有 M 間餐廳，每家公司跟餐廳都有一個座標。
2. 公司有下班時間 t_i，餐廳有打烊時間 c_i。
3. 餐廳有美味程度 v_i。

Problem B 忙碌的國度 – 題目敘述

1. 給定 N 家公司還有 M 間餐廳，每家公司跟餐廳都有一個座標。
2. 公司有下班時間 t_i，餐廳有打烊時間 c_i。
3. 餐廳有美味程度 v_i。
4. 開車只能平行座標軸移動，每單位時間可以移動一單位距離。

Problem B 忙碌的國度 – 題目敘述

1. 給定 N 家公司還有 M 間餐廳，每家公司跟餐廳都有一個座標。
2. 公司有下班時間 t_i，餐廳有打烊時間 c_i。
3. 餐廳有美味程度 v_i。
4. 開車只能平行座標軸移動，每單位時間可以移動一單位距離。
5. 對於每一家公司，輸出員工在下班之後能在打烊前抵達的餐廳的最大美味程度。

Problem B 忙碌的國度 – 題目敘述

$$t_i + |x_i - p_j| + |y_i - q_j| \leq c_j$$

Problem B 忙碌的國度 – Solution

當 $x_i \geq p_j$ 且 $y_i \geq q_i$ 時

Problem B 忙碌的國度 – Solution

當 $x_i \geq p_j$ 且 $y_i \geq q_i$ 時，等價於：

$$
\begin{aligned}
t_i + x_i - p_j + y_i - q_j &\leq c_j \\
t_i + x_i + y_i &\leq c_j + p_j + q_j
\end{aligned}
$$

每間公司詢問左下角的所有餐廳中滿足條件的餐廳的最大美味程度。

Problem B 忙碌的國度 – Solution

找一條鉛直線把所有的點分成左右兩半，用一條掃描線從下往上掃，每個右邊的公司詢問左邊已經被看過的餐廳，之後左右兩遍分治下去做一樣的事。

Problem B 忙碌的國度 – Solution

找一條鉛直線把所有的點分成左右兩半，用一條掃描線從下往上掃，每個右邊的公司詢問左邊已經被看過的餐廳，之後左右兩遍分治下去做一樣的事。

查訊的東西可以用一個動態平衡二元搜尋樹（set）維護一個單調序列，也可以直接離散化之後用 BIT / 線段樹做掉。

Problem B 忙碌的國度 – Solution

找一條鉛直線把所有的點分成左右兩半，用一條掃描線從下往上掃，每個右邊的公司詢問左邊已經被看過的餐廳，之後左右兩遍分治下去做一樣的事。

查訊的東西可以用一個動態平衡二元搜尋樹（set）維護一個單調序列，也可以直接離散化之後用 BIT / 線段樹做掉。

同樣的東西對左上、右下跟右上都要做一遍。
總複雜度是 $O(N \log^2 N)$

Problem G 航線建設

Problem G 航線建設 – 題目敘述

1. 給你一張 N 個點的圖，每條邊都有權重

Problem G 航線建設 – 題目敘述

1. 給你一張 N 個點的圖，每條邊都有權重
2. 接下來有 Q 次操作，每次操作可以增加一條邊或刪除一條邊

Problem G 航線建設 – 題目敘述

1. 給你一張 N 個點的圖，每條邊都有權重
2. 接下來有 Q 次操作，每次操作可以增加一條邊或刪除一條邊
3. 並保證操作完以後，當前圖的邊權都是相異的

Problem G 航線建設 – 題目敘述

1. 給你一張 N 個點的圖，每條邊都有權重
2. 接下來有 Q 次操作，每次操作可以增加一條邊或刪除一條邊
3. 並保證操作完以後，當前圖的邊權都是相異的
4. 每次操作完以後請輸出當前這張圖所形成最小生成樹的直徑（樹邊數量最多）有多少條邊

Problem G 航線建設 – 題目敘述

1. 給你一張 N 個點的圖，每條邊都有權重
2. 接下來有 Q 次操作，每次操作可以增加一條邊或刪除一條邊
3. 並保證操作完以後，當前圖的邊權都是相異的
4. 每次操作完以後請輸出當前這張圖所形成最小生成樹的直徑（樹邊數量最多）有多少條邊
5. 如果當前圖不連通，請輸出-1

Problem G 航線建設 – Solution

1. 離線處理

Problem G 航線建設 – Solution

1. 離線處理
2. 時間線段樹

Problem G 航線建設 – Solution

1. 離線處理
2. 時間線段樹
3. 支援回復操作的並查集

Problem G 航線建設 – Solution

1. 離線處理
2. 時間線段樹
3. 支援回復操作的並查集
4. 動態樹（Link cut tree） $\times$ 2

Problem G 航線建設 – Solution

1. 顯然地，若我們要找出最小生成樹的直徑，勢必要先知道當前圖最小生成樹的邊有哪些。

Problem G 航線建設 – Solution

1. 顯然地，若我們要找出最小生成樹的直徑，勢必要先知道當前圖最小生成樹的邊有哪些。
2. 因此，對於原本圖上的每條邊，我們可以將每條邊存活的時間變成線段樹的一段區間，並把邊插入到時間線段樹中。

Problem G 航線建設 – Solution

1. 藉由 DFS 遍歷整顆時間線段樹，以及搭配支援回復操作的並查集，我們可以判斷當前圖的聯通性，檢查答案是不是-1。

Problem G 航線建設 – Solution

1. 藉由 DFS 遍歷整顆時間線段樹，以及搭配支援回復操作的並查集，我們可以判斷當前圖的聯通性，檢查答案是不是-1。
2. 藉由 DFS 遍歷整顆時間線段樹，以及搭配動態樹，我們可以維護出當前的最小生成樹並得知道前最小生成樹上有多少條邊。

Problem G 航線建設 – Solution

1. 藉由 DFS 遍歷整顆時間線段樹，以及搭配支援回復操作的並查集，我們可以判斷當前圖的聯通性，檢查答案是不是-1。
2. 藉由 DFS 遍歷整顆時間線段樹，以及搭配動態樹，我們可以維護出當前的最小生成樹並得知道前最小生成樹上有多少條邊。
3. 這時候我們可以用另外一棵動態樹來維護直徑。

Problem G 航線建設 – Solution

1. 對於合併兩棵樹，我們會要如何維護直徑呢？

Problem G 航線建設 – Solution

1. 對於合併兩棵樹，我們會要如何維護直徑呢？
2. 假設兩棵樹的直徑兩端分別是 (a, b) 及 (c, d)

Problem G 航線建設 – Solution

1. 對於合併兩棵樹，我們會要如何維護直徑呢？
2. 假設兩棵樹的直徑兩端分別是 (a, b) 及 (c, d)
3. 則合併以後的直徑兩端則會是 $(a, b), (a, c), (a, d), (b, c), (b, d), (c, d)$，六種中的一種。

Problem G 航線建設 – Solution

1. 證明？

Problem G 航線建設 – Solution

1. 證明？
2. 留給各位思考囉！

Problem G 航線建設 – Solution

1. 總而言之，因為要使用時間線段樹，所以各個操作都要好好維護要如何恢復操作！

Problem G 航線建設 – Solution

1. 總時間複雜度 $O((N+Q) \times \log(Q) \times \log(N))$

Problem G 航線建設 – Solution

1. 總時間複雜度 $O((N+Q) \times \log(Q) \times \log(N))$
2. AC！

Problem G 航線建設 – Solution

1. Too hard?

Problem G 航線建設 – Solution

1. 考慮這題 N 和 Q 不大？

Problem G 航線建設 – Solution

1. 好好維護邊的大小順序，每次插入按照邊權插入排序 $O(Q)$。

Problem G 航線建設 – Solution

1. 好好維護邊的大小順序，每次插入按照邊權插入排序 $O(Q)$。
2. 每次操作完以後用並查集維護暴力維護最小生成樹 $O(Q \times \alpha(N, N))$。

Problem G 航線建設 – Solution

1. 好好維護邊的大小順序，每次插入按照邊權插入排序 $O(Q)$。
2. 每次操作完以後用並查集維護暴力維護最小生成樹 $O(Q \times \alpha(N, N))$。
3. 每次 $O(N)$ 暴力找直徑。

Problem G 航線建設 – Solution

1. $O(Q^2 \times \alpha(N, N) + QN)$?

Problem G 航線建設 – Solution

1. $O(Q^2 \times \alpha(N, N) + QN)$?
2. AC!!!!!! Meow!!!

Problem G 航線建設 – Solution

1. 暴力出奇蹟？

Problem G 航線建設 – Solution

1. 暴力出奇蹟？
2. 才不是奇蹟！好好計算一陣就會發現能過 XD

Problem G 航線建設 – Solution

1. 暴力出奇蹟？
2. 才不是奇蹟！好好計算一陣就會發現能過 XD
3. N, Q 實際上更大也可以做，但是這題就會變得太難了，事實上標程是「離線處理 $+$ 時間線段樹 $+$ 支援回復操作的並查集 $+$ 動態樹（Link cut tree）$\times$ 2」的方法。

Problem H 鐵路維修問題

Problem H 鐵路維修問題 – 題目敘述

1. 給一棵 N 個點的帶權樹

Problem H 鐵路維修問題 – 題目敘述

1. 給一棵 N 個點的帶權樹
2. 兩個點 x, y 的 cost 定義為 x 到 y 所經過的最大權重乘以 x 到 y 經過的邊數

Problem H 鐵路維修問題 – 題目敘述

1. 給一棵 N 個點的帶權樹
2. 兩個點 x, y 的 cost 定義為 x 到 y 所經過的最大權重乘以 x 到 y 經過的邊數
3. 求 $N \times (N-1)$ 對 $x \neq y$ 的 cost 總和

Problem H 鐵路維修問題 – 題目敘述

1. 給一棵 N 個點的帶權樹
2. 兩個點 x, y 的 cost 定義為 x 到 y 所經過的最大權重乘以 x 到 y 經過的邊數
3. 求 $N \times (N-1)$ 對 $x \neq y$ 的 cost 總和
4. $1 \leq N \leq 10^5$

Problem H 鐵路維修問題 – Solution 1

首先，對於一個樹，可以在 O(點數) 的時間求出，對於任一點 v，其他所有點到 v 的距離總和。

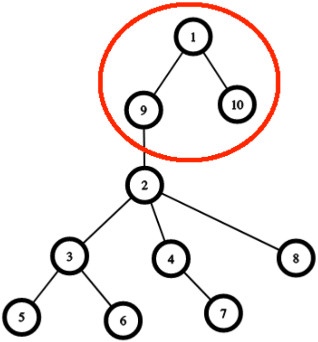
1. 先任意定根做一次 DFS 找出每個點 v 子樹中的所有點到 v 的距離總和（樹 DP）

Problem H 鐵路維修問題 – Solution 1

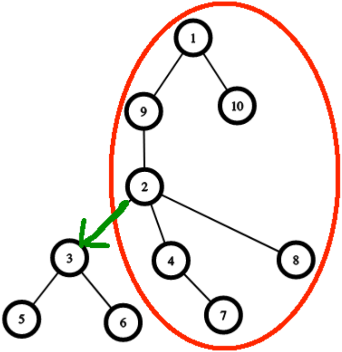
首先，對於一個樹，可以在 O(點數) 的時間求出，對於任一點 v，其他所有點到 v 的距離總和。

1. 先任意定根做一次 DFS 找出每個點 v 子樹中的所有點到 v 的距離總和（樹 DP）
2. 再做第二次 DFS 把祖先的資訊往下帶：紀錄從祖先來的距離總和為多少

Problem H 鐵路維修問題 – Solution 1



(a)



(b)

Figure: 第二次 DFS 到點 2，並往點 3 走

Problem H 鐵路維修問題 – Solution 1

cost 是「最大邊權」乘以路徑長

Problem H 鐵路維修問題 – Solution 1

cost 是「最大邊權」乘以路徑長

$$\Downarrow$$

把邊由小到大加到圖中並計算在這個時候經過這條邊的路徑總長為多少

Problem H 鐵路維修問題 – Solution 1

cost 是「最大邊權」乘以路徑長

$\Downarrow$

把邊由小到大加到圖中並計算在這個時候經過這條邊的路徑總長為多少

$\Downarrow$

動態加邊，每次加完要知道新的連通塊中一個點到其他所有點的距離總和

Problem H 鐵路維修問題 – Solution 1

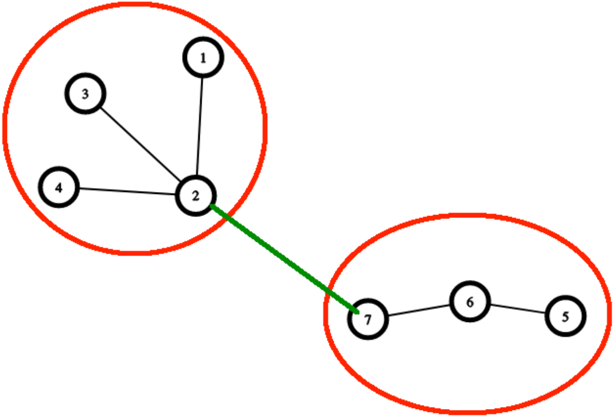
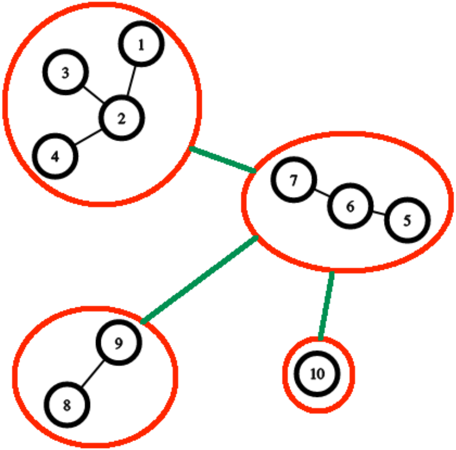


Figure: 原連通塊中所有點到 2 的距離總和為 3 、到 7 的距離總和為 3 ，因此綠色邊對答案的貢獻為 $3 \times 3 + 3 \times 4 + 4 \times 3$

Problem H 鐵路維修問題 – Solution 1

操作分塊：每次處理 K 條邊，處理完後再將它們加到新的圖中，這些邊與已經連通的點集們也會形成一棵樹



Problem H 鐵路維修問題 – Solution 1

1. 每次加一條邊，需要在縮點後的樹上 DFS 求不同連通塊的點到這個點的距離貢獻，花費 $O(K)$ （LCA 可以 $O(N \log N)$ 預處理完之後 $O(1)$ 回答）

Problem H 鐵路維修問題 – Solution 1

1. 每次加一條邊，需要在縮點後的樹上 DFS 求不同連通塊的點到這個點的距離貢獻，花費 $O(K)$ （LCA 可以 $O(N \log N)$ 預處理完之後 $O(1)$ 回答）
2. 每算完 K 條邊之後要把這些邊加入圖中縮點，並計算每個連通塊的距離總和，花費 $O(N)$

Problem H 鐵路維修問題 – Solution 1

1. 每次加一條邊，需要在縮點後的樹上 DFS 求不同連通塊的點到這個點的距離貢獻，花費 $O(K)$ （LCA 可以 $O(N \log N)$ 預處理完之後 $O(1)$ 回答）
2. 每算完 K 條邊之後要把這些邊加入圖中縮點，並計算每個連通塊的距離總和，花費 $O(N)$
3. 取 $K = \sqrt{N}$ ，總複雜度變為 $O(N\sqrt{N})$

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 對於每個節點 v ，維護 v 的子樹裡所有的點到 v 的路徑的(最大邊長, 距離)

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 對於每個節點 v ，維護 v 的子樹裡所有的點到 v 的路徑的 (最大邊長, 距離)
2. 對於小的集合裡的元素 (m_1, d_1) ，對於大的集合裡的元素 (m_2, d_2) ，對答案的貢獻為 $\max(m_1, m_2) \times (d_1 + d_2)$

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 因為有 $\max$ ，所以對大的集合裡的元素由 m 值分 case 處理

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 因為有 $\max$ ，所以對大的集合裡的元素由 m 值分 case 處理
2. 維護 (m, d) 的集合

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 因為有 $\max$ ，所以對大的集合裡的元素由 m 值分 case 處理
2. 維護 (m, d) 的集合
   - 將 m 跟某個數字取 $\max$

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 因為有 $\max$ ，所以對大的集合裡的元素由 m 值分 case 處理
2. 維護 (m, d) 的集合
   - 將 m 跟某個數字取 $\max$
   - 查詢 $m \geq$ 某個數字的人的 $m \times d$ 總和

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 因為有 $\max$ ，所以對大的集合裡的元素由 m 值分 case 處理
2. 維護 (m, d) 的集合
   - 將 m 跟某個數字取 $\max$
   - 查詢 $m \geq$ 某個數字的人的 $m \times d$ 總和
   - 查詢 $m <$ 每個數字的人的 d 總和

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 因為有 $\max$ ，所以對大的集合裡的元素由 m 值分 case 處理
2. 維護 (m, d) 的集合
   - 將 m 跟某個數字取 $\max$
   - 查詢 $m \geq$ 某個數字的人的 $m \times d$ 總和
   - 查詢 $m <$ 每個數字的人的 d 總和
   - 全部 d 加值

Problem H 鐵路維修問題 – Solution 2

啟發式合併，每次算出兩個集合之間的貢獻

1. 因為有 $\max$ ，所以對大的集合裡的元素由 m 值分 case 處理
2. 維護 (m, d) 的集合
   - 將 m 跟某個數字取 $\max$
   - 查詢 $m \geq$ 某個數字的人的 $m \times d$ 總和
   - 查詢 $m <$ 每個數字的人的 d 總和
   - 全部 d 加值
3. Treap，時間複雜度 $O(N \log^2 N)$

Top Tree

Problem F 猜數字

Problem F 猜數字 – 題目敘述

1. 猜 K 個介於 1 到 N 之間的數字

Problem F 猜數字 – 題目敘述

1. 猜 K 個介於 1 到 N 之間的數字
2. 每次詢問所有要猜的數字與某個數的大小關係

Problem F 猜數字 – 題目敘述

1. 猜 K 個介於 1 到 N 之間的數字
2. 每次詢問所有要猜的數字與某個數的大小關係
3. **最少次數**猜到（必須使用最佳策略）

Problem F 猜數字 – Solution

1. 首先，可以發現在猜的過程當中，若詢問 q，則數字會被分為小於 q 和大於等於 q 兩群，在之後的詢問中小於 q 的詢問就與大於等於的那一群無關，反過來亦同，相當於被切成了兩個子問題。

Problem F 猜數字 – Solution

1. 首先，可以發現在猜的過程當中，若詢問 q，則數字會被分為小於 q 和大於等於 q 兩群，在之後的詢問中小於 q 的詢問就與大於等於的那一群無關，反過來亦同，相當於被切成了兩個子問題。
2. 因此，目標是找到一個函數 $f(N, K)$，代表若有 K 個待詢問的數字位在一個大小為 N 的區間中時，最佳策略是詢問這個區間中的哪個位置（$1 \leq f(N, K) < N$）。

Problem F 猜數字 – Solution

1. 首先，可以發現在猜的過程當中，若詢問 q，則數字會被分為小於 q 和大於等於 q 兩群，在之後的詢問中小於 q 的詢問就與大於等於的那一群無關，反過來亦同，相當於被切成了兩個子問題。
2. 因此，目標是找到一個函數 $f(N,K)$，代表若有 K 個待詢問的數字位在一個大小為 N 的區間中時，最佳策略是詢問這個區間中的哪個位置（$1 \leq f(N,K) < N$）。
3. 先說結論（這只是其中一種最佳策略，例如 $g(N,K) = N - f(N,K)$ 顯然是另一種最佳策略）：

$$f(N,K) = \left((N-1) \bmod 2^{\left\lceil \log_2\left(\frac{N-1}{2K}+1\right) \right\rceil} \right) + 1$$

Problem F 猜數字 – Solution

1. 得到這個式子之後可以用數歸（加上繁瑣的計算）得到證明。

Problem F 猜數字 – Solution

1. 得到這個式子之後可以用數歸（加上繁瑣的計算）得到證明。
2. 但是要怎麼知道是這個結果？

Problem F 猜數字 – Solution

1. 得到這個式子之後可以用數歸（加上繁瑣的計算）得到證明。
2. 但是要怎麼知道是這個結果？
3. 找規律！

Problem F 猜數字 – Solution

1. 得到這個式子之後可以用數歸（加上繁瑣的計算）得到證明。
2. 但是要怎麼知道是這個結果？
3. 找規律！
4. 很容易可以得到 $O(N^2K^2)$ 的 DP 解以計算最佳策略要花幾步：

$$dp(N, K) = \min_{0<i\leq N/2} \max_{0\leq j\leq K} dp(i, j) + dp(N-i, K-j)$$

Problem F 猜數字 – Solution

1. 得到這個式子之後可以用數歸（加上繁瑣的計算）得到證明。
2. 但是要怎麼知道是這個結果？
3. 找規律！
4. 很容易可以得到 $O(N^2K^2)$ 的 DP 解以計算最佳策略要花幾步：

$$dp(N,K) = \min_{0<i\leq N/2} \max_{0\leq j\leq K} dp(i,j) + dp(N-i, K-j)$$

5. 加上回溯、輸出結果，就可以開始找規律了！

Problem F 猜數字 – Solution

1. 得到這個式子之後可以用數歸（加上繁瑣的計算）得到證明。
2. 但是要怎麼知道是這個結果？
3. 找規律！
4. 很容易可以得到 $O(N^2K^2)$ 的 DP 解以計算最佳策略要花幾步：

$$dp(N,K) = \min_{0<i\leq N/2} \max_{0\leq j\leq K} dp(i,j) + dp(N-i,K-j)$$

5. 加上回溯、輸出結果，就可以開始找規律了！
6. 雖然式子看起來頗複雜，但是這個規律其實不難找。如果找到之後不放心可以用找出來的式子證一次。

Problem A 逆序輸出

Problem A 逆序輸出 – 題目敘述

1. 有 24 個儲存單元

Problem A 逆序輸出 – 題目敘述

1. 有 24 個儲存單元
2. 每次可以複製、遞增或輸出

Problem A 逆序輸出 – 題目敘述

1. 有 24 個儲存單元
2. 每次可以複製、遞增或輸出
3. 依序輸出 $N, N-1, \cdots, 0$，連續的兩個輸出間只能作 15 次操作

Problem A 逆序輸出 – Solution

1. 首先，不難發現一開始初始化只需要 $N + M$ 次操作（雖然沒有明確限制，但是如果輸出太多操作有可能 TLE），其中 M 是記憶單元數量。

Problem A 逆序輸出 – Solution

1. 首先，不難發現一開始初始化只需要 $N + M$ 次操作（雖然沒有明確限制，但是如果輸出太多操作有可能 TLE），其中 M 是記憶單元數量。
2. **關鍵想法**：每個記憶單元負責輸出 lowbit 在某個位置的所有數字（例如其中一個輸出 $1, 3, 5, 7, \cdots$ 、一個輸出 $2, 6, 10, 14, \cdots$ 、一個輸出 $4, 12, 20, 28, \cdots$ ，以此類推）。

Problem A 逆序輸出 – Solution

1. 首先，不難發現一開始初始化只需要 $N+M$ 次操作（雖然沒有明確限制，但是如果輸出太多操作有可能 TLE），其中 M 是記憶單元數量。
2. **關鍵想法**：每個記憶單元負責輸出 lowbit 在某個位置的所有數字（例如其中一個輸出 $1, 3, 5, 7, \cdots$、一個輸出 $2, 6, 10, 14, \cdots$、一個輸出 $4, 12, 20, 28, \cdots$，以此類推）。
3. 輸出一個數字之後，該記憶單元就要準備輸出下一個它負責的數字（我們稱下一個要輸出的數字為「目標」）。因此，找到當前記憶單元中離目標最近且小於目標的記憶單元，把值複製過去。

Problem A 逆序輸出 – Solution

1. 輸出兩個數字之間的操作就是要在次數限制之內盡量讓記憶單元靠近目標，策略是優先處理 lowbit 位置較低的記憶單元。(當然，如果每個記憶單元都已經在目標上了就直接輸出下個數字。)

Problem A 逆序輸出 – Solution

1. 輸出兩個數字之間的操作就是要在次數限制之內盡量讓記憶單元靠近目標，策略是優先處理 lowbit 位置較低的記憶單元。(當然，如果每個記憶單元都已經在目標上了就直接輸出下個數字。)
2. 可以證明在這個策略之下，若 N 是 2 的冪次，兩次輸出之間其實只需要 1 次的複製操作和 $\lfloor \frac{\log_2 N}{2} \rfloor + 1$ 次遞增操作 (在本題範圍下兩次操作可以最多相隔 13 秒)，並且只需要 $\log_2 N + 1$ 個記憶單元（每個 lowbit 一個，加上一個儲存 0)。

Problem A 逆序輸出 – Solution

1. 輸出兩個數字之間的操作就是要在次數限制之內盡量讓記憶單元靠近目標，策略是優先處理 lowbit 位置較低的記憶單元。(當然，如果每個記憶單元都已經在目標上了就直接輸出下個數字。)
2. 可以證明在這個策略之下，若 N 是 2 的冪次，兩次輸出之間其實只需要 1 次的複製操作和 $\lfloor \frac{\log_2 N}{2} \rfloor + 1$ 次遞增操作 (在本題範圍下兩次操作可以最多相隔 13 秒)，並且只需要 $\log_2 N + 1$ 個記憶單元（每個 lowbit 一個，加上一個儲存 0)。
3. 這個作法只在 N 是 2 的冪次成立。如果 N 不是 2 的冪次，就先從比 N 大的第一個冪次開始作，作到要輸出 N 的時候再以當前的格子內容初始化。