複賽題解

pA. 1999 排隊買飲料 Subtask 1

- 啊不就全部加起來嗎？
- WA 7
- 大家都有拿到這筆，很棒(?

pA. 1999 排隊買飲料

- 直覺可知一個greedy性質：店員只要一完成工作就立刻服務下一個人
- 每次找完成時間最快的店員，更新完成時間
- 暴力找最快$O(NM)$，TLE 45
  - 明明只要找最小值居然有人排序，只好少7分了
- priority_queue $O(N \log N)$，AC

pB. 1998 網路遮罩 Subtask 1

- 每筆詢問把所有的全部遮罩掃一遍
- $O(NM)$，TLE 41

pB. 1998 網路遮罩 解一

- 把每個遮罩換成二進位從高到低位建到Trie裡面，在末端打個標記
- 查詢的時候直接從Trie走下去看能不能走到標記
- $O(32(N+M))$，AC

pB. 1998 網路遮罩 解二

- 既然每個都是連續區間，離散化之後把每個區間開頭存1、結尾存-1
- 可以前綴和之後二分搜，$O((M+N)\log M)$，AC
- 也可以直接離線掃一遍，$O(M\log M + N\log N)$，AC

- 比賽剛開始的時候忘了設分數了(?

pC. 1996 傳送門

- 看來有很多人還不熟悉IOI style的題目，一堆（總共36個）CE(?

#	PID	Submitter	Time	Memory	Verdict	Compiler	Code Length	Score	Submit Time
81924	1996	ckcon17_15	0	0	CE	c++11	118 Bytes	0	2017-09-29 15:29:32
81919	1996	ckcon17_15	0	0	CE	c++11	304 Bytes	0	2017-09-29 15:28:55
81914	1996	ckcon17_05	0	0	CE	c++11	685 Bytes	0	2017-09-29 15:26:26
81875	1996	ckcon17_12	0	0	CE	c++11	491 Bytes	0	2017-09-29 15:13:38
81832	1996	ckcon17_28	0	0	CE	c++11	1.34 KB	0	2017-09-29 15:03:14
81813	1996	ckcon17_05	0	0	CE	c++11	635 Bytes	0	2017-09-29 14:57:19
81811	1996	ckcon17_05	0	0	CE	c++11	636 Bytes	0	2017-09-29 14:56:58
81808	1996	ckcon17_05	0	0	CE	c++11	637 Bytes	0	2017-09-29 14:56:42
81805	1996	ckcon17_27	0	0	CE	c++11	744 Bytes	0	2017-09-29 14:54:59
81804	1996	ckcon17_27	0	0	CE	c++11	744 Bytes	0	2017-09-29 14:54:20

pC. 1996 傳送門 Subtask 1~4

- Subtask 1: 不是貼code就好了嗎，為甚麼有5個人沒拿到啊(?
- Subtask 2: disjoint set判在不在同一個連通塊
- Subtask 3: 隨便亂做
- Subtask 4: Dijkstra’s algorithm 求最短路徑

pC. 1996 傳送門 解一

- 既然有log不會過，看來只能BFS了
- 既然只有三種邊權，那就開3個queue吧
- 遇到邊權0就丟同一個queue、邊權1丟下一個、邊權2丟下兩個，照順序輪流做下去就可以了
- 不過聽說這樣不會過，所以邊權0還是得用disjoint set

pC. 1996 傳送門 解二

- 如果只有邊權0、1的話可以用disjoint set + BFS輕鬆解決
- 邊權2 = 兩條邊權1
- 每條邊權2多建一個點，建成只有邊權0、1的新圖
- 題目保證邊權2的數量不會太多，所以會過

pD. 1997 數列切割 Subtask 1

- 啊不就完全不用切嗎w
- TIOJ上可以拿到分數最短的code
  - main(){}
- 結果還是有6個人沒拿到

pD. 1997 數列切割 Subtask 6~7

- DP
- dp[i][j] = 前j個數字切i份的最大值

$$dp[i][j] = \max_{k<j}\left(dp[i-1][k] + (-1)^{i-1}(sum[j] - sum[k])\right)$$

- 順便記錄轉移來源
- 每一份都要有數字，base case記得處理好
- $O(KN^2)$，TLE 35

pD. 1997 數列切割 解一

- 再觀察一次DP式

$$dp[i][j] = \max_{k<j}\left(dp[i-1][k] + (-1)^{i-1}(sum[j] - sum[k])\right)$$

- 在dp時可以同時記錄$dp[i-1][k] + (-1)^{i} sum[k]$的最大值
- $O(KN)$，AC

- 出題者沒想到這個做法，所以才會有奇怪的$K \leq 6$

pD. 1997 數列切割 解二

- 這個解只能用在$K \leq 6$
- 先考慮$K = 3$，容易發現問題等價於要找一個總和最小的非空子區間，$O(N)$
  - 左右-中間=全部-2*中間
- 計算過程中其實可以知道所有前綴的答案，
- 所以$K = 6$就是前後綴各做一次（一次最小一次最大），之後枚舉中間的切點
- 其他的K也都要分開做
  - Coding複雜度高

pE. 1995 桑京邀請賽

- 為甚麼不能#include <iostream>呢？
- <iostream>大概會占掉10M的記憶體，這樣出題者就沒辦法卡空間了
  - 之這題時限很鬆
- _GLIBCXX_IOSTREAM是gcc中<iostream>的header guard

- 有人忘了沒有iostream還是照樣cout(?

pE. 1995 桑京邀請賽 Subtask 1~2

- Subtask 1: 隨便亂做都會過(?
- Subtask 2: 線段樹或平方分割

pE. 1995 桑京邀請賽 Subtask 3

- 這個subtask只能開大約$N + 4M$個int
- 考慮使用取max的BIT，初始化為-inf
- 將輸入依照左界大到小排序，接著從後往前把數列丟進BIT，同時處理左界為該位置的所有詢問
- $[-inf, -inf, \ldots, -inf, D_l, D_{l+1}, \ldots, D_{N-1}]$的前綴max即為$D_{l \ldots k}$的max
- 這樣是$2N + 4M$個int（原序列、BIT；左右界、query順序、ans），但是因為輸入順序的關係，把上述的過程反過來做（注意BIT還是要從後面開始丟）就可以省下存原序列的空間
- $O((N + M) \log N)$，MLE 87

pE. 1995 桑京邀請賽 Subtask 4

- 這個subtask只能開大約$N + 3M$個int
- 注意到處理完一個詢問之後左右界就沒有用了
- 先把該詢問的值存在原本左界的空間
- 處理完序列之後BIT的空間也沒用了，可以拿來把答案照順序排好
- MLE 100

pE. 1995 桑京邀請賽 EXTRA 解一

- 這個subtask只能開大約$N + 2.25M$個int
- 進入到黑魔法境界(?
- 我們發現左右界和query的編號都不超過2^{24}，所以可以自己做一個3 byte的整數
- 系統會自動memory alignment，所以最好的方法是直接開大小為9的char陣列
- AC

pE. 1995 桑京邀請賽 EXTRA 解二

- 我們繼續把空間壓到$N + 2M$個int
- 換個想法：sparse table
- Sparse table之所以要$O(N \log N)$空間是因為要應付不一樣長度的詢問
- 但是現在是離線，所以我們可以把空間壓回$O(N)$
- 做完一層之後掃一次全部的詢問，如果是該層的長度就填答案
- $O((N + M) \log N)$ ，AC

- 用剛才的黑魔法可以進一步壓到$N + 1.5M$個int