Больше подарков хороших и разных

Подгруппа 1. $n \leqslant 100, k \leqslant 10$.

Используем динамическое программирование.$dp[i]$ = наименьшее количество подотрезков на префиксе длины i, при этом $dp[1] = 1$.

Пересчёт для состояния i: $dp[i] = \min(dp[i], dp[j]+1)$, если подотрезок $[j+1, i]$ содержит не более t различных чисел.

Если пересчитывать данную динамику с конца, то можно проверять условие за $O(1)$, поддерживая количество различных чисел в подотрезке. Таким образом получается решение за $O((nk)^2)$ или $O((nk)^2 \log(nk))$

Подгруппа 2. $t = 1$.

Разобъём исходный массив на блоки одинаковых подряд идущих чисел. Пусть количество блоков будет — cnt. Тогда, ответом будет являться $cnt \cdot k$, если первый блок не совпадает с последним, иначе $(cnt - 1) \cdot k + 1$

Доказательство работы Жадного решения.

Рассмотрим некоторое оптимальное разбиение, пусть оно отличается от жадного. Найдём первую позицию в которой есть различия. В таком случае длина подотрезка в оптимальном оптимальном разбиении меньше, чем в жадном. Сдвинем правую границу данного подотрезка до позиции в нашем жадном решении. Разбиение останется корректным и количество подотрезков не увеличится. Продолжая данный процесс, оптимальное разбиение превратится в жадное, причём количество подотрезков не увеличится. Из этого следует, что жадное разбиение является оптимальным.

Подгруппа 3. $n \leqslant 1000, k \leqslant 1000$.

Построим полностью повторённый массив размера nk. С помощью жадного алгоритма наберём минимальное число подарков. Решение за $O(nk)$ или $O(nk \log(nk))$.

Подгруппа 4. $n \leqslant 1500, k \leqslant 10^6$.

Рассмотрим следующий алгоритм — с позиции i в исходном массиве будем жадно набирать подарки до конца текущей стопки. После чего будем добирать из следующей стопки, пока это не увеличивает количество подотрезков. Для каждой позиции i в исходном массиве запомним позицию следующую за той, на которой мы остановимся, а так же количество подотрезков полученных в процессе и назовём эти величины $go[i]$ и $cnt[i]$.

Насчитаем массивы go и cnt за $O(n^2 \log n)$ втупую. Теперь, чтобы посчитать ответ заведём параметр cur, который будет отвечать за позицию в массиве, до которой мы оптимально разбили массив. Идём циклом от 1 до k, на каждой итерации прибавляем к ответу $cnt[cur]$, а cur меняем на $go[cur]$. Подсчёт ответа получается за $O(k)$, прибавляем предпосчёт и получаем $O(n^2 \log n + k)$

Подгруппа 5. $k \leqslant 10^6$.

Научимся находить массивы go и cnt за $O(n)$. Для этого воспользуемся методом двух указателей, чтобы для позиции i находить максимальную длину подотрезка, который содержит не более, чем t чисел. Назовём эту длину — len. Считая массивы go и cnt с конца исходного массива, пересчёт для позиции i можно производить следующим образом:

- Если $i + len \geqslant n - 1$, то $go[i] = (i + len)\%n + 1$, где % операция взятия числа по модулю, а $cnt[i] = 1$
- Иначе, $go[i] = go[i+len+1]$, где % операция взятия числа по модулю, а $cnt[i] = cnt[i+len+1]+1$

Считая ответ как в подгрупппе 4, получаем решение за $O(n + k)$ или $O(n \log n + k)$

Полное решение.

Оптимизируем подсчёт ответа. Заметим, что если $k > n$, то при подсчёте ответа величина cur примет некоторые значения несколько раз. Найдём первое повторение. Дальше найдём количество шагов, за которое произошло это повторение, а так же величину прибавленную к ответу за это время. Разделив оставшееся число стопок на количество шагов, найдём сколько ещё раз произойдет повторение, а оставшиеся в конце стопки обработаем отдельно

Итого решение за $O(n)$ или $O(n \log n)$