|  |  |
| --- | --- |
| **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。

1. 狀態數 = O(n\*Σp)，狀態轉移時間 = O(1)

O(n\*Σp) \* O(1) = O(106)，可行

1. 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1018。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。

1. Binary search左右界分別為0與1018，每次binary search都會呼叫Dijkstra

O(log1018 \* (ElogV)) = O(60 \* 2\*105 \* 16) ~ O(2\*108)，3.5s內可行。

1. 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。

1. 線段樹建立O(n)，總共q次的查詢與更新O(logn) => O(n + qlogn)，可行。
2. 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 = p1t1p2t2…pntn，其中pi為質因數、ti為其指數部分，則答案為Π[(ti+1)3 – ti3]、其中i=1~n、Π(π的大寫)代表乘積。

例如k=1500=223153，則答案為(33 - 23)(23 - 13)(43 - 33) = 19 \* 7 \* 37 = 4921。

(補充: 正確性說明，以上述例子來說

令lcm(A, B, C) = 1500 = 223153且A = 2a13b15c1, B = 2a23b25c2, C = 2a33b35c3，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)

= 33 – 23 = 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。

)

1. 歐拉線性篩O(k) = O(2\*106)、每個test case都做質因數分解O(tlogk) = O(5\*105 \* 21) = O(107)

可行。

1. 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

一張含有 文字, 筆跡, 白板 的圖片

自動產生的描述

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

一張含有 文字, 筆跡, 墨水, 信 的圖片

自動產生的描述

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。

1. 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。
2. 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>