БИРЕНО ПАРТИ

Смарти е програмист, който предпочита да си лежи, отколкото да пие бира, което е второто най-важно нещо в живота му (и чак тогава идва ред на програмирането). Един ден той бил поставен пред следния проблем. На бирено парти домакините били подредили в много дълга редица N бутилки бира от K различни вида. Тъй като бил колекционер на празни бирени бутилки и нямал в колекцията си нито една бутилка от тези K вида, той трябвало да направи нещо, за да си ги занесе в къщи. Но единственият начин да си вземе някакво количество бирени бутилки бил, да избере една непрекъсната подредица от тях и да ги изпие. Тъй като е много мързелив и му е много трудно да отваря бутилките, решил да избере най-късата непрекъсната подредица от бутилки, която съдържа поне по една от всичките K различни вида. Смарти е вече достатъчно пиян (нали е на бирено парти) и не може да си спомни алгоритъма, който решава тази задача. Затова ще трябва да му помогнете, като напишете програма J, която определя позициите в редицата на първата и последната бутилки от най-късата непрекъсната подредица, изпълняваща условието. Ако има няколко такива подредици, програмата трябва да намери тази, която се среща най-рано в редицата.

Жокер: Идеята за решение на задачата е подобна донякъде на тази в задачата, която вече решавахме "beers 1". Там имахме два индекса, които се движиха един срещу друг. В тази задача индексите ще се движат в една посока (от началото на масива - позиция 0 - към края му - позиция N-1). Първият (десният) right_index ще напредва докато в интервала между втория (левия) и него не се съдържат всички видове бира. Така си гарантираме че сме намерили оптималната дясна граница (най-лявата дясна граница) на локален интервал съдържащ всички видове бира. След това вторият (левият) индекс left_index ще предвижим надясно до първата възможна позиция, в която вече не се съдържат всичките K вида бири (в [left_index, right_index] ще се съдържат K-1 вида бири). Локалният интервал [left_index - 1, right_index] очевидно (иначе left_index нямаше да достигне сегашната си стойност, а някаква по-малка от нея) съдържа всички видове бира и left_index - 1 е неговата оптимална (най-дясна) лява граница. Дължината на този интервал е right_index - (left_index - 1) + 1 = right_index - left_index + 2. Минимизираме тази дължина с променлива best_result (инициализирана с MAX_INT), в която постепенно ще намерим крайния отговор: best_result = min(best_result, right_index - left_index + 2). След това повтаряме отново движение на дясна граница, движение на лява и минимизиране на дължината на новия

интервал [left_index - 1, right_index] с текущия най-добър резултат best_result. Повторенията продължават докато дясната граница не достигне края на масива и лявата не я догони за последно. Променливата best_result ще ни даде отговора на задачата.

Input Format

Програмата трябва да прочете от първия ред на стандартния вход броят T на ситуациите, които трябва да обработи. Данните за всяка ситуация са в два реда на стандартния вход. На първия са записани числата N и K разделени с интервал, а на втория – N числа (от 1 до K), разделени с по един интервал.

Constraints

$1 \leq N \leq 10000000$
$1 \leq K \leq 13000$

Output Format

За всяка ситуация програмата трябва да изведе на стандартния изход две числа – номерата на началната и крайната бутилки в подредицата, която трябва да изпие Смарти.

Sample Input 0

```
2
5 3
1 3 3 2 1
5 3
1 1 2 2 3
```

Sample Output 0

```
3 5
2 5
```