Яндекс. Тренировки по алгоритмам июнь 2021, занятие 5

E. Красота превыше всего

Ограничение времени	2 секунды
Ограничение памяти	64Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

В парке города Питсбурга есть чудесная аллея, состоящая из N посаженных в один ряд деревьев, каждое одного из K сортов. В связи с тем, что Питсбург принимает открытый чемпионат Байтландии по программированию, было решено построить огромную арену для проведения соревнований. Так, согласно этому плану вся аллея подлежала вырубке. Однако министерство деревьев и кустов воспротивилось этому решению, и потребовало оставить некоторые из деревьев в покое. Согласно новому плану строительства все деревья, которые не будут вырублены, должны образовывать один непрерывный отрезок, являющийся подотрезком исходного. Каждого из K видов деревьев требуется сохранить хотя бы по одному экземпляру. На вас возложена задача найти отрезок наименьшей длины, удовлетворяющий указанным ограничениям.

Формат ввода

В первой строке входного файла находятся два числа N и K ($1 \le N, K \le 250000$). Во второй строке входного файла следуют N чисел (разделенных пробелами), i-ое число второй строки задает цвет i-ого слева дерева в аллее. Гарантируется, что присутствует хотя бы одно дерево каждого цвета

Формат вывода

В выходной файл выведите два числа, координаты левого и правого концов отрезка минимальной длины, удовлетворяющего условию. Если оптимальных ответов несколько, выведите любой.

Пример 1

Ввод	Вывод
5 3 1 2 1 3 2	2 4

Пример 2

Ввод	Вывод
6 4 2 4 2 3 3 1	2 6

Язык Python 3.12.1

Набрать здесь | Отправить файл

```python
from collections import defaultdict

# считываем данные
N, K = map(int, input().split())
trees = list(map(int, input().split()))

uniq_trees = set()
for i in range(1, K+1):
    uniq_trees.add(i)

counter_trees = defaultdict(int)

best_lf, best_rg, best_dist = -1, -1, float('inf')
lf = 0
for rg in range(N):
    if trees[rg] in uniq_trees:
        uniq_trees.remove(trees[rg])
    else:
        counter_trees[trees[rg]] += 1

    if len(uniq_trees) == 0:
        if best_dist > rg - lf:
            best_dist = rg - lf
            best_lf = lf
            best_rg = rg

        while lf < rg and len(uniq_trees) == 0:
            if counter_trees[trees[lf]] > 0:
                counter_trees[trees[lf]] -= 1
                lf += 1
                if best_dist > rg - lf:
                    best_dist = rg - lf
                    best_lf = lf
                    best_rg = rg
            else:
                uniq_trees.add(trees[lf])
                lf += 1

```

Отправить

Предыдущая

Следующая