建中校內第五次模擬賽題解

hansonyu123, Yihda Yol

植物大戰殭屍2 Subtask 1

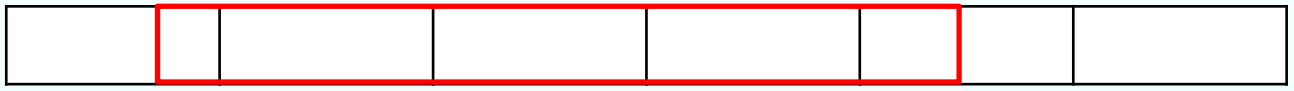
- 直接線性做
  - 複雜度$O(NQ)$，TLE 7分
  - 喵

植物大戰殭屍2 Subtask 2

- 將詢問依左界排序。對於同樣左界的詢問們，依右界排序。
  - 每次都從某個左界開始，將右界由小到大依序處理。對於一個左界，只需$O(N)$（離散化）就可以將所有右界處理完畢。
  - 複雜度$O(N^2 + Q \log Q)$，TLE 26分。

植物大戰殭屍2 Subtask 3

- 將原序列每k個切成一塊
- 對於每個詢問，可以分成「完全包含區塊」以及「不完全包含區塊」兩個部分



- 只需要預處理「區塊詢問」以及在每次詢問時計算「渣渣們對答案的貢獻」即可
  - 渣渣們對答案的貢獻需要由大區間內某渣渣出現次數計算。這可以用vector儲存某值出現的位置後，對該vector二分搜即可得到。

植物大戰殭屍2 Subtask 3

- 預處理區塊間詢問$O(\frac{N}{k}N + N\log N)$
- 對於每個渣渣，要$O(\log N)$的時間計算貢獻
- 對於每個詢問，最多$2k = O(k)$個渣渣
- 總複雜度$O\left(\frac{N^2}{k} + (N + Qk)\log N\right)$
  - k取$\frac{N}{\sqrt{Q\log N}}$時複雜度最小，為$\mathrm{O}\left(N\sqrt{Q\log N}\right)$
  - 這樣取的時候空間複雜度$\mathrm{O}(Q\log N)$
  - 注意這是在線作法喔

植物大戰殭屍2 Subtask 3

- TLE 63分
  - 如果把常數壓超小，然後k取得好的話是有機會AC的—by 吳聖福(?
  - 要分case看渣渣裡的數字數量有沒有低於一個限制，如果有的話用map一次記錄完會比較快

植物大戰殭屍2

- 前面的方法都有「一個一個放進去」的這種feel，不如使用莫隊吧！
- 先離散化後可記錄當前區間某個數出現次數。如此一來區間變化一後的答案很好算，出現次數也很好維護。
- 直接莫隊，複雜度$O\left(N\sqrt{Q}+Q\log Q\right)$。
- AC。

忍者調度問題

- 注意到對於同個主持人，出動愈多忍者愈好。不妨對所有人，看看以他為主持人時滿意度最大為多少，再將這些值取max即可。
  - 問題轉化為，對每個點，詢問以他為根的子樹中在權重和不能超過給定值M的條件下最多可以取幾個點
- 30分的拿法就是對每個點都暴力地將他的後代的權值由小排到大，再二分搜答案。複雜度$O(N^2 \log N)$。

忍者調度問題—整體二分搜

- 這次的解法很多，先從這次的範圍開始講
- 有N筆詢問，每筆都是一個二分搜，那試著整體二分搜吧！
- 二分搜的對象從「選幾個點」改成「選權值不超過某個值的點」
  - 注意因為權重值可能相同，所以要把index放上去以區別兩個權重值一樣的點，或者先將所有人排序後以index為基準

忍者調度問題—整體二分搜

- 對於每次二分搜，可以用樹壓平計算答案以決定要丟給左邊還是右邊
- 對於每個點，最多被二分搜$O(\log N)$次，每次被二分搜到的時候都需要modify一次線段樹，故貢獻的複雜度$O(N \log^2 N)$。

忍者調度問題—整體二分搜

- 對於每個詢問，最多被二分搜$O(\log N)$次，每次被二分搜到的時候需要query一次線段樹，故貢獻複雜度$O(Q \log^2 N) = O(N \log^2 N)$。
- 總複雜度$O(N \log^2 N)$，AC。
- 注意到就算每個點的總預算是分別給定的，這個作法還是行得通。持久化線段樹也可以AC，只是所需空間略大。

忍者調度問題—樹鏈剖分

- 接下來的作法都需要用到「總預算相同」這個性質。
- 接著先講上次範圍的解法吧。
- 從權重最小的節點開始，看看如果他的祖先都選了他的話會不會爆預算
  - 可以證明如果某個點因為這樣爆預算了，那麼他的祖先們也都爆預算了。
  - 此外，加入這個點後被影響到的點只有他的祖先們

忍者調度問題—樹鏈剖分

- 每次加入一個點的時候，先將他的祖先們（含他）的值同加這個點的權重（路徑修改），再看看這條路上第一個爆預算的點在哪（路徑詢問），再把爆預算的點們的值扣掉權重（路徑修改）。
- 問題被轉化為樹鏈剖分可支援的操作。直接套用樹鏈剖分的話複雜度$O(N \log^2 N)$，AC。

忍者調度問題—啟發式合併

- 接下來的解法是從來沒教過的，不過他的概念十分重要。
- 對於其中一個點，假設他的孩子們的答案都已經計算出來，也知道要選哪些節點最好了，那麼將這些節點全部加起來。如果爆預算的話，要從權重最大的那個開始慢慢移除，直到沒有爆預算。
  - 一直移除最大：priority_queue
  - 然而要怎麼樣將一堆priority_queue合併呢？

忍者調度問題—啟發式合併

- 如果有兩個priority_queue需要合併，一個方法是將一個pq的元素一直拔出來放進另一個pq裡。複雜度$O(s_1 \log(s_1 + s_2))$。這引導我們每次合併時，都選小的那個，一直拔元素出來放進大的。
- 這樣做的話，對於每個元素，每次他被移動時所在的pq大小至少會變兩倍，所以至多被移動$O(\log N)$次。因此總複雜度$O(N \log^2 N)$。
  - 此種技巧稱為「啟發式合併」。

忍者調度問題—可合併堆

- 事實上，有些堆可以支援「合併」這個操作，就可以不用啟發式合併而得到更好的複雜度。
  - 以很威的費波那契堆為例，均攤O(1)插入、查詢以及合併，$O(\log N)$刪除。
- 最易實作的可合併堆為左偏樹。插入合併以及刪除皆為$O(\log N)$。
  - 缺點為插入以及刪除皆建立在合併上，故常數比priority_queue還要大許多（畢竟priority_queue的常數很小）

忍者調度問題—左偏樹(Leftist Tree)

- 對於每個節點v，定義S(v)為從v往下走到虛無(?)的最少步數。
- 左偏樹需要滿足兩個性質：
  - 權重需滿足堆的性質
  - 對於任何一個v，S(v->l) >= S(v->r)
- 這樣容易發現一些性質：
  - S(v) = S(v->r)+1
  - 以v為根的子樹至少有$2^{S(v)}-1$個點

忍者調度問題—左偏樹(Leftist Tree)

- 合併兩個左偏樹時，選擇根節點權重較大的那個，並且將他的右子樹和另一棵左偏樹合併。合併完之後記得檢查左偏樹的條件是否滿足。若否，交換左右子樹。
  - 複雜度$O(s(v_1)+s(v_2))$。由前面的性質即知複雜度不會比$O(\log N)$差。
  - 因此「平衡的」左偏樹是一條鏈，「退化的」左偏樹是平衡二元樹。

忍者調度問題—左偏樹(Leftist Tree)

- 插入就是把節點當一棵左偏樹合併。刪除就是把左右子樹合併。
- 將啟發式合併改成左偏樹，即可獲得複雜度$O(N \log N)$的解法。

尋找蘿莉第二彈 – EX

- Subtask 1-4 自己翻之前的題解吧
- Subtask 5 照參考資料用linked list做，複雜度$O(N^2)$

尋找蘿莉第二彈 - EX

- 直接套treap照著參考資料做就可以AC了
  - 每個節點維護當前區間max、「這個是否小於等於下一個」，和當前區間是否有這種數
  - 至於要維護前後樹的大小關係，可以用類似「樹包linked list」的技巧（其實就是對每一個節點都記它的前後元素的指標），並在split和merge的時候好好維護就好了
  - 或者在操作的時候硬搜也可以
- 複雜度$O(N \log N)$

吐鈔機2 Subtask 1-4

- 這題的前四個subtask和最後一個的coding複雜度完全不成比例
- DP，把吐鈔機按照可以買的時間排序
- 令dp[i]是在第D_i天賣出手上的機器後最多可以有多少錢

$$dp[i] = \max_{j<i, dp[j] \geq P_j} \begin{Bmatrix} G_j(D_i - D_j - 1) \\ +R_j - P_j + dp[j] \end{Bmatrix}$$

- $O(N^2)$

吐鈔機2

- 化減(X)一下式子

$$dp[i] = \max_{j<i, dp[j] \geq P_j} \begin{Bmatrix} G_j(D_i - D_j - 1) \\ +R_j - P_j + dp[j] \end{Bmatrix}$$

$$= \max_{j<i, dp[j] \geq P_j} \begin{Bmatrix} G_j D_i + G_j(D_j - 1) \\ +R_j - P_j + dp[j] \end{Bmatrix}$$

- 可以看出這是個斜率優化，但是斜率(G_j)並沒有單調性
- 維護凸包平衡樹

吐鈔機2

- 化減(X)一下式子

$$dp[i] = \max_{j<i, dp[j] \geq P_j} \begin{Bmatrix} G_j(D_i - D_j - 1) \\ +R_j - P_j + dp[j] \end{Bmatrix}$$

$$= \max_{j<i, dp[j] \geq P_j} \begin{Bmatrix} G_j D_i + G_j(D_j - 1) \\ +R_j - P_j + dp[j] \end{Bmatrix}$$

- 可以看出這是個斜率優化，但是斜率(G_j)並沒有單調性
- 維護凸包平衡樹

吐鈔機2

- 千萬不要衝動刻treap(?
- 用set維護線段和交點
- 交點不能用浮點數，要用兩條線表達，不然會WA掉
- 因為截距相乘會爆long long，所以才要用黑魔法
- 複雜度$O(N \log N)$

王老先生 Subtask 1,2

- 直接線性掃過去，複雜度$O(n+Qm)$。
  - TLE 21分

王老先生 Subtask 3

- 每次拍照就相當於區間加權。如果有Q棵線段樹代表每個時刻每個人目前賺的錢，那麼對每個人來說就可以直接二分搜了。
- 每拍一次照是一次修改，故可以考慮使用持久化線段樹。
- 區間加值用懶人標就可以支援了。
- 時間複雜度$O((Q + n\log Q)\log n)$，空間複雜度$O(n + Q\log n)$。WA/TLE 45分。
  - WA原因：區間加值加到重複的人就會爛掉。

王老先生

- 持久化線段樹差一個小問題就過了。不妨試著使用整體二分搜解決問題。
- 對於同一個人管理的不同田地以及一次拍照區間[L,R]，不妨將貢獻全歸給落在[L,R]中最左邊的那個田地。
  - 對於某個田i，假設下一個和他同個主人的田在j，那麼會使i有貢獻的區間[L,R]要滿足L<=i以及R<j。
- 每次二分搜時相當於在解決一個二維查詢修改問題，故將其中一維排序以達成降維。

王老先生

- 對拍照的區間以及田地貢獻區間的右界分別排序即可使用BIT處理前述二維問題。
  - 不是出現第一次了，所以就不仔細講了。
- 因為每次二分搜時是「離線」而不是「在線」，所以資料結構內的東西不能遞給下次二分搜。不過可將目標扣掉現階段賺的錢當作新的目標，遞給下次二分搜。
- 總複雜度$O(n + Q \log n \log Q)$，AC。

最小公倍數 Subtask 1

- 聽說這個時候不會爆long long
- \直接線段樹/
- 複雜度$O(N + Q \log N)$
- 其它subtask會因為時間比較緊而TLE、因overflow而WA。模完後再最小公倍數也會WA（廢話）。

最小公倍數 Subtask 2

- 直接將答案的質因數分解式求出
  - 對每個質因數來說，都是一個RMQ問題
  - 用sparse table有記憶體問題，故使用線段樹
  - Tip:用一棵線段樹存多個值比用多棵線段樹還要有效率（可避免遞迴次數過多）
- 複雜度$O\left(\frac{C}{\log C}(N+Q\log N)\right)$，MLE 23分。

最小公倍數 Subtask 3

- 對於大於$\sqrt{C}$的質因數來說，只有0次或1次兩種可選。
  - 故在意的只有區間中有無該質因數的倍數。區間變動1的時候很容易維護，故考慮使用莫隊。
  - 區間加入/刪除的數最多只會和一個大於$\sqrt{C}$的質因數扯上關係，故莫隊的複雜度夠快。

最小公倍數 Subtask 3

- 對大於$\sqrt{C}$的質因數使用莫隊，小於$\sqrt{C}$的質因數則維持使用線段樹（用莫隊的負擔有點大）。
- 總複雜度$O\left(\frac{\sqrt{C}}{\log C}(N+Q\log N)+N\sqrt{Q}\right)$，MLE 54分。

最小公倍數

- 首先對所有詢問的右界排序： $O(Q \log Q)$
- 假設當前要鉚住的是右界是r的詢問們。對於所有i<r，知道b[i] = query([i,r))/query([i+1, r))就可以用線段樹或BIT之類的處理區間乘積知道query([i,r))了

最小公倍數

- 重點就是 r遞增的時候要怎麼維護b[i]。
- 只要注意到對於一個質數p和一個數i，b[i]的p的冪次變化的話，代表a[r]的p的冪次比a[i+1]到a[r-1]的p的冪次都還大。
- 對於每個質數p，可以用一個單調隊列（嚴格遞減）維護p的冪次數。每次r++時只要用a[r]的質因數冪次數更新top，之後再看看a[r]是不是比top大，是的話就把top拔掉繼續更新，否的話就推入stack。

最小公倍數

- 可以證明這樣變化次數是$O(nf(c))$，其中f是2*3*5*7*...的反函數。因為這裡處理的均攤複雜度O(1)（當然質因數分解$O(n\log n)$要先做好），所以複雜度$O(n\log n)$。

最小公倍數

- 對於每次b[i]的變化，一定都是把b[i]除掉某一個東西。所以這裡的更新可以用線段樹或者是BIT處理。因為更新次數是 $O(nf(c))$ ，所以複雜度$O(nf(c)\log n)$。然後就把右界是r+1的詢問們一次問完。
- 總複雜度$O(q\log q + nf(c)\log n)$，AC。
- $O(f(c)) < O(\log c)$

最小公倍數 大陸人看法

- 其實可以用某種神奇的方法變成跟王老先生的子問題差不多的問題
  - 對於每個數，都拆解成質因數分解式。對於每個質因數p^k，在他原本的位置寫上p,p^2, …, p^k。
  - 把同一個數當作同一個主人，然後某個p^i被考慮的時候都貢獻p。區間最小公倍數就是新區間的所有人的貢獻（同一個主人的數不重複計算），所以可以用類似王老先生整體二分搜的方法做。
- 複雜度估計略，不過和前種方法相去不遠。