E1H 多种口味的月饼

题目描述

对于一个数列 $a_1 \cdots a_n$ ，最小化区间长度，使得区间 $[l, r]$ 内每个数都至少出现一次

数据范围

$1 \le n, k \le 10^5$

可以想到 $O(n)$ 或 $O(n \log n)$

解题思路

遍历所有的右端点，即假定给定一个右端点，求出以其为结束的最小的满足条件的区间

同时可以观察到，对于单调递增的右端点 $r_2 > r_1$ ，那么对应的答案 $l_2 \ge l_1$，即区间左端点不减

考虑具体转移，我们记录当前答案区间内每个数出现的次数，假设求得了右端点为 r 的区间答案，对于 $r+1$，只需要尝试不断移动左端点，直至移动之后会导致数字种类数减少即停止，便求得了对于 $r+1$ 的答案，更新最终答案即可

代码实现

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

using namespace std;

using ll = long long;
using db = long double;  // or double, if TL is tight
using str = string;
using u32 = unsigned int;
using u64 = unsigned long long;
using u128 = unsigned __int128;

using pii = pair<int, int>;
using pll = pair<ll, ll>;
using pdd = pair<db, db>;
#define fi first
#define se second

#define vc vector
using vi = vc<int>;
using vb = vc<bool>;
using vll = vc<ll>;
using vd = vc<db>;
using vs = vc<str>;
using vch = vc<char>;
using vpii = vc<pii>;
using vpll = vc<pll>;
using vpdd = vc<pdd>;
using vvi = vc<vc<int>>;
using vvll = vc<vc<ll>>;
using vvch = vc<vc<char>>;
using vvb = vc<vc<bool>>;
using vvpii = vc<vc<pii>>;
using vvpll = vc<vc<pll>>;

const int mod = 998244353;  // 1e9 + 7;
const int INF = 0x3f3f3f3f;
const ll BIG = 1e18;  // not too close to LLONG_MAX
const db PI = acos((db)-1);
const int dx[4]{1, 0, -1, 0}, dy[4]{0, 1, 0, -1};  // for
every grid problem!!
mt19937
rng((uint32_t)chrono::steady_clock::now().time_since_epoch
().count());

#define cl(x, y) memset(x, y, sizeof(x))
#define de(x) cerr << #x << " = " << x << " "
#define del(x) cerr << #x << " = " << x << endl

#define overload3(a, b, c, d, ...) d
#define rep1(a) for(int i = 0; i < (a); i++)
#define rep2(i, a) for (int i = 0; i < (a); i++)
#define rep3(i, a, b) for (int i = (a); i <= (b); i++)
#define per1(a) for(int i = (a) - 1; i >= 0; i--)
#define per2(i, a) for (int i = (a) - 1; i >= 0; i--)
#define per3(i, a, b) for (int i = (a); i >= (b); i--)
#define rep(args...) overload3(args, rep3, rep2, rep1)
(args)
#define per(args...) overload3(args, per3, per2, per1)
(args)
#define for_subset(t, s) \
    for (ll t = (s); t >= 0; t = (t == 0 ? -1 : (t - 1) &
(s)))

#define co cout
#define dl endl
#define ret return
#define pb push_back
#define YES cout << "YES" << endl
#define Yes cout << "Yes" << endl
#define NO cout << "NO" << endl
#define No cout << "No" << endl
```

```
#define all(s) s.begin(), s.end()
#define rall(s) s.rbegin(), s.rend()
#define sz(v) int(v.size())
#define SUM(a) accumulate(all(a), 0ll)
#define MIN(v) (*min_element(all(v)))
#define MAX(v) (*max_element(all(v)))
#define unisort(v) \
    sort(all(v));  \
    v.resize(unique(all(v)) - v.begin())

int popcnt(int x) { return __builtin_popcount(x); }
int popcnt(u32 x) { return __builtin_popcount(x); }
int popcnt(ll x) { return __builtin_popcountll(x); }
int popcnt(u64 x) { return __builtin_popcountll(x); }

#define LOWBIT(x) ((x) & (-x))
// (0, 1, 2, 3, 4) -> (-1, 0, 1, 1, 2)
int topbit(int x) { return (x == 0 ? -1 : 31 -
__builtin_clz(x)); }
int topbit(u32 x) { return (x == 0 ? -1 : 31 -
__builtin_clz(x)); }
int topbit(ll x) { return (x == 0 ? -1 : 63 -
__builtin_clzll(x)); }
int topbit(u64 x) { return (x == 0 ? -1 : 63 -
__builtin_clzll(x)); }
// (0, 1, 2, 3, 4) -> (-1, 0, 1, 0, 2)
int lowbit(int x) { return (x == 0 ? -1 :
__builtin_ctz(x)); }
int lowbit(u32 x) { return (x == 0 ? -1 :
__builtin_ctz(x)); }
int lowbit(ll x) { return (x == 0 ? -1 :
__builtin_ctzll(x)); }
int lowbit(u64 x) { return (x == 0 ? -1 :
__builtin_ctzll(x)); }
template <typename T> T cdiv(T &a, T &b) {
    return a / b + ((a ^ b) > (T)(0) && a % b);
}  // divide a by b rounded up
template <typename T> T fdiv(T &a, T &b) {
    return a / b - ((a ^ b) < (T)(0) && a % b);
}  // divide a by b rounded down

#define LOCAL_TEST freopen("out.txt", "w", stdout)
#define IOS ios::sync_with_stdio(false);cin.tie(0)

const int maxn = 1e5 + 5;

inline void INIT() {

}

inline void SOLVE() {
    int n, k; cin >> n >> k;

    vi arr(n + 1), cnt(k + 1), ans(n + 1, INF);
    int kind = 0, l = 1, r = 0;
    int minLen = INF;

    rep(i, 1, n) {
        r = i;
        cin >> arr[i];
        cnt[arr[i]]++;
        if(cnt[arr[i]] == 1) kind++;

        while(kind == k) {
            if(cnt[arr[l]] >= 2) cnt[arr[l++]]--;
            else break;
        }

        if(kind == k) {
            ans[r - l + 1] = min(ans[r - l + 1], r);
            minLen = min(minLen, r - l + 1);
        }

    }

    if(minLen == INF) co << "NO" << dl;
    else co << ans[minLen] - minLen + 1 << " " << ans[minLen] << dl;

}

int main() {

    IOS;

    INIT();
    int tt = 1;
    cin >> tt;
    while(tt--)
        SOLVE();

    // system("pause");
    return 0;
}
```