AtCoder Beginner Contest 103

Problema C: /\/\/\/

Prof. Edson Alves - UnB/FGA

AtCoder Beginner Contest 103 – Problem C: /\/\/\/

Problema

A sequence $a_1, a_2, \ldots, a_n$ is said to be /\/\/\/ when the following conditions are satisfied:

- For each $i = 1, 2, \ldots, n-2$, $a_i = a_{i+2}$.
- Exactly two different numbers appear in the sequence.

You are given a sequence $v_1, v_2, \ldots, v_n$ whose length is even. We would like to make this sequence /\/\/\/ by replacing some of its elements. Find the minimum number of elements that needs to be replaced.

Entrada e saída

Constraints

- $2 \leq n \leq 10^5$
- n is even.
- $1 \leq v_i \leq 10^5$
- v_i is an integer.

Input

Input is given from Standard Input in the following format:

$$n$$

$$v_1 \ v_2 \ \ldots \ v_n$$

Output

Print the minimum number of elements that needs to be replaced.

Exemplo de entradas e saídas

Exemplo de Entrada	Exemplo de Saída
4 3 1 3 2	1
6 105 119 105 119 105 119	0
4 1 1 1 1	2

Solução

- A solução consiste em diversas etapas
- A primeira delas é construir o histograma dos elementos que estão nos índices ímpares e dos elementos que estão nos índices pares
- Aqui há um *corner case*: se todos os elementos são iguais, é preciso inserir um valor sentinela no histograma, com número de ocorrências iguais a zero
- A etapa seguinte é computar, para cada elemento distinto do histograma, o custo de tornar todos os valores da subsequência que ele pertence iguais a ele
- Estes custos devem ser armazenados em um vetor de pares, onde o primeiro elemento é o custo e o segundo é o número que preencherá todas as posições

Solução

- Feito isso para ambas subsequências, estes pares devem ser ordenados, do menor para o maior custo
- Por fim, se os elementos de menores custos de ambas subsequências forem distintos, a resposta será a soma destes custos
- Se forem iguais, é preciso ver o que é mais barato: trocar o segundo mais barato da sequência de índices ímpares e o mais barato da sequência dos pares ou o contrário
- Esta solução tem complexidade $O(N \log N)$, devido à construção do histograma e da ordenação dos custos

Solução AC com complexidade $O(N \log N)$

```cpp
#include <bits/stdc++.h>

using namespace std;
using ii = pair<int, int>;

int solve(const vector<int>& vs, int N)
{
    map<int, int> hist[2];
    vector<ii> cost[2];

    for (int k = 0; k < 2; ++k)
    {
        for (int i = k; i < N; i += 2)
            ++hist[k][vs[i]];

        // Corner case: caso todos os elementos sejam iguais, devemos
        // ter uma segunda opção de escolha
        hist[k][100001] = 0;
```

Solução AC com complexidade $O(N \log N)$

```
20          for (auto [v, n] : hist[k])
21              cost[k].emplace_back(N/2 - n, v);
22
23          sort(cost[k].begin(), cost[k].end());
24      }
25
26      enum { ODD = 0, EVEN = 1 };
27
28      auto ans = cost[ODD][0].second == cost[EVEN][0].second ?
29          min(cost[ODD][0].first + cost[EVEN][1].first, cost[ODD][1].first + cost[EVEN][0].first)
30          : cost[ODD][0].first + cost[EVEN][0].first;
31
32      return ans;
33  }
34
35  int main()
36  {
37      ios::sync_with_stdio(false);
```

Solução AC com complexidade $O(N \log N)$

```
39      int N;
40      cin >> N;
41
42      vector<int> vs(N);
43
44      for (int i = 0; i < N; ++i)
45          cin >> vs[i];
46
47      auto ans = solve(vs, N);
48
49      cout << ans << '\n';
50
51      return 0;
52  }
```