2021 校內賽初賽

題解

pA.

ABC體操

First AC: 王淇

AC: 27 人

出題者:
xiplus

題目概述

對於「已排序」數列，多筆詢問區間中位數

作答概述

小朋友們已經迫不及待地「按照身高從矮到高排成一列」

→ 已排序數列

裏道大哥哥會從隊伍的「某一邊」開始為小朋友編號 1, 2, · · · , N 直到另外一邊

→ 這個數列可能是遞增或遞減

→ 無論是遞增或遞減，中位數都是最中間那個，無需判斷遞增或遞減

→ 區間首尾兩個數字相加除以2就是答案

子題:

- 子題2 - 答案為 (X+Y)/2
- 子題3 - 如果你沒發現數列已排序, 硬爆把區間抓出來排序, O(Q N logN) 讓你過
- 子題5~6 - 可能X>Y, 但其實對中位數來說沒影響
- 子題6 - 序列可能遞減, 但其實對中位數來說沒影響
- 不小心對整個數列sort的話, 分數就 <= 60

AC code

完整連結

```
int N;
cin >> N;
for (int i = 1; i <= N; i++) {
    cin >> A[i];
}
int Q, X, Y;
cin >> Q;
for (int i = 0; i < Q; i++) {
    cin >> X >> Y;
    cout << A[(X + Y) / 2] << endl;
}
```

pB.
國中會考分發

First AC: 王淇

AC: 15 人

出題者:

題目概述

給定每個人的成績以及他們的志願序，求分發結果。

解法:

依照成績由高到低開始填。

對於每個人, 我們看他的第一志願是否已經滿人, 如果滿了, 去看第二志願..... 直到上了或落榜為止。

可以很輕易的發現這是正確的, 不會發生較低成績的人錄取而較高成績的人卻落榜的問題。

一些部份分

56 是給知道解法, 但可能對 vector 等可變長度容器不太熟悉的人。

8 分其實就是每個人都能上自己的第一志願。

12 的話可以直接去看那五間學校, 照著排名輸出。

好好撈分的話就算不會 STL 也可以拿到 76 分！

AC code

完整連結: https://ideone.com/GMTqiC

```
for(int i = 1; i <= n; i++){
    for(int j = 1; j <= 5; j++){
        if(in[pick[rk[i]][j]].size() < k){
            in[pick[rk[i]][j]].push_back(rk[i]);
            break;
        }
    }
}
for(int i = 1; i <= m; i++){
    sort(in[i].begin(), in[i].end());
    cout << in[i].size();
    for(auto &j : in[i]){
        cout << " " << j;
    }
    cout << '\n';
}
```

pC. GAMAGAMA

First AC: 王淇

AC: 5 人

出題者:
xiplus
lys0829
Li-Pro

題目概述

對兩條有$\{+a,\ *a\}$的路徑進行以下操作

- 輸出以 k 開始的最大值
- 將$\{+a,\ *a\}$接在路徑 i 的後方
- 輸出唯一整數解 m滿足以 m開始則兩條路徑結果相同

作答概述

- 有 26 分十分簡單的子題。
- 71 分的子題都不含操作 3, 因此不須相關數學即可獲得許多分數。
- 在範圍內所有測資使用 int 都不會 overflow。

子題 2, 3

- 子題 1: $Q \leq 8000$, 且不含操作 3
  $O(Q^2)$暴力模擬操作 1, 2 即可。

- 子題 2: 操作 2 只有 +
  加法可合併, 因此將兩條路徑加法總和記下即可。
  含操作 3, 但在任何情況下皆為無解或多組解。

子題 4

- 操作 1 的初始積分為 0 或 1, 且不含操作 3
  維護兩條路徑以 0, 1 開始的結果, 如此在任一路徑接上 $\{+a,\ *a\}$時也可推出新的結果。

  如此可 $\mathrm{O}(1)$進行操作 1, 2。

考慮開始為 $\mathbf{x}$。

- 經過 $+a_1$ 後變成 $\mathbf{x} + a_1$。
- 經過 $*a_2$ 後變成 $a_2\mathbf{x} + a_1a_2$。
- 經過 $+a_3$ 後變成 $a_2\mathbf{x} + (a_1a_2 + a_3)$。
- 經過 $*a_4$ 後變成 $a_2a_4\mathbf{x} + [(a_1a_2 + a_3)a_4]$。

永遠是**線性的** $(u\mathbf{x} + v)$！

每條路徑只需要維護 ux + v。

- 接上 +a 時 v’=v+a。
- 接上 *a 時 u’=u*a， v’=v*a。
- 以 k 開始的結果可由 x=k 帶入得出。

可以 $O(1)$處理操作 1, 2。

子題 5

- 不含操作 3
  維護兩條路徑的線性係數 u_1, v_1 和 u_2, v_2。

- 那操作 3 呢?
  進而變成 **求一個 x 使得** $u_1x + v_1 = u_2x + v_2$ 。

求一個 x 使得 $u_1x + v_1 = u_2x + v_2$ 。

移項可得 **x** = $\frac{v_1 - v_2}{u_2 - u_1}$。

如果無法整除 ($u_2 - u_1 = 0$ **或** $(v_1 - v_2)$ **非** $(u_2 - u_1)$ **的倍數**), 則有**無解/無限多組解** 或 **無整數解**, 則任一情況下答案皆為 no。

參考解答

在兩條路徑上維護 ux + v。

- 操作 1 以 k 開始的結果可由 x=k 帶入得出。
- 操作 2 接上 +a 時 v’=v+a。
- 操作 2 接上 *a 時 u’=u*a， v’=v*a。
- 操作 3 視情況回答 $\frac{v_1 - v_2}{u_2 - u_1}$ 或 no。

AC code

完整連結

```
if (qx == 1)
{
     int k;
     cin >> k;

     cout << max(k*u[0] + v[0], k*u[1] + v[1]) << "\n";
}
```

```
else if (qx == 3)
{
     int den = v[0] - v[1], nom = u[1] - u[0];

     if ((nom == 0) || (den % nom != 0))
     {
          cout << "no\n";
     }
     else
     {
          int m = den / nom;
          cout << m << "\n";
     }
}
```

```
else if (qx == 2)
{
     int i, a;
     char op;
     cin >> i >> op >> a, --i;

     if (op == '+')
     {
          v[i] += a;
     }
     else if (op == '*')
     {
          u[i] *= a, v[i] *= a;
     }
}
```

pD.

花子一口吞

First AC: 王淇

AC: 4 人

出題者:
Li-Pro

題目概述

- 求 $\mathrm{mod}\ K$下的最大連續和。

作答概述

- 十分簡單的子題 4 分。
- 不難的子題 44 分。
- 需要處理資料操作, 但可使用 C++ STL 解決。

子題 2, 3, 4

- 子題 1: $N \leq 300$
  $O(N^3)$暴力列舉區間暴力加總。

- 子題 2: $N \leq 8000$
  $O(N^2)$暴力列舉區間, 配合前綴和快速計算區間和。

- 子題 3: $K \leq 30$
  左端點前綴和 $\bmod K$最多 30 個。因此可以維護左端點 $\bmod K$, 並與右端點匹配。整體 $O(NK)$

子題 5

- 符合 $a_i \neq a_{i-1}$ 的 i 最多 2 個, 且 $K \leq 2000$
  將連續的相同數字組成一組, 則可將整個序列分成最多 3 組。每組總和在 $\bmod K$下也只有 2000 種。

  列舉區間即可視為列舉左右端點組別及每組數量 (注意被跨過的組別須整組都取)。可將狀況分為 : 同組、跨組 {(1,2), (2,3), (1,3)}。

  每種狀況可在 $O(K^2)$內進行處理。

考慮在固定右端點 j 下的最大答案。

- 令 $\bmod K$ 下前綴和為 S

$$\max_{i \le j} \{(v_i + \cdots + v_j) \mod K\}$$
$$= \max_{i \le j} \{(S_j - S_{i-1}) \mod K\}$$

- 答案至少有 S_j，愈使答案更大（$S_j + 1$ ~ $K - 1$）我們只需考慮 $-S_{i-1} < K - S_j$。因在範圍內越大越好，可視為**對每個右端點 j 找出最大的** $0 < -S_{i-1} < K - S_j$。

這是一個經典的 BST 上的 predecessor 問題。可以選擇使用...

- 一顆 treap 或其他 BST
- 一顆值域的線段樹
- **std::set 的 lower_bound / upper_bound**

參考解答

- 向右列舉區間右端點 j。
- 利用 std::set::lower_bound 找出(若存在)最大的 $0 < -S_{i-1} < K - S_j$ 以 S_j 或 $S_j - S_i$ 更新答案。
- 每次將 $0 \leq -S_j < K$ 存入 std::set 維護。

AC code

完整連結

```
int ans = 0;
set<int> bst;
for (int j=0; j<n; j++)
{
     auto it = bst.lower_bound(k - S[j]);

     if (it == bst.begin() || *prev(it) >= (k-S[j]))
     {
          // "0 < -S[i-1] < K-S[j]" does not exist
          ans = max(ans, S[j]);
     }
     else
     {
          ans = max(ans, S[j] + *prev(it));
     }

     bst.insert(k - S[j]);
}
```

pE.
蓋歐格

First AC: N/A

AC: 0 人

出題者:
xiplus
lys0829
Li-Pro

小知識

電阻的單位是以物理學家的姓氏「歐姆」命名的，而這位物理學家的名字就是「蓋歐格」。

題目標題跟題目本身是有一點點關係的唷～你發現了嗎？

// 雖然「歐姆蛋」在他出生前已經存在，人們卻不相信此物理學家是以「歐姆蛋」命名。

題目概述

給定一個沒有「電橋」的電路，且電流方向已經指定。
輸出兩點間的「等效電阻」。

作答概述

情形2為串聯，情形3為並聯。

從一張圖找到符合題目中所述情形2、3且 X 及 Y 都是傳送通道的「最簡電路」，將兩條邊合併為一條邊，同時更新該條邊電阻值，重複這個步驟直到整張圖只有一條邊。

AC code

完整連結

pF.
最小生成數

First AC: N/A

AC: 0 人

出題者:
NCTU_Daisengen
靈感: 端木大竣偉
題目: 難不成俊宏
正解: 黑旗劉永福
裝飾: 過來周暘典

題目概述

構造有最小字典序的新序列，使得每 k個元素構成的連續子序列，其大小關係皆和原序列一致。

作答概述

- 含有小筆暴力解測資 讓實作好的人可以進複賽
- 好好分析後可解決 $k = 2$ 的狀況
- 按一定 Greedy 策略可拿到 41 分
- 若能處理相同元素則可再拿到 59 分

k = 1 (子題 3)

啊送你們的一分你們要拿餒

k = n (子題 4、5)

長度為 n 的連續子序列只有一個(也就是整個序列)
若要在不破壞大小關係的情況下構出字典序最小的序列
直接將該序列按照其值排序
從小到大依序從 1 開始往上填即可 (離散化)
此時已經拿到 6 分了

相同的元素記得要填一樣的數字
若處理好的話可以拿到 10 分

k = 2 (子題 6、7)

仔細觀察會發現

此時新序列中的值 b_i 只會受 b_{i-1} , b_{i+1} 影響

我們可以依序決定 b_i 的最小(至少)可能值

$a_{i-1} < a_i$ 則考慮 $b_i = b_{i-1} + 1$

$a_{i-1} = a_i$ 則使 $b_i = b_{i-1}$

$a_{i-1} > a_i$, $b_i = 1$

k = 2 (子題 6、7)

若把上頁說的事情反著再做一遍
藉此確認 b_i 跟 b_{i+1} 的關係
最後 b_i 會有兩個可能的值
挑選最大的作為答案即可

沒有處理相同元素得 12 分
處理完共可得 26 分

考慮全部元素相異的情況

那我們直接用 a_i 從小到大依序去決定 b_i

假設目前列舉的 a_i 位於位置 i

則它的值 b_i 會受位置 $[i-(k-1),\ i+(k-1)]$內已經被決定好的值 b_j 影響 (那些比 a_i 小的 a_j 會在此之前被決定好 b_j)

找出上述區間中的最大值, 將其 +1 就是 b_i的值

那如果有相同元素呢?
要是二個相同元素距離在 k 之內
那該怎麼決定找極值的順序?
順序亂做對這二個元素該怎麼決定其生成數?
三個、四個?

管它幾個, 對於這些會互相影響的區間我們一同修改不就好了

n <= 1000 (子題 2)

每次區間查詢極值直接暴力

複雜度為 $O(nk)$

可拿 47 分

保證全部元素相異 (子題 8)

寫個線段樹處理動態區間極值

阿如果你沒處理相同元素 實作不好的話

~~就不會進南區賽~~

欸都... 可拿 38 分

筆者有試著線段樹 + DSU 處理相同元素亂寫

拿到 41 分

正解

有使用線段樹維護區間極值
又有處理好相同元素
可在總複雜度 $O(n \log n)$ 的情況下拿到 100 分

下頁討論筆者原本設計的正解
也是筆者主要想讓大家得知的觀念

正解

我們可以用 DAG 表示大小(偏序)關係

利用 sliding window 維護長度 k 的連續序列

配合可以快速二分查找的資料結構

對於每一個 $(a_i,\ i)$ 找到此區間內稍小於它跟稍大於它的 $(a_j,\ j)$, $(a_k,\ k)$

利用這個資訊建邊

在 DAG 上 DP 求出應該填的數字

*注意 相同元素可利用 DSU handle 掉

將會互相影響的同值做成圖上同一點

正解

考此題的重點在於, 筆者想讓大家學到:

1. DAG 好好用
2. DSU handle 相同元素很重要
3. 好啦線段樹很棒~~毒瘤~~啦

AC code(ST) 完整連結

```
/* Handle Input */
vector<int> v(n);
map<int, vector<int>> valmp;
for (int i=0; i<n; ++i) {
    cin >> v[i];
    valmp[v[i]].push_back(i);
}
```

```
/* Solve */
vector<int> ans(n);
for (auto &it: valmp) {
    // Handle the same a_i
    vector<int> &ids = it.second;
    int vn = ids.size();

    for (int i=0, j; i<vn; i=j) {
        // two pointer method
    }
}
```

```
segtree st(n);
auto test = [&](int i) {
    int L = max(0, i-(k-1));
    int R = min(n-1, i+(k-1));
    return st.query(L, R);
};

// two pointer method
int mxlow = test(ids[i]);
for (j=i+1; j<vn && ids[j]-ids[j-1]+1 <= k;
j++) {
    int vi = ids[j];
    // query the extreme value
    mxlow = max(mxlow, test(vi));
}

for (int jj=i; jj<j; jj++) {
    int vi = ids[jj];
    // find answer and update seg tree
    ans[vi] = mxlow+1, st.update(vi, mxlow+1);
}
```

AC code(DAG)

完整連結

```
for (int l = 0, i = 0; i < n; ++i) {
    /* Expired info */
    if (i - l == k) {
        s.erase(a[l]);
        ++l;
    }
    /* Gen DAG */
    auto it = s.lower_bound(pii(a[i].first, 0));
    if (it->first == a[i].first) // Special case
        Union(it->second, a[i].second);
    else {
        make_edge(a[i].second, it->second);
        make_edge((--it)->second, a[i].second); }
    s.insert(a[i]);
}
```

```
/* Output the answer */
for (int i = 0; i < n; ++i)
    cout << dp[find(i)] + 1 << ' ';
cout << endl;
```

```
/* Topological Sorting */
queue <int> q;
for (int i = 0; i < n; ++i)
    if (!inq[find(i)] &&
    !in[find(i)])
        inq[find(i)] = 1,
        q.push(find(i));

while (!q.empty()) {
    int i = q.front(); q.pop();
    for (int j : g[i]) {
        dp[j] = max(dp[j], dp[i] + 1);
        --in[j];
        if (!in[j]) q.push(j);
    }
}
```

pG.

簡單易懂的現代魔法

最速現代魔法使: 郭育愷

現代魔法使: 34 人

出題者:
xiplus

題目概述

《簡單易懂的現代魔法》是2009年的日本動畫，還不錯看。

第一話的標題就是「hello, world」，整部動畫跟程式都有關係，就拿來當故事了，不然你說說純 cout 題是要怎麼寫題敘？

請不要抱怨我們故意把這題放在最後，因為這題是2天前出的，其他題目早就出完了。

AC code

完整連結

```
cout << "hello, world" << endl;
```

CMS系統小知識

本題的標準答案為 "hello, world\n"

你的輸出	CMS預設white-diff checker	本次競賽指定checker
"　hello, world\n"	行首額外空白 AC	WA
“hello,　world\n”	不同”Token (word)”間額外空白 AC (連續非空格字串為一個 Token)	WA
“hello, world　\n”	行尾額外空白 AC	AC
"hello, world"	輸出最後一行缺少換行 AC	AC
“hello, world\n\n\n”	輸出最後一行額外換行 AC	AC
空白換行以外的額外文字 大小寫差異	WA	WA

只是分享小知識，乖乖按照題目格式要求輸出才不會搞到自己

彩蛋


時刻: 12:51:17
G. 簡單易懂的現代魔法 (ModernMagic) 評測頁面
傳送詳細資料
子問題 1
(100 / 100)
評測結果 詳細資訊 執行時間 記憶體使用量
1 全部正確 輸出正確。恭喜獲得成就：現代魔法初學者 0.002 秒 364 KiB
編譯訊息
編譯結果: 編譯成功
編譯時間: 2.739 秒
記憶體使用量: 143 MiB

AC magic code

完整連結

```
mt19937 mt1(7296546);
for (int i = 0; i < 5; i++) {
     printf("%c", mt1() % 26 + 97);
}
printf(", ");

mt19937 mt2(21535470);
for (int i = 0; i < 5; i++) {
     printf("%c", mt2() % 26 + 97);
}
printf("\n");
```