

建中校內北市賽模擬賽題解

Hansonyu123, Yihda Yol

沒時間一個一個subtask寫了=3=

兩個比賽擠在一起太忙了w

競技場 Subtask 1,2

- 每次都將最右邊的人淘汰掉是最好的。答案就是 n 個數兩兩相乘之和。
- 暴力算出這個值要 $O(n^2)$ 。你也可以把所有數加起來平方後扣掉平方和再除以二。複雜度 $O(n)$
- 對於後面的subtask，這樣不一定最好。

競技場 Subtask 3,4

- 直接DP。假設 $dp[l][r]$ 是區間 $[l,r)$ 的答案。
 - $dp[l][r] = \max(dp[l][k], dp[k][r]) + ([l,k)$ 和 $[k,r)$ 互毆)
 - $[l,k)$ 和 $[k,r)$ 互毆產生的精采度可以直接暴力枚舉相加。轉移複雜度 $O(n)$ ，狀態數 $O(n^2)$ ，故複雜度 $O(n^3)$ 。

競技場

- 剛剛的轉移複雜度 $O(n)$ 顯得有點笨拙
- $[1,k)$ 和 $[k,r)$ 互毆的精采度其實是 $\text{sum}[1,k) * \text{sum}[k,r)$
 - 所以只要預處理前綴和就好。預處理 $O(n)$ 讓轉移複雜度降為 $O(1)$ 。總複雜度 $O(n^2)$ 。

直線線段相交問題 Subtask 2

- 可以暴力枚舉A,B,C,D
 - 問題轉變為給四個點A,B,C,D，問C和D是否在直線AB的兩側
 - 判斷是否在兩側： $CA \times CB$ 和 $DA \times DB$ 是否異號
 - 等於零的不能算
 - 一次判斷僅需 $O(1)$ ，需 $O(n^4)$ 次判斷。
 - 複雜度 $O(n^4)$

直線線段相交問題

- 將 $O(n^4)$ 降為 $O(n^3)$ ，所以需要枚舉一個東西
- 只枚舉A,B以及C，看看C在直線AB兩側中的哪一側（ $CA \times CB$ 大於或小於零）
- 枚舉完C後，看看直線AB兩側分別有幾個點。乘起來就是AB貢獻的個數了。
- 複雜度 $O(n^3)$

對稱之山 Subtask 1

- 原題：最長回文子字串
- 枚舉每個字串，先判斷他是不是回文，再看他的長度多少。找這些長度中的最大值。
 - 共有 $O(n^2)$ 個字串需枚舉，每次需 $O(n)$
 - 總複雜度 $O(n^3)$

對稱之山 Subtask 2

- 枚舉回文的「中間點」
 - 可能是某個點，可能是某兩個點的中點
 - 可以用左右兩個索引，初始間距為1代表在中點，為2代表在點上
- 從中間開始慢慢向外擴張，直至他不是回文便可找到以此點為中心的最長回文長度。
- 有 $O(n)$ 個中間點需枚舉，每次 $O(n)$
 - 總複雜度 $O(n^2)$

對稱之山 Subtask 2

- 也可以在每兩個點之間插入一個不存在的高度。這樣子回文長度必為奇數。

對稱之山 Subtask 3

- 用剛才的方法轉換成奇回文，再仿造Z-value的計算方法
 - $z[i]$ 代表 $a[i-z[i]]$ 到 $a[i+z[i]]$ 是該區域最長的回文
 $z[i]$ 所提供的資訊最右是到 $a[i+z[i]]$ ，故同原本的Z-algorithm，維護當前 $i+z[i]$ 的最大值
 - 和原本的Z-algorithm一樣討論若干種情況。詳細自理或查詢演算法筆記。
- 每個字元只被往前比對成功一次（ $i+z[i]$ 的最大值遞增），而每次在計算 $z[i]$ 的時候只會比對失敗一次，故總複雜度 $O(n)$

土撥鼠 Subtask 1,2,3

- 求一張圖上一群點有沒有外連邊
- Subtask 1, 2暴力，複雜度 $O(M^{0.5} S)$
- Subtask 3併查集，複雜度 $O((M+S)\alpha(N))$

土撥鼠

- \偉哉hash/
- 整題的關鍵只在賦給每個起終點相同的邊一個隨機值 V ，並將起終點的點權和 $\text{xor } V$ 。
- 由於 xor 有交換律、結合律和 $V \text{ xor } V = 0$ 的特性，對於沒有連出去點集來說，他們的點權 xor 起來一定是 0 。對於有連出去的點集來說，他們的點權 xor 起來就是連出去的邊的邊權 xor 值。

土撥鼠

- 因為剛才提到的特性，拔邊只要再xor一次就好了。不過需要以邊查邊權，還要記錄這組起終點總共重了幾次邊（拔到沒有的時候再刪），需要一個map。
- 複雜度 $O((I + D) \lg M + S)$ ，如果用unordered_map就是 $O(I + D + S)$

土撥鼠

- 隨機的任意個數，它們xor值為零的機率為 $1/C$ （ C 是數字範圍），拼人品看看是不是對的
- AC的機率是 $(1-1/C)^Q$ ，如果 C 是int範圍，大約是99.95%
- 小心rand()的範圍在某些電腦上只有 2^{15} （雖然TIOJ是int範圍）
- AC

突變史萊姆 Subtask 1

- 這題有很多小細節頗有趣或者需要小心注意，所以這題我就講得詳細一點。
- Subtask 1可以直接DFS求解。然而要注意不要走進無窮迴圈，也就是說當前DFS還在stack中的點不能再被取到。
 - 不但保證正確性，也可避免無窮迴圈
- 複雜度估計略。這樣寫可以過Subtask 1。

突變史萊姆 Subtask 2,3

- 建圖：如果 i 可以經過一次操作到 j ，那麼建一條邊從 i 到 j ，邊權是 $|i-j|$ 。
 - 小技巧：邊權可由起終點 $O(1)$ 算出，所以邊權不必存起來
- 想法：詢問相當於多點對最大「逆瓶頸」
←自行發明的名詞(?)
- 多點對：試圖使用Floyd-Warshall的想法。
 $dp[i][j][k]$ 代表從 i 到 j 經過中繼點 $1\sim k$ 的答案中最好的。再對第三維滾動DP。

突變史萊姆 Subtask 2,3

- 轉移式為 $dp[i][j][k] = \max(dp[i][j][k-1], \min(dp[i][k][k-1], dp[k][j][k-1]))$
滾動時直接把第三維拔掉。注意順序。
- 小細節：DP時要小心不能存取到 $dp[i][i]$ ，或者讓 $dp[i][i]$ 預設為無限大(設為 n 即可)。而未被邊連接者預設為0(設為1也可以)。
- 複雜度 $O(n^3)$ 。若滾動的話前三個subtask都會過。不滾動的話會第三個subtask會MLE。
 - 雖然 $O(n^3)$ 在第三個subtask是 10^9 ，然而Floyd-Warshall的常數相當小。

突變史萊姆 Subtask 2,3 另解

- 小觀察：如果有從*i*到*j*的邊，那麼也會有從*j*到*i*的邊，而且邊權一樣。
 - 這個圖其實是個無向圖
- 先將邊權由大排到小。對於每次詢問，從邊權大的開始慢慢加入一個空圖中，看看加到哪一條邊為止時，**a**和**b**連通。該邊的邊權即為所求。
 - 使用並查集。複雜度為 $O(E \lg E + QE \alpha(V)) = O(N \lg^2 N + QN \lg N \alpha(N))$.
 - 小細節：給定邊權後可以用線性的時間枚舉邊權為給定值的邊們。所以從邊權 $n/2$ 開始往下枚舉即可。複雜度改進為 $O(E + QE \alpha(V)) = O(QN \lg N \alpha(N))$

突變史萊姆 離線解法

- 利用剛剛的想法再加上整體二分搜以及持久化並查集的技巧(最後一次上課會上到)，可以利用 $O(V \lg E) = O(N \lg N)$ 的空間在 $O((Q + E \alpha(V)) \lg E) = O((Q + N \lg N \alpha(N)) \lg N)$ 的時間內完成任務。
- 這樣就可以AC了。不過其實用之前我們上過的技巧就可以有同樣的複雜度（而且支援在線查詢）了。

突變史萊姆 Subtask 4

- 每次都要重新並查集很麻煩，不妨一次做完
 - 邊權由大到小依序加入時，記下那些有貢獻的邊(也就是說讓原本不連通的兩坨點連通的邊)
 - 這實際上是在實行Kruskal，所做出來的樹是最大生成樹，也就是最小瓶頸樹的max版本。
- 找到樹後，對於每次詢問，只需看兩點之間的邊權最小是多少即可。
 - 每次都DFS的話複雜度 $O(QV)=O(QN)$ 。加上Kruskal變為 $O(E\alpha(V)+QV)=O(N(Q+\alpha(N)\lg N))$

突變史萊姆 在線解法

- 剛剛在找兩點之間邊權最小值太耗時間了。應該可做一些預處理後在短時間內求解。
- **Tip:**把無根樹變有根樹有時可方便求解。
- 隨意挑一個點當作根，那麼a和b之間的路徑可分為a到(a和b的lca)以及b到(a和b的lca)，故只需用倍增法找到lca並維護路徑權重最小值即可。總複雜度

$$O(E\alpha(V)+Q\lg V)=O((N\alpha(N)+Q)\lg N)$$

突變史萊姆 在線解法

- 小細節：因為這題的圖本身的性質，其最大生成樹的深度不會太深。故從a開始往上找第一個是b的祖先的點的複雜度也是好的。實際執行的表現不比倍增法差。

1035 . 傳說中的邪惡賽後judge(?)

[Submit](#)[Status](#)[Ranklist](#)[Edit](#)[Manage Testdata](#)[Rejudge](#)[Back to Problems List](#)

TopCoder

User's AC Ratio

NaN% (0/0)

Submission's AC Ratio

NaN% (0/0)

Tags

[reserve](#)

Description

Input Format

Output Format

Sample Input

Sample Output

Hints

Problem Source

Subtasks

No.	Time Limit (ms)	Memory Limit (KiB)	Output Limit (KiB)
0	600000	65536	65536

[Submit](#)[Status](#)[Ranklist](#)[Edit](#)[Manage Testdata](#)[Back to Top](#)

#	PID	Submitter	Time	Memory	Verdict	Compiler	Code Length	Score	Submit Time	
47731	1035	這是非常時期	2040571	388	TLE	c++11	17 Bytes	0	2016-10-19 15:59:54	Rejudge
47674	1035	ckcon15_40	1500472	388	TLE	c++11	18 Bytes	0	2016-10-19 15:12:34	Rejudge
47662	1035	這是非常時期	1080341	388	TLE	c++11	17 Bytes	0	2016-10-19 15:00:34	Rejudge
47639	1035	這是非常時期	900267	388	TLE	c++11	17 Bytes	0	2016-10-19 14:32:35	Rejudge
47633	1035	ckcon15_40	720226	388	TLE	c++11	17 Bytes	0	2016-10-19 14:19:32	Rejudge
47625	1035	這是非常時期	600172	392	TLE	c++11	17 Bytes	0	2016-10-19 14:13:12	Rejudge
47616	1035	ckcon15_40	420139	388	TLE	c++11	30 Bytes	0	2016-10-19 14:00:25	Rejudge
47610	1035	這是非常時期	300066	392	TLE	c++11	17 Bytes	0	2016-10-19 13:55:38	Rejudge
47607	1035	ckcon15_40	120031	392	TLE	c++11	30 Bytes	0	2016-10-19 13:53:16	Rejudge
47599	1035	這是非常時期	120026	392	TLE	c++11	17 Bytes	0	2016-10-19 13:45:38	Rejudge

Judge開始爛掉怎麼辦？

- Submit前三思
- 如果有暴力解過小subtask，可以自己生測資
 - fstream/freopen
- 自己手生測資
- 如果覺得分數拿夠了就...(ry

變成賽後judge怎麼辦

- 如果有用暴力解過小subtask，可以自己生測資
- 如果來不及驗證暴力解，只好手生個幾筆後就賭他是對的
- 不要寫出bug