MC458A - Laboratório 3: ordenação numa galáxia muito muito distante

1 Introdução

O famoso espaço-porto de Mos Eisley no planeta Tattoine é um dos mais movimentados da galáxia. Ela serve de ponto de parada de muitos viajantes, assim como de muitas companhias responsáveis por transporte de mercadorias para outros planetas.

Uma nave comercial de transporte pode ser carregada com vários containers. Um container é tipicamente muito grande e difícil de mover. Assim, é importante que os containers sejam carregados em uma ordem que permita um descarregamento mais fácil quando a nave fizer a entrega. Tipicamente os containers são colocados em uma esteira e são carregados nesta ordem na nave. O espaço-porto possui um robô (mais para um braço/garra gigante) que pode trocar a posição de dois containers consecutivos/adjacentes na esteira; chamaremos esta operação de **flip**. Este processo é demorado e consome muita energia. Infelizmente, a diretoria não conseguiu contratar nenhum duende usuário da Força que pudesse fazer este serviço sem suar...

Para economizar tempo e energia o espaço-porto gostaria de usar um algoritmo eficiente que "ordenasse" os containers de acordo com a especificação da nave que faz o **menor número possível de flips**. Assim, a diretoria do espaço-porto resolveu contratar um programador de um pequeno planeta azul de um sistema solar pouco conhecido para fazer isto.

2 Especificação de entrada e saída

A entrada é composta por duas linhas:

- A primeira linha contém um inteiro n ($1 \leq n \leq 50000$) que representa o número de containers.
- A segunda linha contém uma permutação $\pi_1, \ldots, \pi_n$ do conjunto $\{1, 2, \ldots, n\}$ representando a configuração dos containers na esteira. O valor π_i indica a ordem em que o container deveria ser carregado na nave.

A saída deve ser o número mínimo de *flips* necessários para ordenar a entrada (há uma quebra de linha após o número).

Observação: não será exigido, mas sua implementação deveria ter complexidade $o(n^2)$. Note que certas entradas exigem $\Omega(n^2)$ flips para ordena-las, mas o trabalho pede apenas

para devolver o número mínimo de *flips*.

Exemplo:

Entrada	Saída
8 3 1 2 6 5 7 8 4	7

Entrada	Saída
10 10 1 5 7 4 6 3 2 9 8	22

3 Implementação e Submissão

- A solução deverá ser implementada em C, C++, Pascal ou Python 2/Python 3. Não é permitido o uso de bibliotecas que não sejam padrão e de flags/diretivas de otimização.
- O programa deve ser submetido no SuSy, com o nome principal **t3** (por exemplo, t3.c).
- O número máximo de submissões é 20.
- A tarefa contém 10 testes abertos e 10 testes fechados. A nota será proporcional ao número de acertos nos testes fechados.

A solução pode ser submetida até o dia 11/10/21.