Final Review Report

學號: 109062318　　姓名: 簡弘哲

補題表格

題號	賽中是否有通過	是否要補題
PA	X	O
PB	X	O
PC	X	O
PD	X	O
PE	X	O
PF	O	X

PA

解題報告

1. 使用 DP，定義狀態 dp[i][j]為

   Dp[i][j] = 1 iff 可以從第 1 ~ i 顆蘋果中挑選幾顆蘋果，使得他們的總和等於 j。

   Dp[i][j] = 0 iff 無法從第 1 ~ i 顆蘋果中挑選幾顆蘋果，使得他們的總和等於 j。

   所以 dp[i][0] = 1 (for i = 0 ~ n)，因為可以都不選，使得總和為 0；也有

   dp[0][i] = 0 (for i = 1 ~ sum of all apples)，因為沒辦法在不選蘋果的情況下還能有總和>=1。

   狀態遞移式:

   不選第 i 顆蘋果 => dp[i][j] = dp[i-1][j]

   選第 i 顆蘋果 => dp[i][j] = dp[i-1][j – 第 i 顆蘋果重量] if j >= 第 i 顆蘋果重量

   建完 dp 後可以用一個 for 迴圈枚舉所有可能的重量總和(i = 0 ~ sum)，檢查 dp[n][i]是否可行 (=1)，並求出兩堆蘋果間最小的重量差異。

2. 狀態數 = O(n*Σp)，狀態轉移時間 = O(1)

   O(n*Σp) * O(1) = O(10^6)，可行

3. 沒想到可以用 dp

補題 AC 連結網址:

https://codeforces.com/gym/495010/submission/240646006

PB

解題報告

1. Dijkstra 與 binary search

   利用 Binary search 去枚舉天數(mid)，看 0 到 N-1 的 distance 從哪一天開始會符合題目的條件，

左右界分別為 0 與 10^{18}。每次 binary search 枚舉一個天數時，把這個天數丟進 Dijkstra 裡面進行計算，但不能每次都根據天數與原來的 Graph 去產生一張新的 Graph(會 TLE)，所以就對 relax 做更改，把 cost 換成 max(cost - day, 1)即可。

2. Binary search 左右界分別為 0 與 10^{18}，每次 binary search 都會呼叫 Dijkstra $O(\log 10^{18} * (E\log V)) = O(60 * 2*10^5 * 16) \sim O(2*10^8)$，3.5s 內可行。
3. 知道要用 Dijkstra 跟 binary search，但當時執著想把 A 先寫出來。

補題 AC 連結網址:

https://codeforces.com/gym/495010/submission/240447884

PC

解題報告

1. Segment tree with lazy tag
   根據觀察，區間加值的 v 恆正，因此區間中的最大值與及其個數具有以下的性質:
   (1) 如果區間[1, 5]有 3 個最大值 7、區間[6, 10]有 2 個最大值 5，則區間最大值較小的那個區間([6, 10])會被另一個([1, 5])併吞，意即區間[1, 10]有 3 個最大值 7。
   (2) 如果兩個區間的最大值一樣，則誰也不會併吞誰，且它們的最大值維持不變、新的最大值個數是兩個區間的最大值個數相加。
   Ex. 區間[1, 5]有 3 個最大值 7、區間[6, 10]有 2 個最大值 7 => [1, 10]有 5 個最大值 7
   另一個顯而易見的觀察是當某個區間被加上了 tag，則該區間的實際最大值會是最大值+tag，
   Ex. [1, 2, 3, 4] with tag = 4，max = 4+4 = 8。
   因此只要套上 segment tree 模板與修改邏輯即可解決。
2. 線段樹建立 O(n)，總共 q 次的查詢與更新 O(logn) => O(n + qlogn)，可行。
3. 知道要用 segment tree with lazy tag，但不知道該怎麼存區間最大值及其個數的資訊。

補題 AC 連結網址:

https://codeforces.com/gym/495010/submission/241068953

PD

解題報告

1. 據一整天的觀察發現，當我們將 k 做質因數分解後，得到 $k = p_1^{t1}p_2^{t2}...p_n^{tn}$，其中 p_i 為質因數、t_i 為其指數部分，則答案為 $\Pi[(t_i+1)^3 - t_i^3]$、其中 i=1~n、Π(π 的大寫)代表乘積。
   例如 $k=1500=2^2 3^1 5^3$，則答案為$(3^3 - 2^3)(2^3 - 1^3)(4^3 - 3^3) = 19 * 7 * 37 = 4921$。
   (補充: 正確性說明，以上述例子來說

令 lcm(A, B, C) = 1500 = $2^2 3^1 5^3$ 且 A = $2^{a1}3^{b1}5^{c1}$, B = $2^{a2}3^{b2}5^{c2}$, C = $2^{a3}3^{b3}5^{c3}$，則

Max{a1, a2, a3}必須=質因數 2 的指數部分=2，否則 lcm(A, B, C)不會等於 1500

Max{b1, b2, b3}必須=質因數 3 的指數部分=1，理由同上

Max{c1, c2, c3}必須=質因數 5 的指數部分=3，理由同上

因此{a1, a2, a3}中(0<=a1,a2,a3<=2)只要至少有一個 2 即可滿足條件，滿足條件的排列組合數

= 全部可能的情況(a1,a2,a3 各有三種選擇 0,1,2) – 完全沒有 2 的情況(僅 0,1 兩種)

= $3^3 – 2^3$ = 19

{b1,b2,b3}與{c1,c2,c3}的情況同理可證，最後將所有數字相乘起來即為所求。

)

2. 歐拉線性篩 O(k) = O($2*10^6$)、每個 test case 都做質因數分解 O(tlogk) = O($5*10^5$ * 21) = O(10^7) 可行。
3. 覺得它可能是需要很多時間觀察與思考的數學題，就先跳過了。

補題 AC 連結網址:

https://codeforces.com/gym/495010/submission/241176062

PE

解題報告

1. 使用 DP + DFS + memoization

定義狀態 dp[u][0]、dp[u][1]為

Dp[u][0] = the max score of subtree rooted at u, without selecting any edge between u and u's children.

Dp[u][1] = the max score of subtree rooted at u, with selecting exactly 1 edge between u and u's child.

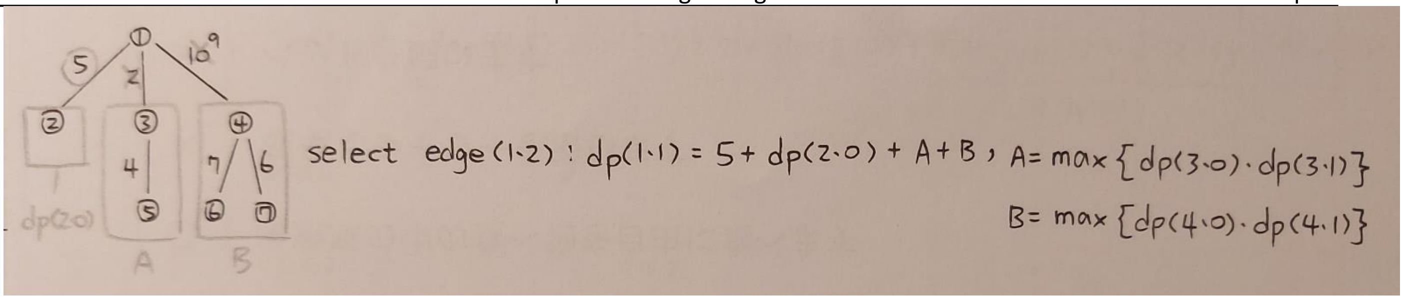
根據題意，如果我們選擇了某條邊而它的端點是 u 和 v (u 的某個 child)的話，則 u 到其他 u's children 的邊會被移除，以及與 v 相鄰的邊也會被移除。所以我們有如下的狀態轉移式:

Dp[u][0] = sum (max { dp[v][0], dp[v][1] })，其中 v = u's child

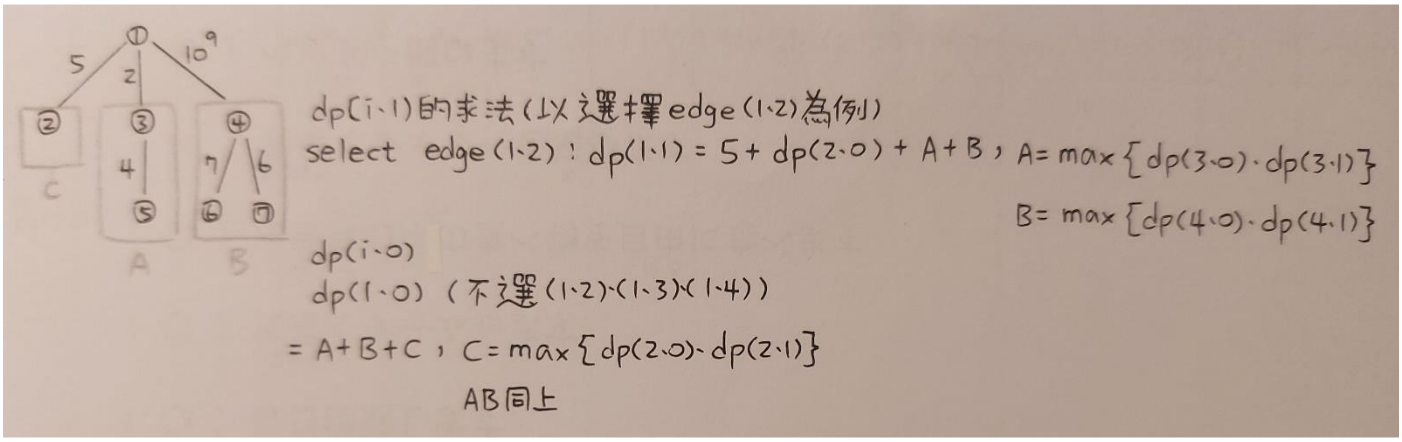
Dp[u][1] = w + dp[x][0] + sum (max { dp[v][0], dp[v][1] })

其中 w 為 edge (u, w)的權重、x 為 u 的其中一個 child、v 為除了 x 之外的 u's child。

以下圖的樹作為例子來說，當我們要計算 dp[1][1] (以 1 為根的子樹，從(1, 2)(1, 3)(1, 4)中選一條邊的情況下，所能得到的最大分數)時，並假設我們選擇 edge(1, 2)，算式如下



當我們要計算 dp[1][0] (以 1 為根的子樹，在不選(1, 2)(1, 3)(1, 4)任何一條邊的情況下，所能得到的最大分數)時，算式如下



另外，從輸入資料中存完樹後，以 node 1 為整棵樹的根，以 1 為起點進行 dfs 走訪去計算每個點 u 的 dp[u][0]與 dp[u][1]。為了避免重複計算，用 memoization 去儲存已經算過的 dp[][]，方便之後存取時可以直接回傳。

2. DP 狀態數為 2n，DFS 會走訪每個點一次，且因為有使用 memoization，所以每個狀態最多只會被計算一次。總時間複雜度為 O(n)，可行。
3. 看完題目覺得太難+沒有想法，就先跳過了

補題 AC 連結網址:

https://codeforces.com/gym/495010/submission/241210094